SIAM J. MATRIX ANAL. APPL.
Vol. 23, No. 4, pp. 1025–1044

THE CENTROID DECOMPOSITION: RELATIONSHIPS BETWEEN DISCRETE VARIATIONAL DECOMPOSITIONS AND SVDs*

MOODY T. CHU† AND ROBERT E. FUNDERLIC‡

Abstract. The centroid decomposition, an approximation for the singular value decomposition (SVD), has a long history among the statistics/psychometrics community for factor analysis research. We revisit the centroid method in its original context of factor analysis and then adapt it to other than a covariance matrix. The centroid method can be cast as an $\mathcal{O}(n)$-step ascent method on a hypercube. It is shown empirically that the centroid decomposition provides a measurement of second order statistical information of the original data in the direction of the corresponding left centroid vectors. One major purpose of this work is to show fundamental relationships between the singular value, centroid, and semidiscrete decompositions. This unifies an entire class of truncated SVD approximations. Applications include semantic indexing in information retrieval.

Key words. data matrix, loading matrix, scoring matrix, indexing matrix, factor analysis, centroid method, singular value decomposition, low rank approximation, semidiscrete decomposition, centroid decomposition, low rank decompositions, integer programming

AMS subject classifications. 15A21, 65F30, 62H25, 15A23, 68Q25

PII. S0895479800382555

1. Introduction.

We first review factor analysis [5, 9, 7] in the terms used by the applied statistics/psychometrics (AS/P) community with the notation of numerical linear algebra. This provides a setting to show how the centroid method developed as an approximate singular value decomposition (SVD). Our work was motivated by a recent article [11] and correspondence from Lawrence Hubert drawing our attention to the application of SVDs in the AS/P context. In particular we have found Horst's [9] description of the centroid decomposition proposed in the AS/P literature quite illuminating. The use of the SVD or ideas associated with it has a rich history [7] in the AS/P community dating back at least to Pearson [15] in 1901. Stewart's scholarly historical treatise [16] has traced the early history of the SVD back to Beltrami in 1873 and Jordan in 1874. Within the numerical linear algebra (NLA) community, besides Hotelling's work [10] and that of Eckert and Young [6], there seems little awareness of the AS/P work. The AS/P community generally considers Thurston's 1931 paper [17] as being the most complete description of the centroid method. In point of fact the centroid method was used in 1917 by Burt [2]. So what turned out to be an approximation for the SVD had its beginnings before there was widespread knowledge of the SVD itself.

We begin in section 2 with the factor analysis setting, providing a brief but practical background for further understanding of the underlying matrix decompositions. This should unify the differences in vocabulary and notation used by the AS/P and NLA communities. The classical Wedderburn rank reduction formula has been used by the AS/P community at least since the early 1940s. In section 3 we show how they

*Received by the editors December 14, 2000; accepted for publication (in revised form) by L. Eldén October 18, 2001; published electronically April 10, 2002.
http://www.siam.org/journals/simax/23-4/38255.html
†Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205 (chu@math.ncsu.edu). This author's research was supported in part by the National Science Foundation under grants DMS-9803759 and DMS-0073056.
‡Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8206 (ref@csc.ncsu.edu).

connect the rank reduction formula with the centroid method, which provides insight into the reduced matrix. What they call centroid factors are indeed the centroids of a sequence of the orthogonally reduced loading matrices. Section 4 further develops the centroid method with the necessary modifications for the reduction of the covariance matrix. Section 5 gives the details of the centroid algorithm along with an ascent hypercube description, proof of convergence, and computational complexity. Section 6 provides a general stochastic treatment of the truncated SVD and thereby the statistical soundness of the centroid decomposition. Section 7 compares the similar yet different setups between the centroid method and some latent semantic indexing techniques used in data mining. Section 8 discusses a modified centroid algorithm that does not require the explicit formation of the product moment or covariance matrix. Finally, in section 9 we use the SVD triad of variational formulations to unify a class of approximations to the SVD, including the data retrieval semidiscrete decomposition (SDD) and the centroid decomposition.

2. The factor analysis setting. An indispensable task in almost every discipline of science is the analysis of data in search of relationships between sets of externally caused and internal variables. Such a task has become especially important in this era of information and digital technologies, when massive amounts of data are being generated at almost all levels of applications. In many situations, the digitized information is gathered and stored as a data matrix. Quite often the data observed from complex phenomena represent the integrated result of several interrelated variables acting together. When these variables are less precisely defined, it becomes important to distinguish which variable is related to which and how the variables are related before deductive sciences can further be applied. Toward that end, factor analysis is a class of procedures that can help identify and test what constructs might be used to explain the interrelationships among the variables.

Let $Y = [y_{ij}] \in \mathbb{R}^{n\times\ell}$ denote the matrix of observed data. One of the main applications of factor analysis is to analyze relationships between questions on tests. Thus we will use here, as is done for almost any application, y_{ij} to represent, in a broad sense, the *standard score* of variable i on entity j. By a standard score we mean that a raw score has been normalized to have mean 0 and standard deviation 1. The matrix

$$R := \frac{1}{\ell} YY^T, \tag{2.1}$$

therefore, represents the correlation matrix of all n variables. Note that $r_{ii} = 1$ and $|r_{ij}| \le 1$ for all $i, j = 1, \ldots, n$.

In a linear model, it is assumed that the score y_{ij} is a linearly weighted score of entity j on several factors. That is, we assume

$$Y = AF, \tag{2.2}$$

where $A = [a_{ik}] \in \mathbb{R}^{n\times m}$ is a matrix with a_{ik} denoting loadings of variable i on factor k, and $F = [f_{kj}] \in \mathbb{R}^{m\times\ell}$ with f_{kj} denoting the score of factor k on entity j. To better grasp the notion of linear modeling in (2.2), readers might want to think, for example, that each of the ℓ columns of the observed matrix Y represents the transcript of a college student (an entity) at his/her freshman year on n fixed subjects (the variables), e.g., calculus, English, chemistry, and so on. It is generally believed that a college freshman's academic performance depends on a number of factors, including, for instance, family social status, finances, high school GPA, cultural background, and

so on. Upon entering the college, each student could be asked to fill out a questionnaire inquiring about these factors of his/her background. In turn, individual responses to those factors are translated into scores and placed in the corresponding column of the scoring matrix F. What is not clear to the educators/administrators is how to choose the factors to compose the questionnaire or how each of the chosen factors would be weighted (the loadings) to reflect the effect on each particular subject. In practice, we usually do not have a priori knowledge about the number and character of underlying factors in A. Sometimes we do not even know the factor scores in F. Only the data matrix Y is observable. Explaining the complex phenomena observed in Y with the help of a minimal number of factors extracted from the data matrix is the primary and most important goal of factor analysis.

It is customary to assume that all sets of factors being considered are uncorrelated with each other. If we further assume, similar to Y, that the scores in F for each factor are normalized, then it is true that

$$\frac{1}{\ell} F F^T = I_m, \tag{2.3}$$

where I_m stands for the identity matrix in $\mathbb{R}^{m\times m}$. It follows that the correlation matrix R can be expressed directly in terms of the loading matrix A, i.e.,

$$R = AA^T. \tag{2.4}$$

Factor extraction now becomes a problem of decomposing the correlation matrix R into the product AA^T using as few factors as possible.

As a whole, the ith row of A may be interpreted as how the data variable i is weighted across the list of current factors. If the sum of squares of this row, called the *communality* of variable i, is small, it suggests that this specific variable is of little consequence to the current factors. On the other hand, the kth column of A may be interpreted as correlations of the data variables with that particular kth factor. Those data variables with high factor loadings are considered to be more like the factor in some sense, and those with zero or near-zero loadings are treated as being unlike the factor. The quality of this likelihood, which we call the *significance* of the corresponding factor, is measured by the norm of the kth column of A. One basic idea in factor analysis is to rewrite the loadings of variables over some newly selected factors so as to manifest more clearly the correlation between variables and factors. Suppose the newly selected factors are expressed in terms of columns of the orthogonal matrix

$$V := [\mathbf{v}_1, \ldots, \mathbf{v}_m] \in \mathbb{R}^{m\times m}. \tag{2.5}$$

Then this rewriting of factor loadings with respect to V is mathematically equivalent to a change of basis, i.e., A is now written as $B := AV$. One of the fundamental problems in the practice of factor analysis is to determine some appropriate new basis for V. Note that because $VV^T = I_m$, the very same observed data now is decomposed as $Y = AF = (AV)(V^T F) = BG$ with B and $G = V^T F$ representing, respectively, the factor loadings and uncorrelated standard factor scores corresponding to the factors in V. From this we also see that the correlation matrix $R = AA^T = BB^T \in \mathbb{R}^{n\times n}$ is independent of factors selected. This is another reason that in the process of defining new factors it is often desirable to retrieve information directly from the correlation matrix R rather than from any particular loading matrix A. The centroid method,

the main topic of this paper, has been used for retrieving such factors. The new factors in the centroid method were defined via successive rank reduction applied to the correlation matrix R.

3. Centroid factor. We shall denote $A_1 := A$, $R_1 := R$, and call the relationship $R_1 = A_1 A_1^T$ the *product moment* of A_1. Temporarily assuming that a loading matrix A_1 is given, the coordinate axes in $\mathbb{R}^m$ represent a set of m abstractly defined factors. The centroid method amounts to a procedure of defining a new coordinate system representing what are called the *centroid factors*. The most important feature of the centroid method is that loadings with respect to the centroid factors can be calculated without the knowledge of A_1 or even of the centroid factors. The assumption of knowing A_1 a priori, therefore, is not needed. But in the following we continue to use A_1 to gain insight into the meaning of the extraction steps.

The ith row in the matrix A_1 denotes the loadings of variable i across the spectrum of the current set of factors. Denoting each row of A_1 as a point in the factor space $\mathbb{R}^m$, the arithmetic mean of these points *might* be used to indicate a collective trend of the variables. That direction could be a substantial factor to be weighted in and thus constitutes the essential idea of a centroid factor. Before we move into details, we should comment that generally variables that tend to vary together form clusters. If all the variables are truly independent, there should be no clusters at all. On the other extreme, if all the variables are dependent on the same factor, then a single cluster should be formed. In between the two extremes, we do not know a priori how many clusters are to be expected. There are many cluster detection techniques. See, for example, the book [1]. Among these, the so called *k-means method* is perhaps the most commonly used in practice. The centroid method we are about to describe is the simplest special variation of the k-means method. The centroid method iteratively searches for one mean a time. Since the goal of this paper is to compare the relationships of various discrete variational decompositions with the SVD, our present discussion will be concentrating on the (1-mean) centroid method only. The generalization of comparison to a k-means method should be another interesting research topic in the future.

Given $A_1 \in \mathbb{R}^{n\times m}$, the centroid of these n variables is given by the column vector

$$\mathbf{c}_1 := \frac{A_1^T \mathbf{1}_n}{n} = \left[\frac{\sum_{i=1}^n a_{i1}}{n}, \ldots, \frac{\sum_{i=1}^n a_{im}}{n}\right]^T, \tag{3.1}$$

where $\mathbf{1}_n$ denotes the column vector $\mathbf{1}_n := [1, \ldots, 1]^T \in \mathbb{R}^n$. The first *centroid factor* is defined to be the normalized vector

$$\mathbf{v}_1 := \frac{\mathbf{c}_1}{\|\mathbf{c}_1\|}. \tag{3.2}$$

The new loadings of variables with respect to this new factor $\mathbf{v}_1$, i.e., the first column $\mathbf{b}_1 = [b_{11}, \ldots, b_{n1}]^T$ of the new loading matrix B (which is yet to be found), can be calculated without referring to A_1 as follows: Each component b_{j1} is precisely the projection component of variable j along the unit vector $\mathbf{v}_1$, i.e., $\mathbf{b}_1 = A_1\mathbf{v}_1$. This can be rewritten as

$$\mathbf{b}_1 = A_1 \frac{A_1^T \mathbf{1}_n}{\|A_1^T \mathbf{1}_n\|} = \frac{R_1 \mathbf{1}_n}{\sqrt{\mathbf{1}_n^T R_1 \mathbf{1}_n}}. \tag{3.3}$$

In this way, we note that the first loading vector $\mathbf{b}_1$ is extracted directly from R_1. No reference to A_1 or $\mathbf{v}_1$ is needed.

Once the loadings $\mathbf{b}_1$ for the centroid factor $\mathbf{v}_1$ are found, the product moment R_1 is conventionally updated to a new matrix R_2 defined by

$$R_2 := R_1 - \frac{R_1 \mathbf{1}_n \mathbf{1}_n^T R_1}{\mathbf{1}_n^T R_1 \mathbf{1}_n}. \tag{3.4}$$

It is important to understand the meaning of R_2. Define

$$A_2 := A_1 - A_1 \mathbf{v}_1 \mathbf{v}_1^T. \tag{3.5}$$

Observe that each row in the matrix A_2 represents the component of the original loadings A_1 in the direction orthogonal to $\mathbf{v}_1$. We shall call A_2 the *orthogonally reduced loading matrix* of A_1 with respect to $\mathbf{v}_1$. Note that A_2 inherits most of the loading information of the original A_1 except for the loadings along the direction $\mathbf{v}_1$. The information along $\mathbf{v}_1$ is subtracted from A_1 to form A_2. The following statement provides an interesting interpretation of R_2.

THEOREM 3.1 (see [3]). *The traditional rank-one update* (3.4) *from* R_1 *is simply another way to calculate the product moment of the orthogonally reduced loading matrix* A_2 *without directly referring to* A_2.

Proof. The product moment of A_2 can be computed as follows:

$$\begin{aligned} A_2 A_2^T &= \left(A_1 - A_1 \mathbf{v}_1 \mathbf{v}_1^T\right)\left(A_1^T - \mathbf{v}_1 \mathbf{v}_1^T A_1^T\right) \\ &= A_1 A_1^T - A_1 \mathbf{v}_1 \mathbf{v}_1^T A_1^T \\ &= R_1 - \frac{R_1 \mathbf{1}_n \mathbf{1}_n^T R_1}{\mathbf{1}_n^T R_1 \mathbf{1}_n}, \end{aligned}$$

where the last equality follows from (3.1). ∎

With A_2 or R_2 in hand, it seems that the above procedure can be repeated to extract the next centroid factor for A_2, to introduce the next reduced loading matrix, and so on. Unfortunately, this is not the case. The procedure cannot be repeated because $A_2^T \mathbf{1}_n = \mathbf{0}_m$. In other words, because the centroid of A_2 is residing squarely at the origin of $\mathbb{R}^m$, the second centroid factor is null. We have to modify the notion of centroid somewhat to circumvent this situation.

It is worth mentioning that the update (3.4) is simply one special case of the well-known Wedderburn rank reduction formula [4]. The rank of R_2 is precisely one less than that of R_1.

4. Modified centroid factor. In factor analysis, one major task is to ascribe the loadings in A_1 to as few *essential factors* as possible. We consider that a factor is essential if loadings with respect to that particular factor are relatively weighty. Being the average of all variables, the centroid factor $\mathbf{v}_1$ would delineate an essential factor under the following circumstances:

1. When all points in $\mathbb{R}^m$ representing rows of A_1 stay near the line determined by $\mathbf{v}_1$: In this case, each variable is approximately a scalar multiple of $\mathbf{v}_1$. The scalar can be positive or negative, indicating a positive or negative linear correlation between the variable and the factor $\mathbf{v}_1$. In either case, it is clear that a substantial portion of loadings in A_1 should be attributed to the factor $\mathbf{v}_1$.
2. When the centroid $\mathbf{c}_1$ is far away from the origin: In this case, the variables are asymmetrically distributed in the factor space $\mathbb{R}^m$. The quantity $\|\mathbf{c}_1\|$

measures, in some sense, the *eccentricity* of the system of variables with respect to the origin. That is, the farther $\mathbf{c}_1$ is away from the origin, the more variables are qualitatively scattered in a general area surrounding $\mathbf{c}_1$. Thus the larger $\|\mathbf{c}_1\|$ is, the better an essential factor $\mathbf{v}_1$ represents.

It is worth noting again, as we have already pointed out in the first remark above, that replacing one particular variable by its negative does not cause trouble in the identification of an essential factor. We therefore should change the sign of certain rows if that helps to bring out other properties such as that described in the second remark above. On the other hand, the scalar

$$\mathbf{1}_n^T R \mathbf{1}_n = \|A_1^T \mathbf{1}_n\|^2 = n^2 \|\mathbf{c}_1\|^2 \tag{4.1}$$

is a fixed multiple of $\|\mathbf{c}_1\|$. Combining these observations, we are motivated to consider the integer programming problem

$$\max_{|\mathbf{z}|=1} \mathbf{z}^T R_1 \mathbf{z}, \tag{4.2}$$

where $|\mathbf{z}| = 1$ means the components of the column vector $\mathbf{z}$ are either 1 or -1. We call $\mathbf{z}$ a *sign vector*. There are only 2^n sign vectors for a fixed n. Without causing any ambiguity, we shall use the same notation to represent the vectors

$$\mathbf{c}_1 := \frac{A_1^T \mathbf{z}_1}{n}, \tag{4.3}$$

$$\mathbf{v}_1 := \frac{A_1^T \mathbf{z}_1}{\|A_1^T \mathbf{z}_1\|}, \tag{4.4}$$

where $\mathbf{z}_1$ is the optimizer of (4.2), and call them the *modified centroid* and the *modified centroid factor*, respectively. For later reference, we shall call

$$\mu_1 := \frac{1}{n} \max_{|\mathbf{z}|=1} \mathbf{z}^T R_1 \mathbf{z} \tag{4.5}$$

the first *centroid value* of A_1. The following results are generalizations of (3.3) and Theorem 3.1.

THEOREM 4.1. *The loading* $\mathbf{b}_1$ *with respect to the modified centroid factor* $\mathbf{v}_1$ *defined by* (4.4) *is given by the projection* $\mathbf{b}_1 = A_1 \mathbf{v}_1$ *and can be computed by*

$$\mathbf{b}_1 = \frac{R_1 \mathbf{z}_1}{\sqrt{\mathbf{z}_1^T R_1 \mathbf{z}_1}}. \tag{4.6}$$

The product moment R_2 *of the orthogonally reduced loading matrix* $A_2 = A_1 - A_1 \mathbf{v}_1 \mathbf{v}_1^T$ *can be computed by*

$$R_2 = R_1 - \frac{R_1 \mathbf{z}_1 \mathbf{z}_1^T R_1}{\mathbf{z}_1^T R_1 \mathbf{z}_1}. \tag{4.7}$$

We remark again that in the above expression both $\mathbf{b}_1$ and R_2 can be calculated without making explicit reference to A_1. By now, it should be clear that the notion of modified centroid factor can be applied to R_2 to induce the next R_3, and so on. With this generalization, we should also point out that henceforth the matrix R no longer denotes a correlation matrix but rather a general symmetric and positive semidefinite

matrix. Each application of this centroid factor retrieval will reduce the rank of the loading matrix by one. The procedure therefore has to come to a stop in finitely many steps. In this way, with the recurrence

$$A_i = A_{i-1} - A_{i-1}\mathbf{v}_{i-1}\mathbf{v}_{i-1}^T, \quad i = 2, \ldots, \gamma, \tag{4.8}$$

where $\mathbf{v}_i$ is the modified centroid factor of A_i, γ is the rank of A_1, and with the loadings $b_i = A_i\mathbf{v}_i$, we may write

$$A = A_1 = \mathbf{b}_\gamma \mathbf{v}_\gamma^T + \cdots + \mathbf{b}_1\mathbf{v}_1^T, \tag{4.9}$$

which we will call a *centroid decomposition* of A.

Let $\mathbf{z}_2$ be the sign vector that maximizes $z^T R_2 z$. Note that $\mathbf{z}_2 \neq \mathbf{z}_1$ because $R_2\mathbf{z}_1 = 0$. The modified centroid $\mathbf{c}_2$ for A_2, according to (4.3), should be $\mathbf{c}_2 = \frac{A_2^T\mathbf{z}_2}{n}$. It is interesting to note that

$$\mathbf{c}_1^T\mathbf{c}_2 = \frac{1}{n^2}\left(\mathbf{z}_1^T A_1\right)\left(A_2^T\mathbf{z}_2\right) = \frac{1}{n^2}\left(\mathbf{z}_1^T A_1\right)\left[A_1^T\left(\mathbf{z}_2 - \frac{\mathbf{z}_1^T R_1\mathbf{z}_2}{\mathbf{z}_1^T R_1\mathbf{z}_1}\mathbf{z}_1\right)\right] = 0, \tag{4.10}$$

i.e., the modified centroids (and factors) are mutually orthogonal even though they are not explicitly calculated.

5. Centroid method. To perform the centroid decomposition, a sequence of integer programming problems such as (4.2) must be solved. The feasible set consists of 2^n sign vectors. An exhaustive search would be expensive. Fortunately, an interesting quick iterative approach, called the *centroid method*, has been developed in the AS/P literature for solving the underlying maximization problem. We shall briefly review the centroid method in this section. In particular, we want to provide a geometric interpretation of the centroid method.

Upon identifying -1 as 0 and keeping 1 as 1, we can associate a unique binary tag to each sign vector. Each binary tag, in turn, is translated into a unique integer between 0 and $2^n - 1$ that provides a natural ordering of the sign vectors. For example, sign vectors $[-1,-1,-1,-1]^T$ and $[-1,1,-1,1]^T$ have binary tags 0000 and 0101 and are the 0th and the 5th in the order, respectively. If we consider each sign vector as one node connected to all other sign vectors whose binary tags differ from its own by exactly one bit, then topologically the set of 2^n sign vectors can be identified as an n-dimensional *hypercube*. A 4-dimensional hypercube layout together with the ordering of its vertices is depicted in Figure 5.1. Note (see Figure 5.1) that each n-dimensional hypercube consists of two $(n-1)$-dimensional subhypercubes where one subhypercube is simply a *bit reversal* of the other. The objective values $z^T Rz$ therefore always appear in pairs.

The integer programming problem over sign vectors now becomes the maximization of $\mathbf{z}^T R\mathbf{z}$ over vertices on the hypercube. Without causing any ambiguity, let R stand for any of the product moments R_i involved in the process. Write $R = [r_{ij}] = P + \mathrm{diag}(\mathrm{diag}(R))$. Since $\mathbf{z}^T R\mathbf{z} = \mathbf{z}^T P\mathbf{z} + \sum_{i=1}^n r_{ii}$, it suffices to consider the problem of maximizing

$$f(\mathbf{z}) := \mathbf{z}^T P\mathbf{z}$$

with $|\mathbf{z}| = 1$. The classical centroid method is described next.

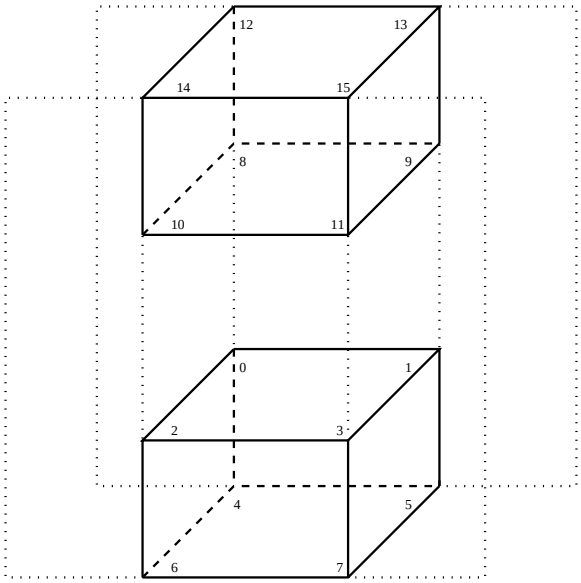

FIG. 5.1. *Topology of a 4-dimensional hypercube.*

ALGORITHM 5.1. *Given any sign vector* $\mathbf{z}^{(0)}$ *and machine zero threshold* ϵ*, define* $\mathbf{w}^{(0)} := P\mathbf{z}^{(0)}$*. Repeat the following steps for* $i = 0, 1, \ldots$:

1. *If* $sgn(\mathbf{w}_k^{(i)}) = sgn(\mathbf{z}_k^{(i)})$ *for all* $k = 1, \ldots, n$*, then stop; otherwise, choose* k *so that* $|\mathbf{w}_k^{(i)}| > \epsilon$ *and is the largest among all* $|\mathbf{w}_j^{(i)}|$*'s where* $sgn(\mathbf{w}_j^{(i)}) \neq sgn(\mathbf{z}_j^{(i)})$.
2. *Define* $\mathbf{z}^{(i+1)}$ *by simply changing the sign of* $\mathbf{z}_k^{(i)}$.
3. *Define* $\mathbf{w}^{(i+1)} := \mathbf{w}^{(i)} + 2sgn(\mathbf{z}_k^{(i+1)})P(:, k)$.

Since at most one bit is changed in each cycle, it is seen from the above that the centroid method involves advancing from one node to one of its neighboring nodes on the hypercube. The convergence behavior of this algorithm can be seen from the following result.

THEOREM 5.1. *The sequence* $\{f(\mathbf{z}^{(i)})\}$ *where* $\mathbf{z}^{(i)}$ *is generated by the centroid method from any starting value* $\mathbf{z}^{(0)}$ *is finite and increasing.*

Proof. We can rewrite the definition of $\mathbf{z}^{(i+1)}$ as

$$\mathbf{z}^{(i+1)} := \mathbf{z}^{(i)} - 2\text{sgn}(\mathbf{z}_k^{(i)})\mathbf{e}_k,$$

where $\mathbf{e}_k$ is the standard kth unit vector. Observe

$$\begin{aligned} f(\mathbf{z}^{(i+1)}) &= \left(\mathbf{z}^{(i)} - 2\text{sgn}(\mathbf{z}_k^{(i)})\mathbf{e}_k\right)^T P \left(\mathbf{z}^{(i)} - 2\text{sgn}(\mathbf{z}_k^{(i)})\mathbf{e}_k\right) \\ &= f(\mathbf{z}^{(i)}) - 4\text{sgn}(\mathbf{z}_k^{(i)})(\mathbf{e}_k^T P\mathbf{z}^{(i)}) \\ &= f(\mathbf{z}^{(i)}) - 4\text{sgn}(\mathbf{z}_k^{(i)})\mathbf{w}_k^{(i)}. \end{aligned}$$

Note that, by the definition of k, the second term in the last equality is negative, showing that $f(\mathbf{z}^{(i+1)})$ is strictly larger than $f(\mathbf{z}^{(i)})$ by $4|\mathbf{w}_k^{(i)}|$. The centroid method

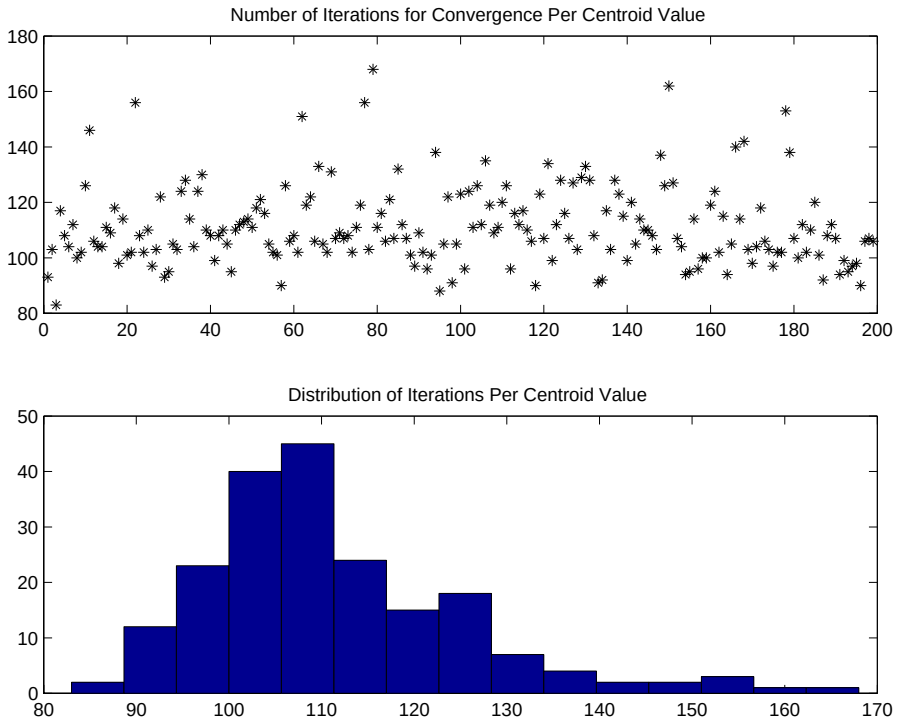

FIG. 5.2. *Number of steps per centroid value in the centroid method for a matrix of size* 200.

can be regarded as a steepest ascent method along the nodes of the hypercube. There are only finitely many nodes; the sequence therefore has to converge in finitely many steps. □

Although there are 2^n nodes on an n-dimensional hypercube, to go from one node to another node in order to maximize $\mathbf{z}^T R\mathbf{z}$ is not an NP problem. Indeed, recall that each node is identified by a binary tag of length n. Recall also that the centroid method (Algorithm 5.1) has the unique feature of changing only one bit at a time and never descends. The worst scenario is that the iteration moves from one binary tag, say 1010 in the case $n = 4$, to its bit reversal tag, say 0101. In other words, it takes at most n iterations to locate a maximum. We can prove by induction that the expected number of iterations required for convergence to a centroid value is in fact $\frac{n}{2}$. To illustrate this point, we report in Figure 5.2 just one of the many numerical simulations we have conducted on the number of iterations needed to generate each centroid value. The lower graph in Figure 5.2, depicting the histogram of these number of iterations, suggests that the mean is about 100.

We conclude this section by cautioning that the centroid method only finds a *local* maximum. Even after excluding the parity resulting from bit reversal mentioned before, the local maximum may not be unique for a given P. For example, with

$$P = \begin{bmatrix} 0 & 3.5 & 3 & 1 \\ 3.5 & 0 & -4 & -3 \\ 3 & -4 & 0 & -3.5 \\ 1 & -3 & -3.5 & 0 \end{bmatrix},$$

the objective value $z^T Pz$ has local maxima at the 1st, 2nd, and 4th sign vectors. The mechanism built in the first step of the centroid method dictates that the algorithm converges to the 2nd (or its bit reversal, 14th) sign vector $[-1,-1,-1,1]^T$ unless the starting value happens to be the other two local maximizers, in which case the algorithm stalls right there.

6. Relationship to truncated matrices. In the practice of information retrieval, quite often the original data matrix Y is not exact due to noise. It is often sufficient to replace Y by a simpler approximation. This approximation matrix is obtained by *truncating* the original matrix in some sense. In this section we shall provide a statistical meaning of truncation. At the end of this section we establish the statistical soundness of the centroid method and compare its decomposition with the SVD. In contrast,we shall see in the next section that the SDD [12], though approximating a SVD decomposition, does not readily imply the same desirable stochastic meaning.

We first consider a general random variable $\mathcal{X}$ in $\mathbb{R}^n$. Let $\mathcal{E}[\mathcal{X}]$ denote the expected value of $\mathcal{X}$. Typically, $\text{cov}(\mathcal{X}) := \mathcal{E}[(\mathcal{X}-\mathcal{E}[\mathcal{X}])(\mathcal{X}-\mathcal{E}[\mathcal{X}])^T] \in \mathbb{R}^{n\times n}$ is defined as the *covariance matrix* of $\mathcal{X}$. Let

$$\text{cov}(\mathcal{X}) = \sum_{j=1}^{n} \lambda_j \mathbf{u}_j \mathbf{u}_j^T \tag{6.1}$$

denote the spectral decomposition of $\text{cov}(\mathcal{X})$ with eigenvalues arranged in the descending order $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Note that $\mathbf{u}_1, \ldots, \mathbf{u}_n$ form an orthonormal basis for $\mathbb{R}^n$. Express the random column variable $\mathcal{X}$ as

$$\mathcal{X} = \sum_{j=1}^{n} (\mathbf{u}_j^T \mathcal{X}) \mathbf{u}_j. \tag{6.2}$$

Note that the columns in the matrix $U := [\mathbf{u}_1, \ldots, \mathbf{u}_n]$ are deterministic vectors themselves. The randomness of $\mathcal{X}$ therefore must come solely from the randomness of each coefficient in (6.2). The following observation in [3] sheds important insight on the portion of randomness of $\mathcal{X}$ in each eigenvector direction of $\text{cov}\mathcal{X}$.

THEOREM 6.1. *Let $\boldsymbol{\alpha} := U^T \mathcal{X}$. Then $\boldsymbol{\alpha}$ is a random variable whose components are mutually stochastically independent. Indeed,*

$$\mathcal{E}[\boldsymbol{\alpha}] = U^T \mathcal{E}[\mathcal{X}], \tag{6.3}$$

$$cov(\boldsymbol{\alpha}) = \text{diag}\{\lambda_1, \ldots, \lambda_n\}. \tag{6.4}$$

In other words, the larger the eigenvalue λ_j of $\text{cov}(\mathcal{X})$ is, the larger the variance of α_j is, i.e., the more stochastic properties such as randomness the vector $\alpha_j \mathbf{u}_j$ contributes to $\mathcal{X}$. From (6.2) it appears intuitive that those coefficients α_j with larger variance represent a more integral part of $\mathcal{X}$. We therefore can *rank* the importance of corresponding eigenvectors u_j as essential components for the variable $\mathcal{X}$ according to the magnitude of λ_j.

If it becomes desirable to approximate the random variable $\mathcal{X}$ by another unbiased yet *simpler* variable $\hat{\mathcal{X}}$, we see from Theorem 6.1 that $\hat{\mathcal{X}}$ had better capture those components corresponding to larger λ_j in the expression (6.2). Indeed, it is entirely sensible to require that $\text{cov}(\hat{\mathcal{X}})$ be reasonably close to $\text{cov}(\mathcal{X})$. We quantify this notion with the following theorem, which provides the basic idea of truncation. The proof can be found in [3], which uses results from [13, 14].

THEOREM 6.2. *Suppose that $\mathcal{X}$ is a random variable in $\mathbb{R}^n$ with mean zero and that its covariance matrix has a spectral decomposition given by* (6.1). *Then among all unbiased variables restricted to any r-dimensional subspaces in $\mathbb{R}^n$, the random variable*

$$\hat{\mathcal{X}} := \sum_{j=1}^{r} (\mathbf{u}_j^T \mathcal{X}) \mathbf{u}_j \tag{6.5}$$

is the best approximation to $\mathcal{X}$ in the sense that $\|cov(\hat{\mathcal{X}}) - cov(\mathcal{X})\|$ is minimized. In addition, $\hat{\mathcal{X}}$ is the best linear minimum-variance estimate of $\mathcal{X}$ in the sense that $\mathcal{E}[\|\hat{\mathcal{X}} - \mathcal{X}\|^2]$ is minimized.

It is important to note that in the above linear minimum-variance estimation, the variable $\mathcal{X}$ is *centered* at zero. If $\mathcal{X}$ is not centered at zero, the expression for truncation would be much more complicated. Without the centering, the mere truncated data in the form of a low rank approximation would suffer from the loss of some significant statistical meanings.

The above observation is based on the fact that the random variable $\mathcal{X}$ is completely known. Such an assumption is not practical in reality since often the probability distribution function of $\mathcal{X}$ is not known a priori. One common practice in applications then is to simulate the random variable $\mathcal{X}$ by a collection of ℓ random samples. These samples are recorded in an $n \times \ell$ matrix. Our data matrix $Y = [y_{ij}]$ is precisely such an example where each column of Y represents one random sample of (standard) score for a certain random variable $\mathcal{X} \in \mathbb{R}^n$ which, in this case, has mean zero. It is known that when ℓ is large enough, many of the stochastic properties of $\mathcal{X}$ can be recouped from Y.

The question now is how to retrieve a sample data matrix from Y to represent the truncated variable $\hat{\mathcal{X}}$. The connection lies in the observations that the matrix R is close to $\mathrm{cov}(\mathcal{X})$ by the law of large numbers. Note that the eigenvalues of R are precisely the squares of the singular values of $Y/\sqrt{\ell}$ and that the singular values, by Theorem 6.1, measure the degree of randomness (of $\mathcal{X}$) in the direction of the left singular vectors (of Y). In the spirit of truncation described in Theorem 6.2 above, the data matrix $\hat{Y}$ for $\hat{\mathcal{X}}$ should be such that both $\|Y - \hat{Y}\|$ and $\|YY^T - \hat{Y}\hat{Y}^T\|$ are minimized. It turns out that the truncated SVD of Y by throwing away its smaller singular values satisfies precisely these requirements. See [3] for a more detailed discussion. In this way, we understand now that the truncated SVD $\hat{Y}$ not only is the best approximation to Y in the sense of norm but more importantly is the closest approximation to Y in the sense of statistics. It maintains the most significant stochastic portion of the original data matrix Y. Generally speaking, any lower rank approximation to an empirical data matrix Y should carry properties similar to the truncated SVD, i.e., should contain substantial stochastic information about the original random variable $\mathcal{X}$.

Coming back to the factor retrieval problem (2.4), we should note that while the product moment AA^T gives rise to the same (covariance) matrix R as Y does, the loading matrix A itself does not represent a sample data matrix of any random variable. Indeed, the number m of factors (or columns) in A could be far shorter than ℓ to represent meaningful samples. However, as far as approximating R by the product moment of some lower rank matrices is concerned, the idea of truncation can still be carried over. That is, we would like that a significant portion of those components in R corresponding to larger eigenvalues be captured by its low rank approximation $\hat{R}$ regardless of whether $\hat{R}$ is calculated via truncated random samples $\hat{Y}$ or reduced

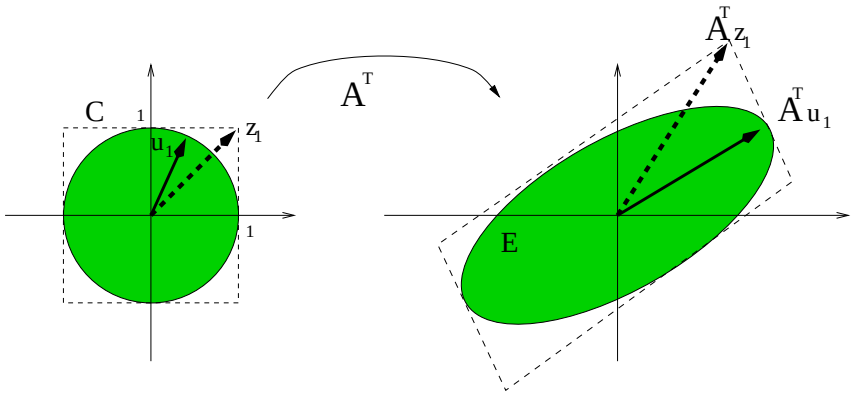

FIG. 6.1. *Comparison of geometric meanings of* $\mathbf{z}_1$ *and* $\mathbf{u}_1(R_1)$ *when* $n = 2$.

factors $\hat{A}$. The centroid method is an alternative way to accomplish that goal, as we shall now explain below.

For convenience, let $\lambda_1(M) \geq \lambda_2(M) \geq \cdots \geq \lambda_n(M)$ denote the eigenvalues of any given real-valued symmetric M. Let the corresponding unit eigenvectors be denoted as $\mathbf{u}_1(M), \ldots, \mathbf{u}_n(M)$. Recall the Rayleigh–Ritz theorem asserting that [8]

$$\lambda_1(M) = \max_{\|\mathbf{x}\|=1} \mathbf{x}^T M \mathbf{x}, \tag{6.6}$$

$$\lambda_k(M) = \max_{\substack{\|\mathbf{x}\|=1 \\ \mathbf{x}\perp \mathbf{u}_1(M),\ldots,\mathbf{u}_{k-1}(M)}} \mathbf{x}^T M \mathbf{x} \quad \text{for } k = 2, \ldots, n. \tag{6.7}$$

This variational characterization suggests a scheme for eigenvalue computation. Although the scheme is of little practical value in itself, its comparison with the centroid method is worth mentioning. First, observe that

$$\begin{aligned} \lambda_1(R_1) = (\mathbf{u}_1(R_1))^T R_1 \mathbf{u}_1(R_1) &= \max_{\|\mathbf{u}\|=1} \mathbf{u}^T R_1 \mathbf{u} = \max_{\|\mathbf{u}\|=1} \|A_1^T \mathbf{u}\|^2 \\ &\geq \mu_1 = \frac{1}{n} \mathbf{z}_1^T R_1 \mathbf{z}_1 = \frac{1}{n} \max_{|\mathbf{z}|=1} \mathbf{z}^T R_1 \mathbf{z} = \frac{1}{n} \max_{|\mathbf{z}|=1} \|A_1^T \mathbf{z}\|^2, \end{aligned} \tag{6.8}$$

where $\mathbf{z}_1$ is used to define the first modified centroid (see (4.2)). This relationship suggests that the sign vector $\mathbf{z}_1$ and the centroid value μ_1 are *mimicking* the left singular vector $\mathbf{u}_1$ and the square of the singular value λ_1 of A_1, respectively. Recall that the singular values of A_1^T are precisely the lengths of the semiaxes of the hyperellipsoid E defined by

$$E := \{A_1^T \mathbf{x} \in \mathbb{R}^m | \mathbf{x} \in \mathbb{R}^n, \|\mathbf{x}\| = 1\},$$

whence the first left singular vector $\mathbf{u}_1(R_1)$ of A_1 is mapped via A_1^T to the first major semiaxis of E. On the other hand, the unit cube

$$C := \{\mathbf{x} \in \mathbb{R}^n | \|\mathbf{x}\|_\infty = 1\}$$

is mapped under A_1^T to a hyperparallelepiped that circumscribes E. The geometric meanings of $\mathbf{z}_1$ and $\mathbf{u}_1(R_1)$ can be compared via Figure 6.1, where we draw C and E for the case $n = 2$.

Recall that once a factor represented by a unit vector $\mathbf{v}$ is determined, the components in the product $\mathbf{b} = A_1\mathbf{v}$ represent the loadings of all variables in that factor. The size of $\|\mathbf{b}\|$, called the significance earlier, can be used to indicate how essential that factor is to the variables. While the modified centroid is obtained by weighting loadings of all variables uniformly in the factor space, i.e., by constant weight $\frac{1}{n}$ except signs, the SVD amounts to weighting these loadings unevenly so as to maximize the significance of the resulting *biased centroid* (giving rise to the left singular vector). The latter is a fairly expensive and difficult task to accomplish, while the former is relatively easy to do via the centroid method. In this sense, we say that $\mathbf{z}_1$ gives a foretaste of the location of $\mathbf{u}_1$.

Recall in the centroid method that once the first centroid factor is found, the matrix R_1 is reduced to R_2 according to (4.7) and the search for the next centroid factor continues. In exactly the same way, once the first eigenvector $\mathbf{u}_1(R_1)$ is found, the matrix R_1 can be reduced, by the same Wedderburn rank reduction formula, to

$$\overline{R}_2 := R_1 - \lambda_1 \mathbf{u}_1(R_1)\,(\mathbf{u}_1(R_1))^T\,. \tag{6.9}$$

It can easily be proved from (6.7) that

$$(\mathbf{u}_2(R_1))^T\, R_1 \mathbf{u}_2(R_1) = \lambda_2(R_1) = \lambda_1(\overline{R}_2) = \left(\mathbf{u}_1(\overline{R}_2)\right)^T \overline{R}_2 \mathbf{u}_1(\overline{R}_2). \tag{6.10}$$

The relationship described above between $\mathbf{z}_1$ and $\mathbf{u}_1(R_1)$ in principle can be carried over to a similar relationship between $\mathbf{z}_2$ and $\mathbf{u}_2(R_1)$. The only problem is that

$$R_2 \neq \overline{R}_2 \tag{6.11}$$

because the two matrices are reduced from R_1 using $\mathbf{z}_1$ and $\mathbf{u}_1(R_1)$, respectively. However, we have pointed out earlier that at least in the initial stage $\mathbf{z}_1$ mimics the role of $\mathbf{u}_1(R_1)$ reasonably, so most of the stochastic information in R_2 should remain close to that in $\overline{R}_2$. As the iteration continues, of course, the closeness between R_i and $\overline{R}_i$ begins to depart. Consequently, the resemblance between $\mathbf{z}_i$ and $u_i(R_i)$ is expected to deteriorate progressively. Regardless, if we are interested in only the first few essential factors, i.e., in capturing the qualitative behavior of the (truncated) SVD of A_1, the centroid decomposition seems to be a reasonable and quick alternative. We summarize the comparison of the centroid decomposition and SVD in Table 6.1. We indicate only the first step in both decompositions. The successive steps are done similarly.

Furthermore, we plot in Figure 6.2 the centroid values and the singular values of the correlation matrix of a randomly generated 200×200 matrix A_1. Recall from Theorem 6.1 that the singular values indicate the degree of contribution to the randomness by the left singular vectors. Figure 6.2 is a typical representation of our many random tests. From the figure we see that the centroid values seem to mimic the behavior of singular values reasonably well and, hence, should provide a reasonable measurement of the original stochastic nature.

On the other hand, we point out that the reduced matrices R_i are no longer the same as $\overline{R}_i$ after the first step. We therefore plot in Figure 6.3 the logarithmic values of $|\cos(\theta_i)|$ where θ_i is the angle between $\mathbf{z}_i$ and $\mathbf{u}_i$ for $i = 1, \ldots, 200$. These values allow us to examine the degree of alignment of the sign vector $\mathbf{z}_i$ for the matrix R_i with the ith left singular vector of R_1. This diagram seems to suggest that the loss of alignment is not bad. In fact, toward the end of the calculation, it seems that the alignment is remarkably good. Further research is needed to understand this alignment issue.

TABLE 6.1
Comparison of centroid decomposition and SVD.

Centroid decomposition	SVD
$\mu_1 = \frac{1}{n}\max_{\|\mathbf{z}\|=1} \mathbf{z}^T R_1 \mathbf{z}$ (centroid value)	$\lambda_1 = \max_{\|\mathbf{x}\|=1} \mathbf{x}^T R_1 \mathbf{x}$ (eigenvalue)
$\mathbf{z}_1 = \arg\max_{\|\mathbf{z}\|=1} \mathbf{z}^T R_1 \mathbf{z}$ (sign vector for modified centroid)	$\mathbf{u}_1 = \arg\max_{\|\mathbf{x}\|=1} \mathbf{x}^T R_1 \mathbf{x}$ (left singular vector)
easy to obtain $\mathbf{z}_1$ in $O(n)$ steps (transverse hypercube)	not easy to obtain $\mathbf{u}_1$ via iterations (nonlinear iteration)
$\mathbf{v}_1 = \frac{A_1^T \mathbf{z}_1}{\sqrt{n\mu_1}}$ (centroid factor)	$\hat{\mathbf{v}}_1 = \frac{A_1^T \mathbf{u}_1}{\sqrt{\lambda_1}}$ (right singular vector)
$\gamma_1 = \|A_1 \mathbf{v}_1\|$ (significance)	$\sigma_1 = \sqrt{\lambda_1} = \|A_1 \hat{\mathbf{v}}_1\|$ (largest singular value)
$b_1 = A_1 \mathbf{v}_1$ (loading vector)	$\sigma_1 \mathbf{u}_1 = A_1 \hat{\mathbf{v}}_1$ (internal relation)
$A_1 = \sum b_i \mathbf{v}_i^T$ (centroid decomposition)	$A_1 = \sum \sigma_i \mathbf{u}_i \hat{\mathbf{v}}_i^T$ (SVD)
$R = \sum b_i b_i^T = \sum \gamma_i^2 \frac{b_i}{\|b_i\|}\left(\frac{b_i}{\|b_i\|}\right)^T$ (factor decomposition)	$R = \sum \lambda_i \mathbf{u}_i \mathbf{u}_i^T = \sum \sigma_i^2 \mathbf{u}_i \mathbf{u}_i^T$ (spectral decomposition)
$R_2 = R_1 - \frac{R_1 \mathbf{z}_1 \mathbf{z}_1^T R_1}{\mathbf{z}_1^T R_1 \mathbf{z}_1} = R_1 - \gamma_1^2 \frac{b_1}{\|b_1\|}\left(\frac{b_1}{\|b_1\|}\right)^T$ (rank reduction)	$\overline{R}_2 = R_1 - \frac{R_1 \mathbf{u}_1 \mathbf{u}_1^T R_1}{\mathbf{u}_1^T R_1 \mathbf{u}_1} = R_1 - \lambda_1 \mathbf{u}_1 \mathbf{u}_1^T$ (rank reduction)

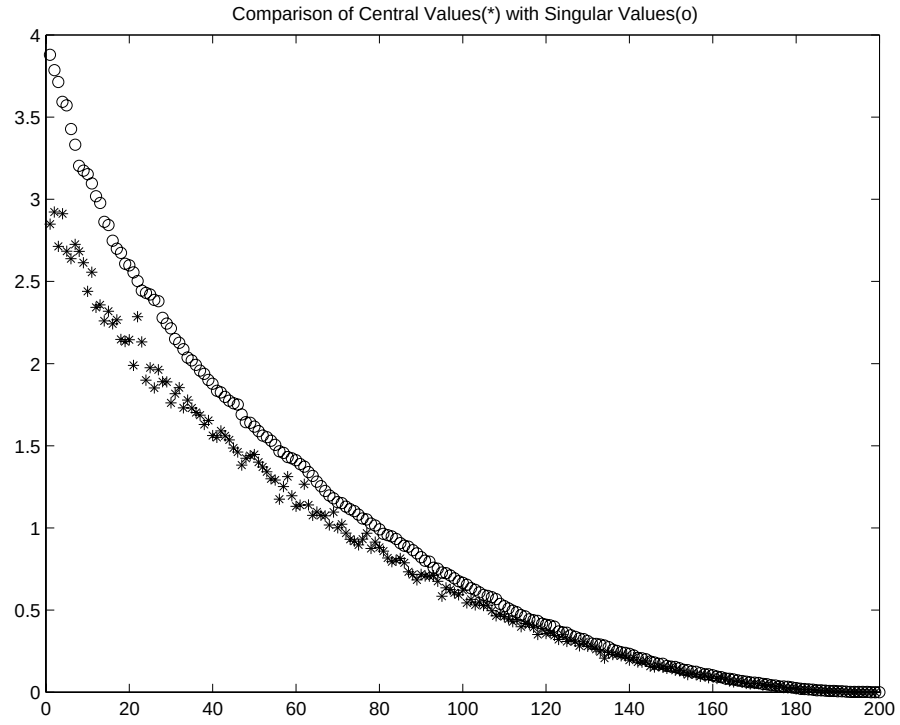

FIG. 6.2. *Comparison of centroid values and singular values for correlation matrix of* $n = 200$.

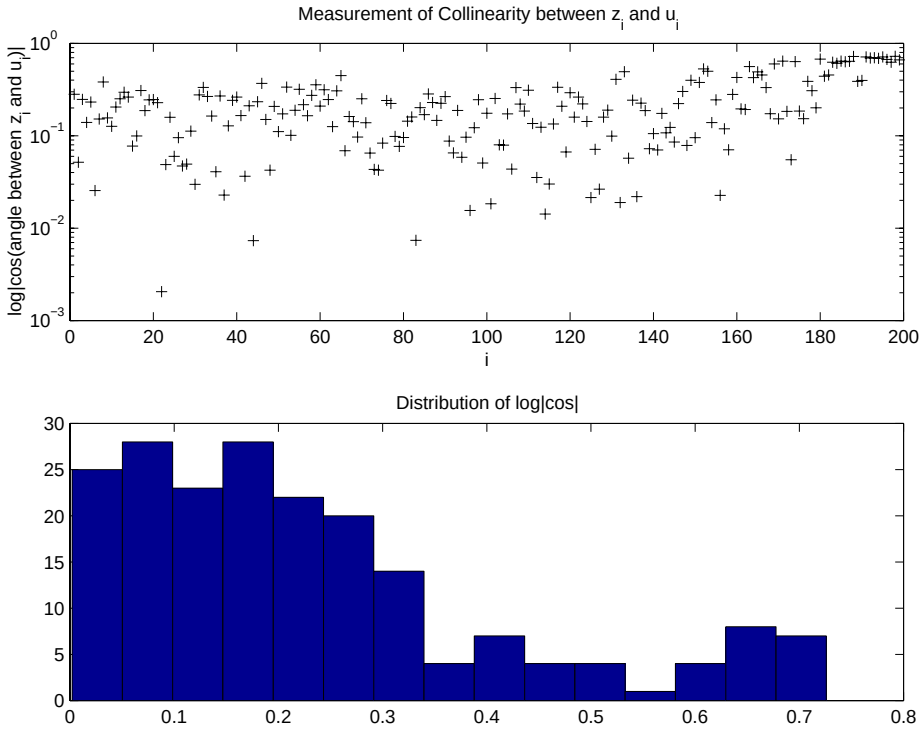

FIG. 6.3. *Degree of alignment between* $\mathbf{z}_i$ *and* u_i.

7. Relationship to data retrieval. The research and notions of factor analysis have been used in many disciplines, notably in educational, social, psychological, and behavioral measurements [5]. In this section, we connect and illustrate factor analysis development of the centroid method to that of information retrieval and data mining. This illustration leads to the unification of a class of approximations to the SVD.

We shall limit our goal in information retrieval to the task of finding documents relevant to given queries [12]. The idea in the so-called *latent semantic indexing* (LSI) is as follows: The textual documents are usually collected in an *indexing matrix* $H = [h_{kj}]$ in $\mathbb{R}^{m\times\ell}$. Each document is represented by one column in H. The entry h_{kj} in H represents the *weight* of one particular *term* k in document j whereas each term could be defined by just one single word or a string of phrases. A natural choice of the weight h_{kj} is obtained by counting the number of times that the term k occurs in document j. More elaborate weighting schemes can be found in the literature (see, for example, [12]) and are observed to yield better performance. Each query is represented as a row vector $\mathbf{q}_i^T = [q_{i1}, \ldots, q_{im}]$ in $\mathbb{R}^m$ where q_{ik} represents the weight of term k in the query i. Again, the weighting for terms in a query can also use more elaborate schemes. To measure how the query $\mathbf{q}_i^T$ matches the documents, we calculate the row vector

$$\mathbf{s}_i^T = \mathbf{q}_i^T H \tag{7.1}$$

and rank the relevance of documents to $\mathbf{q}$ according to the *scores* in $\mathbf{s}$. To put the

notation in the context of our discussion in the preceding sections, we observe the following analogies:

$$\begin{aligned}
\text{indexing matrix } H &\longleftrightarrow \text{scoring matrix } F,\\
\text{document } j &\longleftrightarrow \text{entity } j,\\
\text{term } k &\longleftrightarrow \text{factor } k,\\
\text{weight } h_{kj} &\longleftrightarrow \text{score of factor } k \text{ on entity } j,\\
\text{one query } \mathbf{q}_i^T &\longleftrightarrow \text{one row in loading matrix } A,\\
\text{weights } q_{ik} &\longleftrightarrow \text{loadings of query } i \text{ on factor } k,\\
\text{scores in } \mathbf{s}_i^T &\longleftrightarrow \text{scores in } i \text{ row of data matrix } Y.
\end{aligned}$$

Nevertheless, in contrast to the factor retrieval described above, the calculations involved in an LSI application place emphasis not so much on computing the factors based on the scores in $\mathbf{s}_i^T$, $i = 1, \ldots, n$, but rather on the vector-matrix multiplication (7.1). Indeed, in a search engine application, quite often there is only one query, i.e., $n = 1$, per the user's request. The factors are already specified by predetermined terms. The focus in LSI has been on representing the indexing matrix and the queries in a more compact form so as to facilitate the computation of the scores. Toward that end, one way to do LSI is to use the truncated SVD of H.

From our discussion in section 6, we now understand well why using the truncated SVD to represent H makes sense, provided data in H have been centered. It is not just the best approximation to H in norm, but more importantly it also contains a substantial portion of stochastic nature of the original H. On the other hand, since the original indexing matrix H is never exact, truncation also has the benefit of cutting away noise when the signal-to-noise ratio (SNR) is too small. Suppose that

$$H = \sum_{i=1}^{\gamma} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{7.2}$$

denotes the SVD of H of rank γ. A general practice in LSI is to replace H by

$$\hat{H}_k := \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{7.3}$$

with $k \ll \gamma$ and compute $\mathbf{s} \approx \mathbf{q}^T \hat{H}_k$. The problem is that the low rank $\hat{H}_k$ could require more storage than the original H that often is sparse. One of the suggestions for saving storage has been to approximate H by the SDD [12] that also resembles the SVD, i.e.,

$$\tilde{H}_k := \sum_{i=1}^{k} \delta_i \mathbf{x}_i \mathbf{y}_i^T, \tag{7.4}$$

where each $\mathbf{x}_i$ and $\mathbf{y}_i$ is constrained to have integer entries $-1, 0$, or 1, and the d_i are positive real numbers.

One purpose of this paper is to suggest using the truncated centroid decomposition as an alternative low rank approximation to an indexing matrix H, even though the objective of LSI is not to retrieve factors from observed data of scores. There is considerable similarity between the centroid decomposition and the SDD, but there is also a significant difference, as we shall address later in section 9.

8. General centroid algorithm. In the context of factor retrieval, we would like to have as few factors as possible. It is often the case that $m \ll \ell$ and $n \ll \ell$. The centroid method is applied directly to the product moment $R = AA^T$. In the context of data retrieval, what is given is the index matrix H. If $m \ll \ell$, we could simply apply Algorithm 5.1 to the product moment HH^T. This would correspond to a subject area where the number of terms is relatively limited, while the number of documents is large. However, in the context of LSI, it is common that each document contains many more terms (or keywords) and that the contents of the documents should sparsely overlap each other. It is reasonable to assume that $m \gg \ell$. In this case, it probably is not economical to form the product moment HH^T, as is the practice with factor analysis (2.1). We modify the centroid method to work directly with H.

ALGORITHM 8.1 (general centroid algorithm). *Assume that initial values* $|\mathbf{z}^{(0)}| = \mathbf{1}, \mathbf{w}^{(0)} := H^T\mathbf{z}^{(0)}$ *as well the vector* $\mathbf{d} = \operatorname{diag}(HH^T)$ *are available. Repeat the following steps for* $i = 0, 1, \ldots$:

1. *Compute* $\mathbf{g}^{(i)} := \mathbf{d} - sgn(\mathbf{z^{(i)}}) \circ (H\mathbf{w}^{(i)})$.
2. *Choose* k *so that* $\mathbf{g}_k^{(i)}$ *is maximal and greater than* ϵ*; otherwise, stop.*
3. *Define* $\mathbf{z}^{(i+1)}$ *by simply changing the sign of* $\mathbf{z}_k^{(i)}$.
4. *Update* $\mathbf{w}^{(i+1)} := \mathbf{w}^{(i)} + 2sgn(\mathbf{z}_k^{(i+1)})H(k,:)^T$.

As with Algorithm 5.1 this is an ascent method (on an m-dimensional hypercube) because

$$\begin{aligned}\|H^T\mathbf{z}^{(i+1)}\|^2 &= \left(\mathbf{z}^{(i)} - 2sgn(\mathbf{z}_k^{(i)})\mathbf{e}_k\right)^T HH^T\left(\mathbf{z}^{(i)} - 2sgn(\mathbf{z}_k^{(i)})\mathbf{e}_k\right)\\ &= \|H^T\mathbf{z}^{(i)}\|^2 + 4\mathbf{e}_k^T HH^T\mathbf{e}_k - 4sgn(\mathbf{z}_k^{(i)})(\mathbf{e}_k^T H\mathbf{w}^{(i)})\\ &= \|H^T\mathbf{z}^{(i)})\|^2 + 4\mathbf{g}_k^{(i)}\end{aligned}$$

and $\mathbf{g}_k^{(i)} > \epsilon$. Once the optimal $\mathbf{z_1}$ is found, the matrix $\mathbf{H_1} = \mathbf{H}$ is reduced to

$$H_2 := H_1 - H_1\mathbf{v}_1\mathbf{v}_1^T, \tag{8.1}$$

where $\mathbf{v}_1$ is the normalized unit vector of $H_1^T\mathbf{z}_1$, and then the algorithm is applied to H_2 and so on. A common practice in factor analysis is to terminate the algorithm whenever the resulting significance $\|\mathbf{b}_k\|$ drops below a specific threshold level. If the algorithm is carried to the end, we obtain the centroid decomposition

$$H = H_1 = \mathbf{b}_\gamma\mathbf{v}_\gamma^T + \cdots + \mathbf{b}_1\mathbf{v}_1^T, \tag{8.2}$$

where $\mathbf{b}_k := H_k\mathbf{v}_k$, $k = 1, \ldots, \gamma$, is the *loading* whose significance is analogous to the singular value of H_k. (See the comparison in Table 6.1.) Furthermore, from the discussion in section 6, we see that the first few loadings carry most of the stochastic information in H. That is, a truncated centroid decomposition may often be as effective as the truncated SVD and can be expected to be much cheaper computationally, as we will now see.

Since a single centroid iteration on a correlation has an order n sorting (step 1 in Algorithm 5.1) and an n-dimensional vector addition (step 3 in Algorithm 5.1), the complexity is $\mathcal{O}(kn^2)$ for a rank-k approximation. When the centroid algorithm is executed on H rather than HH^T, then the number of expected iteration steps is $\frac{m}{2}$ for each centroid value. Note that the first step in Algorithm 8.1 involves an m-dimensional vector subtraction and an $\mathcal{O}(m\ell)$ matrix to vector multiply, the next

step an m-dimensional sorting, and an ℓ-dimensional vector addition in the last step. A rank-k centroid decomposition approximation to the rank-k truncated SVD of the scoring matrix H would involve $\mathcal{O}(km^2\ell)$ complexity. Obviously, the complexity may be further reduced if sparsity can be exploited.

9. Conclusions. We have recast the centroid method as an $\mathcal{O}(n)$-step optimization problem on a hypercube. This interpretation enables us to view the centroid method as a matrix approximation with many similarities to the truncated SVD.

Furthermore, we offer the insight that given any data matrix (with mean zero) whose columns represent random samples from a certain unknown distribution, its singular values then provide a measurement of the second order statistical information of the original data in the direction of the corresponding left singular vectors. This insight explains why, how, and when a low rank approximation can be used as a reasonable approximation to the original matrix. Although low rank approximation has been a common practice used in many important applications, we have not seen a satisfactory stochastic justification. There seems to be much misuse and misunderstanding of low rank approximation techniques.

We justify the truncated SVD as not only the nearest distancewise approximation, but also as the minimum-variance approximation to the original data. It seems fitting that any low rank approximation should carry stochastic properties similar to the truncated SVD. We have shown this to be true of the centroid decomposition empirically. Furthermore, we have shown that the centroid method can be generalized so that it might be used for many applications, e.g., the LSI problem.

Figure 9.1 can be viewed as a fundamental triad which includes the three equivalent variational formulations for the largest singular value of a matrix A.

The SDD method is analogous to the top vertex of the triad using only vectors $\mathbf{u}$ and $\mathbf{v}$, whose components are restricted to the set $\{0, 1, -1\}$. The centroid method

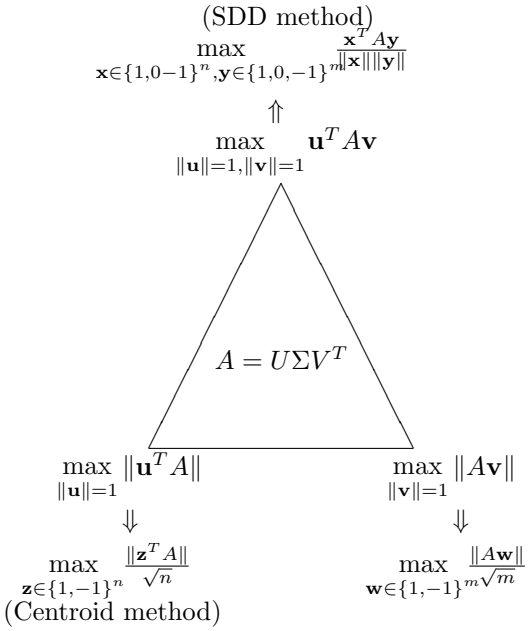

FIG. 9.1. *Fundamental SVD triad.*

is analogous to the left vertex of the triad, with the restriction that the vector $\mathbf{u}$ is allowed to have components only from the set $\{0, 1, -1\}$; note, however, that the inclusion of 0 does not give any additional discrete approximation because the corresponding objective function is convex and the maximum must occur at a sign vector (vertex). We call these sets *restricted*, whereas the vectors that determine the singular values are completely unrestricted. The form in the lower right corner stands ready to be analyzed in future work.

Finally, the important unification idea is that we can consider the centroid and semidiscrete methods as producing approximations to the truncated SVD using just two of the classes of many restricted discrete sets. We summarize relationships of centroid decomposition, SDD, and SVD in Table 9.1.

TABLE 9.1
Comparison of centroid decomposition, SVD, and SDD.

Decomposition		
Centroid	Singular value	Semidiscrete
$\mu = \frac{1}{n}\max_{\lvert\mathbf{z}\rvert=\mathbf{1}} \lVert\mathbf{z}^T A\rVert^2$ (centroid value)		
	$\sigma = \max_{\lVert\mathbf{u}\rVert=1} \lVert\mathbf{u}^T A\rVert$ (singular value)	
$\mathbf{v} = \frac{A^T\mathbf{z}}{\lVert A^T\mathbf{z}\rVert}$ (centroid factor)	$\mathbf{v} = \frac{A^T\mathbf{u}}{\lVert A^T\mathbf{u}\rVert}$ (right singular vector)	
	$\sigma = \max_{\lVert\mathbf{u}\rVert=\lVert\mathbf{v}\rVert=1} \lvert\mathbf{u}^T A\mathbf{v}\rvert$	$\delta = \max_{\lvert\mathbf{x}\rvert=\lvert\mathbf{y}\rvert\in\{1,0\}} \frac{\lvert\mathbf{x}^T A\mathbf{y}\rvert}{\lVert\mathbf{x}\rVert\lVert\mathbf{y}\rVert}$
	$\sigma = \max_{\lVert\mathbf{v}\rVert=1} \lVert A\mathbf{v}\rVert$	
$\mathbf{b} = A\mathbf{v}$ (loading vector)	$\sigma\mathbf{u} = A\mathbf{v}$ (internal relation)	
$\gamma = \lVert\mathbf{b}\rVert$ (significance)		
$A \approx (A\mathbf{v})\mathbf{v}^T$	$A \approx (A\mathbf{v})\mathbf{v}^T$	$A \approx (\delta\mathbf{x})\mathbf{y}^T$
$A_1 = \sum \mathbf{b}_i\mathbf{v}_i^T$ (CD)	$A_1 = \sum \sigma_i\mathbf{u}_i\mathbf{v}_i^T$ (SVD)	$A_1 = \sum \delta_i\mathbf{x}_i\mathbf{y}_i^T$ (SDD)
$\operatorname{rank}(A - \gamma\frac{\mathbf{b}}{\lVert\mathbf{b}\rVert}\mathbf{v}^T) =$ $\operatorname{rank}(A) - 1$	$\operatorname{rank}(A - \sigma\mathbf{u}\mathbf{v}^T) =$ $\operatorname{rank}(A) - 1$	(no rank subtraction)

Acknowledgment. The authors wish to give special thanks to the referees for important insights, suggestions, criticisms, and corrections.

REFERENCES

[1] M. ANDERBERG, *Cluster Analysis for Applications*, Academic Press, New York, 1973.
[2] C. BURT, *The Distribution and Relations of Educational Abilities*, P. S. King and Son, London, 1917.
[3] M. T. CHU, *On the Statistical Meaning of the Truncated Singular Decomposition*, manuscript.

[4] M. T. Chu, R. E. Funderlic, and G. H. Golub, *A rank-one reduction formula and its applications to matrix factorizations*, SIAM Rev., 37 (1995), pp. 512–530.
[5] A. L. Comrey and H. B. Lee, *A First Course in Factor Analysis*, Erlbaum Associates, Hillsdale, NJ, 1992.
[6] C. Eckert and G. Young, *The approximation of one matrix by another of lower rank*, Psych., 1 (1936), pp. 211–218.
[7] H. H. Harman, *Modern Factor Analysis*, University of Chicago Press, Chicago, 1967.
[8] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, UK, 1991.
[9] P. Horst, *Factor Analysis of Data Matrices*, Holt, Rinehart and Winston, New York, 1965.
[10] H. Hotelling, *Analysis of a complex of statistical variables into principal components*, J. Ed. Psych., 24 (1933), pp. 417–441, 498–502.
[11] L. Hubert, J. Meulman, and W. Heiser, *Two purposes for matrix factorization: A historical appraisal*, SIAM Rev., 42 (2000), pp. 68–82.
[12] T. G. Kolda and D. P. O'Leary, *A semi-discrete matrix decomposition for latent semantic indexing in information retrieval*, ACM Trans. Inform. Systems, 16 (1998), pp. 322–346.
[13] D. G. Luenberger, *Optimization by Vector Space Methods*, John Wiley & Sons, New York, 1968.
[14] J. L. Melsa and D. L. Cohn, *Decision and Estimation Theory*, McGraw-Hill, New York, 1978.
[15] K. Pearson, *On lines and planes of closest fit to systems in space*, Phil. Mag., 6 (1901), pp. 559–572.
[16] G. W. Stewart, *On the early history of singular value decomposition*, SIAM Rev., 35 (1993), pp. 551–566.
[17] L. L. Thurston, *Multiple factor analysis*, Psych. Rev., 38 (1931), pp. 406–427.