Contents

1 Singular Value Decomposition (SVD)

The singular value decomposition of a matrix A is the factorization of A into the product of three matrices $A = UDV^T$ where the columns of U and V are orthonormal and the matrix D is diagonal with positive real entries. The SVD is useful in many tasks. Here we mention some examples. First, in many applications, the data matrix A is close to a matrix of low rank and it is useful to find a low rank matrix which is a good approximation to the data matrix . We will show that from the singular value decomposition of A, we can get the matrix B of rank k which best approximates A; in fact we can do this for every k. Also, singular value decomposition is defined for all matrices (rectangular or square) unlike the more commonly used spectral decomposition in Linear Algebra. The reader familiar with eigenvectors and eigenvalues (we do not assume familiarity here) will also realize that we need conditions on the matrix to ensure orthogonality of eigenvectors. In contrast, the columns of V in the singular value decomposition, called the right singular vectors of A, always form an orthogonal set with no assumptions on A. The columns of U are called the left singular vectors and they also form an orthogonal set. A simple consequence of the orthogonality is that for a square and invertible matrix A, the inverse of A is $VD^{-1}U^T$, as the reader can verify.

To gain insight into the SVD, treat the rows of an $n \times d$ matrix A as n points in a d-dimensional space and consider the problem of finding the best k-dimensional subspace with respect to the set of points. Here best means minimize the sum of the squares of the perpendicular distances of the points to the subspace. We begin with a special case of the problem where the subspace is 1-dimensional, a line through the origin. We will see later that the best-fitting k-dimensional subspace can be found by k applications of the best fitting line algorithm. Finding the best fitting line through the origin with respect to a set of points $\{\mathbf{x_i} | 1 \leq i \leq n\}$ in the plane means minimizing the sum of the squared distances of the points to the line. Here distance is measured perpendicular to the line. The problem is called the *best least squares fit.*

In the best least squares fit, one is minimizing the distance to a subspace. An alternative problem is to find the function that best fits some data. Here one variable y is a function of the variables $x_1, x_2, \ldots, x_d$ and one wishes to minimize the vertical distance, i.e., distance in the y direction, to the subspace of the x_i rather than minimize the perpendicular distance to the subspace being fit to the data.

Returning to the best least squares fit problem, consider projecting a point $\mathbf{x_i}$ onto a line through the origin. Then

$$x_{i1}^2 + x_{i2}^2 + \cdots + _{id}^2 = (\text{length of projection})^2 + (\text{distance of point to line})^2 .$$

See Figure 1.1. Thus

$$(\text{distance of point to line})^2 = x_{i1}^2 + x_{i2}^2 + \cdots + _{id}^2 - (\text{length of projection})^2 .$$

To minimize the sum of the squares of the distances to the line, one could minimize

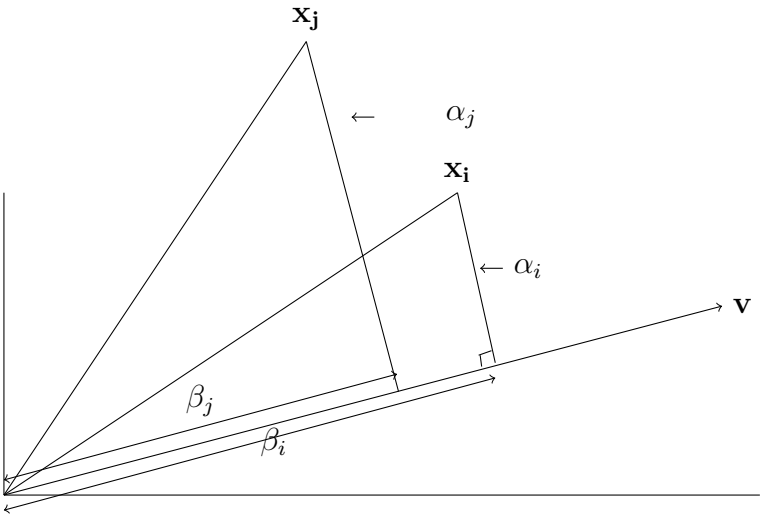


Figure 1.1: The projection of the point $\mathbf{x_i}$ onto the line through the origin in the direction of $\mathbf{v}$

$\sum_{i=1}^{n} (x_{i1}^2 + x_{i2}^2 + \cdots +_{id}^2)$ minus the sum of the squares of the lengths of the projections of the points to the line. However, $\sum_{i=1}^{n} (x_{i1}^2 + x_{i2}^2 + \cdots +_{id}^2)$ is a constant (independent of the line), so minimizing the sum of the squares of the distances is equivalent to maximizing the sum of the squares of the lengths of the projections onto the line. Similarly for best-fit subspaces, we could maximize the sum of the squared lengths of the projections onto the subspace instead of minimizing the sum of squared distances to the subspace.

The reader may wonder why we minimize the sum of squared perpendicular distances to the line. We could alternatively have defined the best-fit line to be the one which minimizes the sum of perpendicular distances to the line. There are examples where this definition gives a different answer than the line minimizing the sum of perpendicular distances squared. [The reader could construct such examples.] The choice of the objective function as the sum of squared distances seems arbitrary and in a way it is. But the square has many nice mathematical properties - the first of these is the use of Pythagoras theorem above to say that this is equivalent to maximizing the sum of squared projections. We will see that in fact we can use the Greedy Algorithm to find best-fit k dimensional subspaces (which we will define soon) and for this too, the square is important. The reader should also recall from Calculus that the best-fit function is also defined in terms of least-*squares* fit. There too, the existence of nice mathematical properties is the motivation for the square.

1.1 Singular Vectors

We now define the *singular vectors* of an $n \times d$ matrix A. Consider the rows of A as n points in a d-dimensional space. Consider the best fit line through the origin. Let $\mathbf{v}$ be a

unit vector along this line. The length of the projection of $\mathbf{a_i}$, the i^{th} row of A, onto $\mathbf{v}$ is $|\mathbf{a_i} \cdot \mathbf{v}|$. From this we see that the sum of length squared of the projections is $|A\mathbf{v}|^2$. The best fit line is the one maximizing $|A\mathbf{v}|^2$ and hence minimizing the sum of the squared distances of the points to the line.

With this in mind, define the *first singular vector*, $\mathbf{v_1}$, of A, which is a column vector, as the best fit line through the origin for the n points in d-space that are the rows of A. Thus

$$\mathbf{v_1} = \arg\max_{|\mathbf{v}|=1} |A\mathbf{v}|.$$

The value $\sigma_1(A) = |A\mathbf{v_1}|$ is called the *first singular value* of A. Note that σ_1^2 is the sum of the squares of the projections of the points to the line determined by $\mathbf{v_1}$.

The greedy approach to find the best fit 2-dimensional subspace for a matrix A, takes $\mathbf{v_1}$ as the first basis vector for the 2-dimenional subspace and finds the best 2-dimensional subspace containing $\mathbf{v_1}$. The fact that we are using the sum of squared distances will again help. For every 2-dimensional subspace containing $\mathbf{v_1}$, the sum of squared lengths of the projections onto the subspace equals the sum of squared projections onto $\mathbf{v_1}$ plus the sum of squared projections along a vector perpendicular to $\mathbf{v_1}$ in the subspace. Thus, instead of looking for the best 2-dimensional subspace containing $\mathbf{v_1}$, look for a unit vector, call it $\mathbf{v_2}$, perpendicular to $\mathbf{v_1}$ that maximizes $|A\mathbf{v}|^2$ among all such unit vectors. Using the same greedy strategy to find the best three and higher dimensional subspaces, defines $\mathbf{v_3}, \mathbf{v_4}, \ldots$ in a similar manner. This is captured in the following definitions. There is no apriori guarantee that the greedy algorithm gives the best fit. But, in fact, the greedy algorithm does work and yields the best-fit subspaces of every dimension as we will show.

The *second singular vector*, $\mathbf{v_2}$, is defined by the best fit line perpendicular to $\mathbf{v_1}$

$$\mathbf{v_2} = \arg\max_{\mathbf{v}\perp\mathbf{v_1},|\mathbf{v}|=1} |A\mathbf{v}|.$$

The value $\sigma_2(A) = |A\mathbf{v_2}|$ is called the *second singular value* of A**.** The *third singular vector* $\mathbf{v}_3$ is defined similarly by

$$\mathbf{v_3} = \arg\max_{\mathbf{v}\perp\mathbf{v_1},\mathbf{v_2},|\mathbf{v}|=1} |A\mathbf{v}|$$

and so on. The process stops when we have found

$$\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$$

as singular vectors and

$$\arg\max_{\substack{\mathbf{v}\perp\mathbf{v_1},\mathbf{v_2},\ldots,\mathbf{v_r}\\|\mathbf{v}|=1}} |A\mathbf{v}| = 0.$$

If instead of finding $\mathbf{v_1}$ that maximized $|A\mathbf{v}|$ and then the best fit 2-dimensional subspace containing $\mathbf{v_1}$, we had found the best fit 2-dimensional subspace, we might have

done better. This is not the case. We now give a simple proof that the greedy algorithm indeed finds the best subspaces of every dimension.

Theorem 1.1 *Let A be an $n \times d$ matrix where $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$ are the singular vectors defined above. For $1 \leq k \leq r$, let V_k be the subspace spanned by $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}$. Then for each k, V_k is the best-fit k-dimensional subspace for A.*

Proof: The statement is obviously true for $k = 1$. For $k = 2$, let W be a best-fit 2-dimensional subspace for A. For any basis $\mathbf{w_1}, \mathbf{w_2}$ of W, $|A\mathbf{w_1}|^2 + |A\mathbf{w_2}|^2$ is the sum of squared lengths of the projections of the rows of A onto W. Now, choose a basis $\mathbf{w}_1, \mathbf{w_2}$ of W so that $\mathbf{w_2}$ is perpendicular to $\mathbf{v_1}$. If $\mathbf{v_1}$ is perpendicular to W, any unit vector in W will do as $\mathbf{w_2}$. If not, choose $\mathbf{w_2}$ to be the unit vector in W perpendicular to the projection of $\mathbf{v_1}$ onto W. Since $\mathbf{v_1}$ was chosen to maximize $|A\mathbf{v_1}|^2$, it follows that $|A\mathbf{w_1}|^2 \leq |A\mathbf{v_1}|^2$. Since $\mathbf{v_2}$ was chosen to maximize $|A\mathbf{v_2}|^2$ over all $\mathbf{v}$ perpendicular to $\mathbf{v_1}$, $|A\mathbf{w_2}|^2 \leq |A\mathbf{v_2}|^2$. Thus

$$|A\mathbf{w_1}|^2 + |A\mathbf{w_2}|^2 \leq |A\mathbf{v_1}|^2 + |A\mathbf{v_2}|^2.$$

Hence, V_2 is at least as good as W and so is a best-fit 2-dimensional subspace.

For general k, proceed by induction. By the induction hypothesis, V_{k-1} is a best-fit k-1 dimensional subspace. Suppose W is a best-fit k-dimensional subspace. Choose a basis $\mathbf{w_1}, \mathbf{w_2}, \ldots, \mathbf{w_k}$ of W so that $\mathbf{w_k}$ is perpendicular to $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_{k-1}}$. Then

$$|A\mathbf{w_1}|^2 + |A\mathbf{w_2}|^2 + \cdots + |A\mathbf{w_k}|^2 \leq |A\mathbf{v_1}|^2 + |A\mathbf{v_2}|^2 + \cdots + |A\mathbf{v_{k-1}}|^2 + |A\mathbf{w_k}|^2$$

since V_{k-1} is an optimal k -1 dimensional subspace. Since $\mathbf{w_k}$ is perpendicular to $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_{k-1}}$, by the definition of $\mathbf{v_k}$, $|A\mathbf{w_k}|^2 \leq |A\mathbf{v_k}|^2$. Thus

$$|A\mathbf{w_1}|^2 + |A\mathbf{w_2}|^2 + \cdots + |A\mathbf{w_{k-1}}|^2 + |A\mathbf{w_k}|^2 \leq |A\mathbf{v_1}|^2 + |A\mathbf{v_2}|^2 + \cdots + |A\mathbf{v_{k-1}}|^2 + |A\mathbf{v_k}|^2,$$

proving that V_k is at least as good as W and hence is optimal. ∎

Note that the n-vector $A\mathbf{v_i}$ is really a list of lengths (with signs) of the projections of the rows of A onto $\mathbf{v_i}$. Think of $|A\mathbf{v_i}| = \sigma_i(A)$ as the "component" of the matrix A along $\mathbf{v_i}$. For this interpretation to make sense, it should be true that adding up the squares of the components of A along each of the $\mathbf{v_i}$ gives the square of the "whole content of the matrix A". This is indeed the case and is the matrix analogy of decomposing a vector into its components along orthogonal directions.

Consider one row, say $\mathbf{a_j}$, of A. Since $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$ span the space of all rows of A, $\mathbf{a_j} \cdot \mathbf{v} = 0$ for all $\mathbf{v}$ perpendicular to $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$. Thus, for each row $\mathbf{a_j}$, $\sum_{i=1}^{r} (\mathbf{a_j} \cdot \mathbf{v_i})^2 = |\mathbf{a_j}|^2$. Summing over all rows j,

$$\sum_{j=1}^{n} |\mathbf{a_j}|^2 = \sum_{j=1}^{n} \sum_{i=1}^{r} (\mathbf{a_j} \cdot \mathbf{v_i})^2 = \sum_{i=1}^{r} \sum_{j=1}^{n} (\mathbf{a_j} \cdot \mathbf{v_i})^2 = \sum_{i=1}^{r} |A\mathbf{v_i}|^2 = \sum_{i=1}^{r} \sigma_i^2(A).$$

But $\sum_{j=1}^{n} |\mathbf{a_j}|^2 = \sum_{j=1}^{n} \sum_{k=1}^{d} a_{jk}^2$, the sum of squares of all the entries of A. Thus, the sum of squares of the singular values of A is indeed the square of the "whole content of A", i.e., the sum of squares of all the entries. There is an important norm associated with this quantity, the Frobenius norm of A, denoted $||A||_F$ defined as

$$||A||_F = \sqrt{\sum_{j,k} a_{jk}^2}.$$

Lemma 1.2 *For any matrix A, the sum of squares of the singular values equals the Frobenius norm. That is, $\sum \sigma_i^2(A) = ||A||_F^2$.*

Proof: By the preceding discussion. ∎

A matrix A can be described fully by how it transforms the vectors $\mathbf{v_i}$. Every vector $\mathbf{v}$ can be written as a linear combination of $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$ and a vector perpendicular to all the $\mathbf{v_i}$. Now, $A\mathbf{v}$ is the same linear combination of $A\mathbf{v_1}, A\mathbf{v_2}, \ldots, A\mathbf{v_r}$ as $\mathbf{v}$ is of $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$. So the $A\mathbf{v_1}, A\mathbf{v_2}, \ldots, A\mathbf{v_r}$ form a fundamental set of vectors associated with A. We normalize them to length one by

$$\mathbf{u_i} = \frac{1}{\sigma_i(A)} A\mathbf{v_i}.$$

The vectors $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_r}$ are called the *left singular vectors* of A. The $\mathbf{v_i}$ are called the *right singular vectors*. The SVD theorem (Theorem 1.5) will fully explain the reason for these terms.

Clearly, the right singular vectors are orthogonal by definition. We now show that the left singular vectors are also orthogonal and that $A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T$.

Theorem 1.3 *Let A be a rank r matrix. The left singular vectors of A, $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_r}$, are orthogonal.*

Proof: The proof is by induction on r. For $r = 1$, there is only one $\mathbf{u_i}$ so the theorem is trivially true. For the inductive part consider the matrix

$$B = A - \sigma_1 \mathbf{u_1} \mathbf{v_1}^T.$$

The implied algorithm in the definition of singular value decomposition applied to B is identical to a run of the algorithm on A for its second and later singular vectors and singular values. To see this, first observe that $B\mathbf{v_1} = A\mathbf{v_1} - \sigma_1 \mathbf{u_1} \mathbf{v_1^T} \mathbf{v_1} = 0$. It then follows that the first right singular vector, call it $\mathbf{z}$, of B will be perpendicular to $\mathbf{v_1}$ since if it had a component $\mathbf{z_1}$ along $\mathbf{v_1}$, then, $\left| B \frac{\mathbf{z}-\mathbf{z_1}}{|\mathbf{z}-\mathbf{z_1}|} \right| = \frac{|B\mathbf{z}|}{|\mathbf{z}-\mathbf{z_1}|} > |B\mathbf{z}|$, contradicting the arg max definition of z. But for any $\mathbf{v}$ perpendicular to $\mathbf{v_1}$, $B\mathbf{v} = A\mathbf{v}$. Thus, the top singular

vector of B is indeed a second singular vector of A. Repeating this argument shows that a run of the algorithm on B is the same as a run on A for its second and later singular vectors. This is left as an exercise.

Thus, there is a run of the algorithm that finds that B has right singular vectors $\mathbf{v_2}, \mathbf{v_3}, \ldots, \mathbf{v_r}$ and corresponding left singular vectors $\mathbf{u_2}, \mathbf{u_3}, \ldots, \mathbf{u_r}$. By the induction hypothesis, $\mathbf{u_2}, \mathbf{u_3}, \ldots, \mathbf{u_r}$ are orthogonal.

It remains to prove that $\mathbf{u_1}$ is orthogonal to the other $\mathbf{u_i}$. Suppose not and for some $i \geq 2$, $\mathbf{u_1}^T\mathbf{u_i} \neq 0$. Without loss of generality assume that $\mathbf{u_1}^T\mathbf{u_i} > 0$. The proof is symmetric for the case where $\mathbf{u_1}^T\mathbf{u_i} < 0$. Now, for infinitesimally small $\varepsilon > 0$, the vector

$$A\left(\frac{\mathbf{v}_1 + \varepsilon\mathbf{v_i}}{|\mathbf{v_1} + \varepsilon\mathbf{v_i}|}\right) = \frac{\sigma_1\mathbf{u_1} + \varepsilon\sigma_i\mathbf{u_i}}{\sqrt{1+\varepsilon^2}}$$

has length at least as large as its component along $\mathbf{u_1}$ which is

$$\mathbf{u_1^T}\left(\frac{\sigma_1\mathbf{u_1} + \varepsilon\sigma_i\mathbf{u_i}}{\sqrt{1+\varepsilon^2}}\right) = \left(\sigma_1 + \varepsilon\sigma_i\mathbf{u_1}^T\mathbf{u_i}\right)\left(1 - \tfrac{\varepsilon^2}{2} + O\left(\varepsilon^4\right)\right) = \sigma_1 + \varepsilon\sigma_i\mathbf{u_1}^T\mathbf{u_i} - O\left(\varepsilon^2\right) > \sigma_1$$

a contradiction. Thus, $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_r}$ are orthogonal. ∎

1.2 Singular Value Decomposition (SVD)

Let A be an $n \times d$ matrix with singular vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$ and corresponding singular values $\sigma_1, \sigma_2, \ldots, \sigma_r$. Then $\mathbf{u_i} = \frac{1}{\sigma_i}A\mathbf{v_i}$, for $i = 1, 2, \ldots, r$, are the left singular vectors and by Theorem 1.5, A can be decomposed into a sum of rank one matrices as

$$A = \sum_{i=1}^{r} \sigma_i\mathbf{u_i}\mathbf{v_i}^T.$$

We first prove a simple lemma stating that two matrices A and B are identical if $A\mathbf{v} = B\mathbf{v}$ for all $\mathbf{v}$. The lemma states that in the abstract, a matrix A can be viewed as a transformation that maps vector $\mathbf{v}$ onto $A\mathbf{v}$.

Lemma 1.4 *Matrices A and B are identical if and only if for all vectors $\mathbf{v}$, $A\mathbf{v} = B\mathbf{v}$.*

Proof: Clearly, if $A = B$ then $A\mathbf{v} = B\mathbf{v}$ for all $\mathbf{v}$. For the converse, suppose that $A\mathbf{v} = B\mathbf{v}$ for all $\mathbf{v}$. Let $\mathbf{e_i}$ be the vector that is all zeros except for the i^{th}component which has value 1. Now $A\mathbf{e_i}$ is the i^{th} column of A and thus $A = B$ if for each i, $A\mathbf{e_i} = B\mathbf{e_i}$.

∎

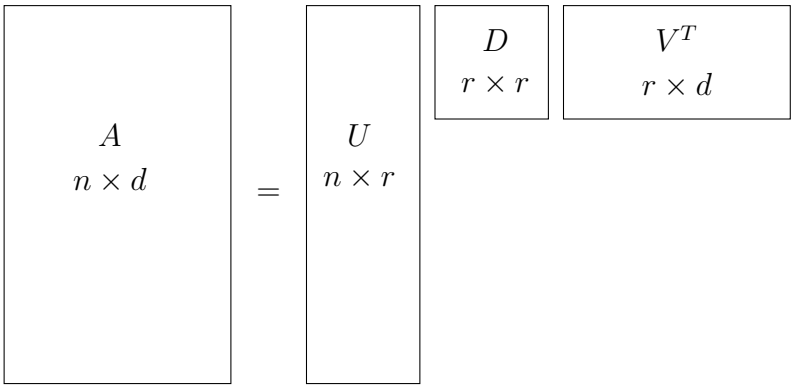


Figure 1.2: The SVD decomposition of an $n \times d$ matrix.

Theorem 1.5 *Let A be an $n \times d$ matrix with right singular vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$, left singular vectors $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_r}$, and corresponding singular values $\sigma_1, \sigma_2, \ldots, \sigma_r$. Then*

$$A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T.$$

Proof: For each singular vector $\mathbf{v_j}$, $A\mathbf{v_j} = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T \mathbf{v_j}$. Since any vector $\mathbf{v}$ can be expressed as a linear combination of the singular vectors plus a vector perpendicular to the $\mathbf{v_i}$, $A\mathbf{v} = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T \mathbf{v}$ and by Lemma 1.4, $A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T$.

The decomposition is called the *singular value decomposition*, *SVD*, of A. In matrix notation $A = UDV^T$ where the columns of U and V consist of the left and right singular vectors, respectively, and D is a diagonal matrix whose diagonal entries are the singular values of A.

For any matrix A, the sequence of singular values is unique and if the singular values are all distinct, then the sequence of singular vectors is unique also. However, when some set of singular values are equal, the corresponding singular vectors span some subspace. Any set of orthonormal vectors spanning this subspace can be used as the singular vectors.

1.3 Best Rank k Approximations

There are two important matrix norms, the Frobenius norm denoted $||A||_F$ and the 2-norm denoted $||A||_2$. The 2-norm of the matrix A is given by

$$\max_{|\mathbf{v}|=1} |A\mathbf{v}|$$

and thus equals the largest singular value of the matrix.

Let A be an $n \times d$ matrix and think of the rows of A as n points in d-dimensional space. The Frobenius norm of A is the square root of the sum of the squared distance of the points to the origin. The 2-norm is the square root of the sum of squared distances to the origin along the direction that maximizes this quantity.

Let

$$A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T$$

be the SVD of A. For $k \in \{1, 2, \ldots, r\}$, let

$$A_k = \sum_{i=1}^{k} \sigma_i \mathbf{u_i} \mathbf{v_i}^T$$

be the sum truncated after k terms. It is clear that A_k has rank k. Furthermore, A_k is the best rank k approximation to A when the error is measured in either the 2-norm or the Frobenius norm.

Lemma 1.6 *The rows of A_k are the projections of the rows of A onto the subspace V_k spanned by the first k singular vectors of A.*

Proof: Let $\mathbf{a}$ be an arbitrary row vector. Since the $\mathbf{v_i}$ are orthonormal, the projection of the vector a onto V_k is given by $\sum_{i=1}^{k} (\mathbf{a} \cdot \mathbf{v_i}) \mathbf{v_i}^T$. Thus, the matrix whose rows are the projections of the rows of A onto V_k is given by $\sum_{i=1}^{k} A \mathbf{v_i} \mathbf{v_i}^T$. This last expression simplifies to

$$\sum_{i=1}^{k} A \mathbf{v_i} \mathbf{v_i}^T = \sum_{i=1}^{k} \sigma_i \mathbf{u_i} \mathbf{v_i}^T = A_k.$$

■

The matrix A_k is the best rank k approximation to A in both the Frobenius and the 2-norm. First we show that the matrix A_k is the best rank k approximation to A in the Frobenius norm.

Theorem 1.7 *For any matrix B of rank at most k*

$$\|A - A_k\|_F \leq \|A - B\|_F$$

Proof: Let B minimize $\|A - B\|_F^2$ among all rank k or less matrices. Let V be the space spanned by the rows of B. The dimension of V is at most k. Since B minimizes $\|A - B\|_F^2$, it must be that each row of B is the projection of the corresponding row of A onto V,

otherwise replacing the row of B with the projection of the corresponding row of A onto V does not change V and hence the rank of B but would reduce $\|A-B\|_F^2$. Since each row of B is the projection of the corresponding row of A, it follows that $\|A-B\|_F^2$ is the sum of squared distances of rows of A to V. Since A_k minimizes the sum of squared distance of rows of A to any k-dimensional subspace, from Theorem (1.1), it follows that $\|A-A_k\|_F \leq \|A-B\|_F$. ∎

Next we tackle the 2-norm. We first show that the square of the 2-norm of $A-A_k$ is the square of the $(k+1)^{st}$ singular value of A,

Lemma 1.8 $\|A-A_k\|_2^2 = \sigma_{k+1}^2$.

Proof: Let $A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i}\mathbf{v_i}^T$ be the singular value decomposition of A. Then $A_k = \sum_{i=1}^{k} \sigma_i \mathbf{u_i}\mathbf{v_i}^T$ and $A - A_k = \sum_{i=k+1}^{r} \sigma_i \mathbf{u_i}\mathbf{v_i}^T$. Let $\mathbf{v}$ be the top singular vector of $A-A_k$. Express $\mathbf{v}$ as a linear combination of $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_r}$. That is, write $\mathbf{v} = \sum_{i=1}^{r} \alpha_i \mathbf{v_i}$. Then

$$\begin{aligned} |(A-A_k)\mathbf{v}| &= \left| \sum_{i=k+1}^{r} \sigma_i \mathbf{u_i}\mathbf{v_i}^T \sum_{j=1}^{r} \alpha_j \mathbf{v_j} \right| = \left| \sum_{i=k+1}^{r} \alpha_i \sigma_i \mathbf{u_i}\mathbf{v_i}^T \mathbf{v_i} \right| \\ &= \left| \sum_{i=k+1}^{r} \alpha_i \sigma_i \mathbf{u_i} \right| = \sqrt{\sum_{i=k+1}^{r} \alpha_i^2 \sigma_i^2}. \end{aligned}$$

The $\mathbf{v}$ maximizing this last quantity, subject to the constraint that $|\mathbf{v}|^2 = \sum_{i=1}^{r} \alpha_i^2 = 1$, occurs when $\alpha_{k+1} = 1$ and the rest of the α_i are 0. Thus, $\|A-A_k\|_2^2 = \sigma_{k+1}^2$ proving the lemma. ∎

Finally, we prove that A_k is the best rank k 2-norm approximation to A.

Theorem 1.9 *Let A be an $n \times d$ matrix. For any matrix B of rank at most k*

$$\|A-A_k\|_2 \leq \|A-B\|_2$$

Proof: If A is of rank k or less, the theorem is obviously true since $\|A-A_k\|_2 = 0$. Thus assume that A is of rank greater than k. By Lemma 1.8, $\|A-A_k\|_2^2 = \sigma_{k+1}^2$. Now suppose there is some matrix B of rank at most k such that B is a better 2-norm approximation to A than A_k. That is, $\|A-B\|_2 < \sigma_{k+1}$. The null space of B, Null (B), (the set of vectors $\mathbf{v}$ such that $B\mathbf{v} = 0$) has dimension at least $d-k$. Let $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_{k+1}}$ be the first $k+1$ singular vectors of A. By a dimension argument, it follows that there exists a $\mathbf{z} \neq 0$ in

$$\text{Null}\,(B) \cap \text{Span}\left\{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_{k+1}}\right\}.$$

[Indeed, there are $d-k$ independent vectors in Null(B); say, $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_{d-k}}$ are any $d-k$ independent vectors in Null(B). Now, $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_{d-k}}, \mathbf{v_1}, \mathbf{v_2}, \ldots \mathbf{v_{k+1}}$ are $d+1$ vectors in d space and they are dependent, so there are real numbers $\alpha_1, \alpha_2, \ldots, \alpha_{d-k}$ and $\beta_1, \beta_2, \ldots, \beta_k$ not all zero so that $\sum_{i=1}^{d-k} \alpha_i \mathbf{u_i} = \sum_{j=1}^{k} \beta_j \mathbf{v_j}$. Take $\mathbf{z} = \sum_{i=1}^{d-k} \alpha_i \mathbf{u_i}$ and observe that $\mathbf{z}$ cannot be the zero vector.] Scale $\mathbf{z}$ so that $|\mathbf{z}| = 1$. We now show that for this vector $\mathbf{z}$, which lies in the space of the first $k+1$ singular vectors of A, that $(A-B)\,\mathbf{z} \geq \sigma_{k+1}$. Hence the 2-norm of $A-B$ is at least σ_{k+1} contradicting the assumption that $\|A-B\|_2 < \sigma_{k+1}$. First

$$\|A-B\|_2^2 \geq |(A-B)\,\mathbf{z}|^2 .$$

Since $B\mathbf{z} = 0$,

$$\|A-B\|_2^2 \geq |A\mathbf{z}|^2 .$$

Since $\mathbf{z}$ is in the Span $\{\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_{k+1}}\}$

$$|A\mathbf{z}|^2 = \left|\sum_{i=1}^{n} \sigma_i \mathbf{u_i} \mathbf{v_i}^T \mathbf{z}\right|^2 = \sum_{i=1}^{n} \sigma_i^2 \left(\mathbf{v_i}^T \mathbf{z}\right)^2 = \sum_{i=1}^{k+1} \sigma_i^2 \left(\mathbf{v_i}^T \mathbf{z}\right)^2 \geq \sigma_{k+1}^2 \sum_{i=1}^{k+1} \left(\mathbf{v_i}^T \mathbf{z}\right)^2 = \sigma_{k+1}^2.$$

It follows that

$$\|A-B\|_2^2 \geq \sigma_{k+1}^2$$

contradicting the assumption that $||A-B||_2 < \sigma_{k+1}$. This proves the theorem. ∎

1.4 Power Method for Computing the Singular Value Decomposition

Computing the singular value decomposition is an important branch of numerical analysis in which there have been many sophisticated developments over a long period of time. Here we present an "in-principle" method to establish that the approximate SVD of a matrix A can be computed in polynomial time. The reader is referred to numerical analysis texts for more details. The method we present, called the Power Method, is simple and is in fact the conceptual starting point for many algorithms. It is easiest to describe first in the case when A is square symmetric and has the same right and left singular vectors, namely,

$$A = \sum_{i=1}^{r} \sigma_i \mathbf{v_i} \mathbf{v_i}^T.$$

In this case, we have

$$A^2 = \left(\sum_{i=1}^{r} \sigma_i \mathbf{v_i} \mathbf{v_i}^T\right)\left(\sum_{j=1}^{r} \sigma_j \mathbf{v_j} \mathbf{v_j}^T\right) = \sum_{i,j=1}^{r} \sigma_i \sigma_j \mathbf{v_i} \mathbf{v_i}^T \mathbf{v_j} \mathbf{v_j}^T = \sum_{i=1}^{r} \sigma_i^2 \mathbf{v_i} \mathbf{v_i}^T,$$

where, first we just multiplied the two sums and then observed that if $i \neq j$, the dot product $\mathbf{v_i}^T \mathbf{v_j}$ equals 0 by orthogonality. [Caution: The "outer product" $\mathbf{v_i} \mathbf{v_j}^T$ is a matrix

and is not zero even for $i \neq j$.] Similarly, if we take the k th power of A, again all the cross terms are zero and we will get

$$A^k = \sum_{i=1}^{r} \sigma_i^k \mathbf{v_i} \mathbf{v_i}^T.$$

If we had $\sigma_1 > \sigma_2$, we would have

$$\frac{1}{\sigma_1^k} A^k \to \mathbf{v_1} \mathbf{v_1}^T.$$

Now we do not know σ_1 beforehand and cannot find this limit, but if we just take A^k and divide by $||A^k||_F$ so that the Frobenius norm is normalized to 1 now, that matrix will converge to the rank 1 matrix $\mathbf{v_1} \mathbf{v_1}^T$ from which $\mathbf{v_1}$ may be computed. [This is still an intuitive description, which we will make precise shortly. First, we cannot make the assumption that A is square and has the same right and left singular vectors. But, $B = AA^T$ satisfies both these conditions. If again, the SVD of A is $\sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^T$, then by direct multiplication

$$\begin{aligned} B = AA^T &= \left(\sum_i \sigma_i \mathbf{u_i} \mathbf{v_i}^T \right) \left(\sum_j \sigma_j \mathbf{v_j} \mathbf{u_j}^T \right) \\ &= \sum_{i,j} \sigma_i \sigma_j \mathbf{u_i} \mathbf{v_i}^T \mathbf{v_j} \mathbf{u_j}^T = \sum_{i,j} \sigma_i \sigma_j \mathbf{u_i} (\mathbf{v}_i^T \cdot \mathbf{v_j}) \mathbf{u_j}^T \\ &= \sum_i \sigma_i^2 \mathbf{u_i} \mathbf{u_i}^T, \end{aligned}$$

since $\mathbf{v_i}^T \mathbf{v_j}$ is the dot product of the two vectors and is zero unless $i = j$. This is the spectral decomposition of B. Using the same kind of calculation as above,

$$B^k = \sum_i \sigma_i^{2k} \mathbf{u_i} \mathbf{u_i}^T.$$

As k increases, for $i > 1$, $\sigma_i^{2k} / \sigma_1^{2k}$ goes to zero and B^k is approximately equal to

$$\sigma_1^{2k} \mathbf{u_1} \mathbf{u_1}^T$$

provided that for each $i > 1$, $\sigma_i(A) < \sigma_1(A)$.

This suggests a way of finding σ_1 and $\mathbf{u_1}$, by successively powering B. But there are two issues. First, if there is a significant gap between the first and second singular values of a matrix, then the above argument applies and the power method will quickly converge to the first left singular vector. Suppose there is no significant gap. In the extreme case, there may be ties for the top singular value. Then the above argument does not work. There are cumbersome ways of overcoming this by assuming a "gap" between σ_1 and σ_2;

such proofs do have the advantage that with a greater gap, better results can be proved, but at the cost of some mess. Here, instead, we will adopt a clean solution in Theorem 1.11 below which states that even with ties, the power method converges to some vector in the span of those singular vectors corresponding to the "nearly highest" singular values.

A second issue is that computing B^k costs k matrix multiplications when done in a straight-forward manner or $O(\log k)$ when done by successive squaring. Instead we compute

$$B^k\mathbf{x}$$

where $\mathbf{x}$ is a random unit length vector, the idea being that the component of $\mathbf{x}$ in the direction of $\mathbf{u_1}$ would get multiplied by σ_1^2 each time, while the component of $\mathbf{x}$ along other $\mathbf{u_i}$ would be multiplied only by σ_i^2. Of course, if the component of $\mathbf{x}$ along $\mathbf{u_1}$ is zero to start with, this would not help at all - it would always remain 0. But, this problem is fixed by picking $\mathbf{x}$ to be random as we show in Lemma (1.10).

Each increase in k requires multiplying B by the vector $B^{k-1}\mathbf{x}$, which we can further break up into

$$B^k\mathbf{x} = A\left(A^T\left(B^{k-1}\mathbf{x}\right)\right).$$

This requires two matrix-vector products, involving the matrices A^T and A. In many applications, data matrices are sparse - many entries are zero. [A leading example is the matrix of hypertext links in the web. There are more than 10^{10} web pages and the matrix would be $10^{10}$by 10^{10}. But on the average only about 10 entries per row are non-zero; so only about 10^{11} of the possible 10^{20} entries are non-zero.] Sparse matrices are often represented by giving just a linked list of the non-zero entries and their values. If A is represented in this sparse manner, then the reader can convince him/herself that we can a matrix vector product in time proportional to the number of nonzero entries in A. Since $B^k\mathbf{x} \approx \sigma_1^{2k}\mathbf{u_1}(\mathbf{u}_1^T \cdot \mathbf{x})$ is a scalar multiple of $\mathbf{u_1}$, $\mathbf{u_1}$ can be recovered from $B^k\mathbf{x}$ by normalization.

We start with a technical Lemma needed in the proof of the theorem.

Lemma 1.10 *Let $(x_1, x_2, \ldots, x_d)$ be a unit d-dimensional vector picked at random from the set $\{\mathbf{x} : |\mathbf{x}| \leq 1\}$. The probability that $|x_1| \geq \frac{1}{20\sqrt{d}}$ is at least 9/10.*

Proof: We first show that for a vector $\mathbf{v}$ picked at random with $|\mathbf{v}| \leq 1$, the probability that $v_1 \geq \frac{1}{20\sqrt{d}}$ is at least 9/10. Then we let $\mathbf{x} = \mathbf{v}/|\mathbf{v}|$. This can only increase the value of v_1, so the result follows.

Let $\alpha = \frac{1}{20\sqrt{d}}$. The probability that $|v_1| \geq \alpha$ equals one minus the probability that $|v_1| \leq \alpha$. The probability that $|v_1| \leq \alpha$ is equal to the fraction of the volume of the unit sphere with $|v_1| \leq \alpha$. To get an upper bound on the volume of the sphere with $|v_1| \leq \alpha$, consider twice the volume of the unit radius cylinder of height α. The volume of the

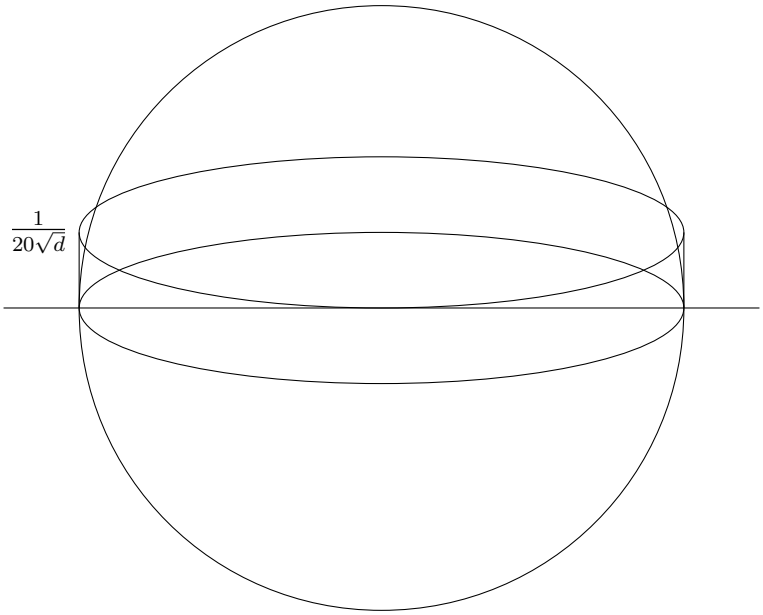


Figure 1.3: The volume of the cylinder of height $\frac{1}{20\sqrt{d}}$ is an upper bound on the volume of the hemisphere below $x_1 = \frac{1}{20\sqrt{d}}$

portion of the sphere with $|v_1| \leq \alpha$ is less than or equal to $2\alpha V(d-1)$ (recall notation from Chapter 2) and

$$\text{Prob}(|v_1| \leq \alpha) \leq \frac{2\alpha V(d-1)}{V(d)}$$

Now the volume of the unit radius sphere is at least twice the volume of the cylinder of height $\frac{1}{\sqrt{d-1}}$ and radius $\sqrt{1-\frac{1}{d-1}}$ or

$$V(d) \geq \frac{2}{\sqrt{d-1}} V(d-1)(1-\frac{1}{d-1})^{\frac{d-1}{2}}$$

Using $(1-x)^a \geq 1-ax$

$$V(d) \geq \frac{2}{\sqrt{d-1}} V(d-1)(1-\frac{d-1}{2}\frac{1}{d-1}) \geq \frac{V(d-1)}{\sqrt{d-1}}$$

and

$$\text{Prob}(|v_1| \leq \alpha) \leq \frac{2\alpha V(d-1)}{\frac{1}{\sqrt{d-1}} V(d-1)} \leq \frac{\sqrt{d-1}}{10\sqrt{d}} \leq \frac{1}{10}.$$

Thus the probability that $v_1 \geq \frac{1}{20\sqrt{d}}$ is at least 9/10. ∎

Theorem 1.11 *Let A be an $n \times d$ matrix and $\mathbf{x}$ a random unit length vector. Let V be the space spanned by the left singular vectors of A corresponding to singular values greater*

than $(1-\varepsilon)\,\sigma_1$. *Let* k *be* $\Omega\left(\frac{\ln(d/\varepsilon)}{\varepsilon}\right)$. *Let* $\mathbf{w}$ *be unit vector after* k *iterations of the power method, namely,*

$$\mathbf{w} = \frac{\left(AA^T\right)^k \mathbf{x}}{\left|\left(AA^T\right)^k \mathbf{x}\right|}.$$

The probability that $\mathbf{w}$ *has a component of at least* ϵ *perpendicular to* V *is at most 1/10.*

Proof: Let

$$A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T$$

be the SVD of A. If the rank of A is less than d, then complete $\{\mathbf{u_1}, \mathbf{u_2}, \ldots \mathbf{u}_r\}$ into a basis $\{\mathbf{u_1}, \mathbf{u_2}, \ldots \mathbf{u}_d\}$ of d-space. Write $\mathbf{x}$ in the basis of the $\mathbf{u_i}'s$ as

$$\mathbf{x} = \sum_{i=1}^{n} c_i \mathbf{u_i}.$$

Since $(AA^T)^k = \sum_{i=1}^{d} \sigma_i^{2k} \mathbf{u_i} \mathbf{u_i}^T$, it follows that $(AA^T)^k \mathbf{x} = \sum_{i=1}^{d} \sigma_i^{2k} c_i \mathbf{u_i}$. For a random unit length vector $\mathbf{x}$ picked independent of A, the $\mathbf{u_i}$ are fixed vectors and picking $\mathbf{x}$ at random is equivalent to picking random c_i. From Lemma 1.10, $|c_1| \geq \frac{1}{20\sqrt{d}}$ with probability at least 9/10.

Suppose that $\sigma_1, \sigma_2, \ldots, \sigma_m$ are the singular values of A that are greater than or equal to $(1-\varepsilon)\,\sigma_1$ and that $\sigma_{m+1}, \ldots, \sigma_n$ are the singular values that are less than $(1-\varepsilon)\,\sigma_1$. Now

$$|(AA^T)^k \mathbf{x}|^2 = \left|\sum_{i=1}^{n} \sigma_i^{2k} c_i \mathbf{u_i}\right|^2 = \sum_{i=1}^{n} \sigma_i^{4k} c_i^2 \geq \sigma_1^{4k} c_1^2 \geq \frac{1}{400d} \sigma_1^{4k},$$

with probability at least 9/10. Here we used the fact that a sum of positive quantities is at least as large as its first element and the first element is greater than or equal to $\frac{1}{400d}\sigma_1^{4k}$ with probability at least 9/10. [Note: If we did not choose $\mathbf{x}$ at random in the beginning and use Lemma 1.10, c_1 could well have been zero and this argument would not work.]

The component of $|(AA^T)^k \mathbf{x}|^2$ perpendicular to the space V is

$$\sum_{i=m+1}^{d} \sigma_i^{4k} c_i^2 \leq (1-\varepsilon)^{4k}\, \sigma_1^{4k} \sum_{i=m+1}^{d} c_i^2 \leq (1-\varepsilon)^{4k}\, \sigma_1^{4k},$$

since $\sum_{i=1}^{n} c_i^2 = |\mathbf{x}| = 1$. Thus, the component of $\mathbf{w}$ perpendicular to V is at most

$$\frac{(1-\varepsilon)^{2k}\sigma_1^{2k}}{\frac{1}{20\sqrt{d}}\sigma_1^{2k}} = O(\sqrt{d}(1-\varepsilon)^{2k}) = O(\sqrt{d}e^{-2\varepsilon k}) = O\left(\sqrt{d}e^{-\Omega(\ln(d/\varepsilon))}\right) = O(\varepsilon)$$

as desired. ∎

1.5 Applications of Singular Value Decomposition

1.5.1 Principal Component Analysis

The traditional use of SVD is in Principal Component Analysis (PCA). PCA is illustrated by an example - customer-product data where there are n customers buying d products. Let matrix A with elements a_{ij} represent the probability of customer i purchasing product j (or the amount or utility of product j to customer i). One hypothesizes that there are really only k underlying basic factors like age, income, family size, etc. that determine a customer's purchase behavior. An individual customer's behavior is determined by some weighted combination of these underlying factors. That is, a customer's purchase behavior can be characterized by a k-dimensional vector where k is much smaller that n and d. The components of the vector are weights for each of the basic factors. Associated with each basic factor is a vector of probabilities, each component of which is the probability of purchasing a given product by someone whose behavior depends only on that factor. More abstractly, A is an $n \times d$ matrix that can be expressed as the product of two matrices U and V where U is an $n \times k$ matrix expressing the factor weights for each customer and V is a $k \times d$ matrix expressing the purchase probabilities of products that correspond to that factor. One twist is that A may not be exactly equal to UV, but close to it since there may be noise or random perturbations.

Taking the best rank k approximation A_k from SVD (as described above) gives us such a U, V. In this traditional setting, one assumed that A was available fully and we wished to find U, V to identify the basic factors or in some applications to "denoise" A (if we think of $A - UV$ as noise). Now imagine that n and d are very large, on the order of thousands or even millions, there is probably little one could do to estimate or even store A. In this setting, we may assume that we are given just given a few elements of A and wish to estimate A. If A was an arbitrary matrix of size $n \times d$, this would require $\Omega(nd)$ pieces of information and cannot be done with a few entries. But again hypothesize that A was a small rank matrix with added noise. If now we also assume that the given entries are randomly drawn according to some known distribution, then there is a possibility that SVD can be used to estimate the whole of A. This area is called collaborative filtering and one of its uses is to target an ad to a customer based on one or two purchases. We will not be able to describe it here.

1.5.2 Clustering a Mixture of Spherical Gaussians

In clustering, we are given a set of points in $d-$space and the task is to partition the points into k subsets (clusters) where each cluster consists of "nearby" points. Different definitions of the goodness of a clustering lead to different solutions. Clustering is an important area which we will study in detail in Chapter **??**. Here we will see how to solve a particular clustering problem using Singular Value Decomposition.

In general, a solution to any clustering problem comes up with k *cluster centers*

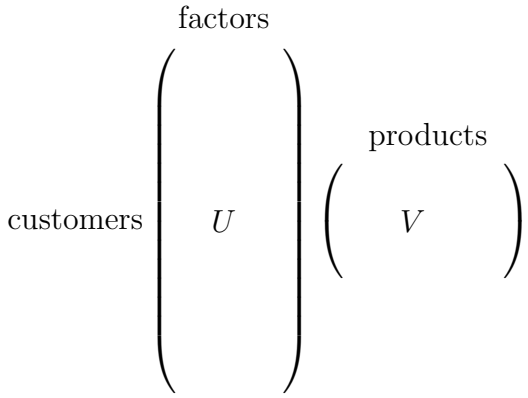


Figure 1.4: Customer-product data

which define the k clusters - a cluster is the set of data points which have a particular cluster center as the closest cluster center. Hence the Vornoi cells of the cluster centers determine the clusters. Using this observation, it is relatively easy to cluster points in two or three dimensions. However, clustering is not so easy in higher dimensions. Many problems have high-dimensional data and clustering problems are no exception. Clustering problems tend to be NP-hard, so we do not have polynomial time algorithms to solve them. One way around this is to assume stochastic models of input data and devise algorithms to cluster under such models.

Mixture models are a very important class of stochastic models. A mixture is a probability density or distribution that is the weighted sum of simple component probability densities. It is of the form $w_1p_1 + w_2p_2 + \cdots + w_kp_k$ where $p_1, p_2, \ldots, p_k$ are the basic densities and $w_1, w_2, \ldots, w_k$ are positive real numbers called weights that add up to one. Clearly, $w_1p_1 + w_2p_2 + \cdots + w_kp_k$ is a probability density, it integrates to one.

PUT IN PICTURE OF A 1-DIMENSIONAL GAUSSIAN MIXTURE

The *model fitting problem* is to fit a mixture of k basic densities to n samples, each sample drawn according to the same mixture distribution. The class of basic densities is known, but the component weights of the mixture are not. Here, we deal with the case where the basic densities are all spherical Gaussians. The samples are generated by picking an integer i from the set $\{1, 2, \ldots, k\}$ with probabilities $w_1, w_2, \ldots, w_k$, respectively. Then, picking a sample according to p_i and repeating the process n times. This process generates n samples according to the mixture where the set of samples is naturally partitioned into k sets, each set corresponding to one p_i.

The model-fitting problem consists of two sub problems. The first sub problem is to cluster the sample into k subsets where each subset was picked according to one component density. The second sub problem is to fit a distribution to each subset. We discuss

only the clustering problem here. The problem of fitting a single Gaussian to a set of data points is a lot easier and was discussed in Section **??** of Chapter 2.

If the component Gaussians in the mixture have their centers very close together, then the clustering problem is unresolvable. In the limiting case where a pair of component densities are the same, there is no way to distinguish between them. What condition on the inter-center separation will guarantee unambiguous clustering? First, by looking at 1-dimensional examples, it is clear that this separation should be measured in units of the standard deviation, since the density is a function of the number of standard deviation from the mean. In one dimension, if two Gaussians have inter-center separation at least ten times the maximum of their standard deviations, then they hardly overlap. What is the analog of this statement in higher dimensions?

For a d-dimensional spherical Gaussian, with standard deviation σ in each direction[1], it is easy to see that the expected distance squared from the center is $d\sigma^2$. Define the radius r of the Gaussian to be the square root of the average distance squared from the center; so r is $\sqrt{d}\sigma$. If the inter-center separation between two spherical Gaussians, both of radius r is at least $2r = 2\sqrt{d}\sigma$, then it is easy to see that the densities hardly overlap. But this separation requirement grows with d. For problems with large d, this would impose a separation requirement not met in practice. The main aim is to answer affirmatively the question:

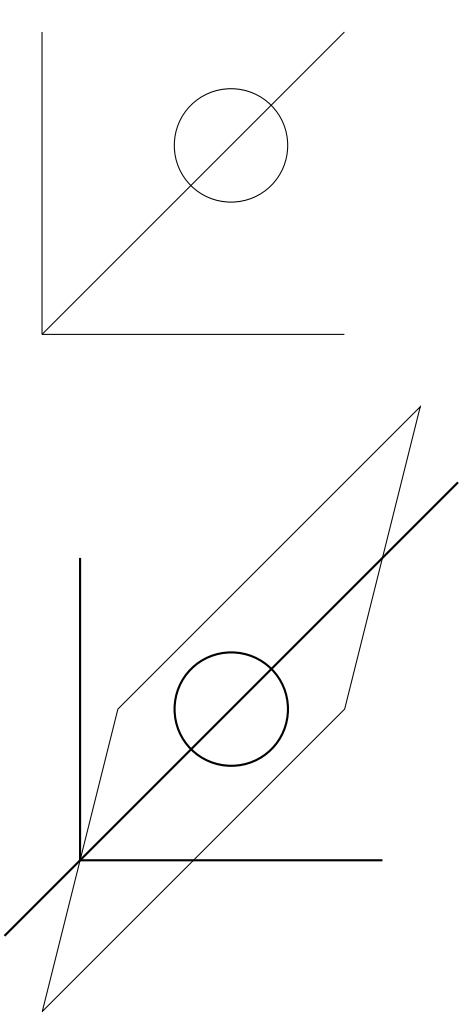
Can we show an analog of the 1-dimensional statement for large d : In a mixture of k spherical Gaussians in d space, if the centers of each pair of component Gaussians are $\Omega(1)$ standard deviations apart, then we can separate the sample points into the k components (for $k \in O(1)$).

The central idea is the following: Suppose for a moment, we can (magically) find the subspace spanned by the k centers. Imagine projecting the sample points to this subspace. It is easy to see (see Lemma (1.12) below) that the projection of a spherical Gaussian with standard deviation σ is still a spherical Gaussian with standard deviation σ. But in the projection, now, the inter-center separation still remains the same. So in the projection, the Gaussians are distinct provided the inter-center separation (in the whole space) is $\Omega(\sqrt{k}\sigma)$ which is a lot smaller than the $\Omega(\sqrt{d}\sigma)$ for $k << d$. Interestingly we will see that the subspace spanned by the $k-$ centers is essentially the best-fit k dimensional subspace which we can find by Singular Value Decomposition.

Lemma 1.12 *Suppose p is a d-dimensional spherical Gaussian with center μ and standard deviation σ. The density of p projected onto an arbitrary k-dimensional subspace V is a spherical Gaussian with the same standard deviation.*

Proof: Since p is spherical, the projection is independent of the k-dimensional subspace. Pick V to be the subspace spanned by the first k coordinate vectors. For a point $\mathbf{x} =$

[1]Since a spherical Gaussian has the same standard deviation in every direction, we call it the standard deviation of the Gaussian.



1. Best fit 1-dimension subspace to a spherical Gaussian is the line through its center and the origin.
2. Any k-dimensional subspace containing the line is a best fit k-dimensional subspace for the Gaussian.
3. The best fit k-dimensional subspace for k Gaussians is the subspace containing their centers.

1. Best fit 1-dimension subspace to a spherical Gaussian is the line through its center and the origin.
2. Any k-dimensional subspace containing the line is a best fit k-dimensional subspace for the Gaussian.
3. The best fit k-dimensional subspace for k Gaussians is the subspace containing their centers.

Figure 1.5: Best fit subspace to a spherical Gaussian.

$(x_1, x_2, \ldots, x_d)$, we will use the notation $\mathbf{x}' = (x_1, x_2, \ldots x_k)$ and $\mathbf{x}'' = (x_{k+1}, x_{k+2}, \ldots, x_n)$. The density of the projected Gaussian at the point $(x_1, x_2, \ldots, x_k)$ is

$$ce^{-\frac{|\mathbf{x}'-\mu'|^2}{2\sigma^2}} \int_{\mathbf{x}''} e^{-\frac{|\mathbf{x}''-\mu''|^2}{2\sigma^2}} d\mathbf{x}'' = c'e^{-\frac{|\mathbf{x}'-\mu'|^2}{2\sigma^2}}.$$

This clearly implies the lemma. ∎

We now show that the top k singular vectors produced by the SVD span the space of the k centers. First, we extend the notion of best fit to probability distributions. Then we show that for a single spherical Gaussian (whose center is not the origin), the best fit 1-dimensional subspace is the line though the center of the Gaussian and the origin. Next, we show that the best fit k-dimensional subspace for a single Gaussian (whose center is

not the origin) is any k-dimensional subspace containing the line through the Gaussian's center and the origin. Finally, for k spherical Gaussians, the best fit k-dimensional subspace is the subspace containing their centers. Thus, the SVD finds the subspace that contains the centers.

Recall that for a set of points, the best-fit line is the line passing through the origin which minimizes the sum of squared distances to the points. We extend this definition to probability densities instead of a set of points.

Definition 4.1: If p is a probability density in d space, the best fit line for p is the line l passing through the origin that minimizes the expected squared (perpendicular) distance to the line, namely,

$$\int \text{dist}\left(\mathbf{x}, l\right)^2 p\left(\mathbf{x}\right) d\mathbf{x}.$$

Recall that a k-dimensional subspace is the best-fit subspace if the sum of squared distances to it is minimized or equivalently, the sum of squared lengths of projections onto it is maximized. This was defined for a set of points, but again it can be extended to a density as above.

Definition 4.2: If p is a probability density in d-space and V is a subspace, then the expected squared perpendicular distance of V to p, denoted $f(V,p)$, is given by

$$f(V,p) = \int \text{dist}^2\left(\mathbf{x}, V\right) p\left(\mathbf{x}\right) d\mathbf{x},$$

where $\text{dist}(\mathbf{x}, V)$ denotes the perpendicular distance from the point $\mathbf{x}$ to the subspace V.

For the uniform density on the unit circle centered at the origin, it is easy to see that any line passing through the origin is a best fit line for the probability distribution.

Lemma 1.13 *Let the probability density p be a spherical Gaussian with center $\boldsymbol{\mu} \neq 0$. The best fit 1-dimensional subspace is the line passing through $\boldsymbol{\mu}$ and the origin.*

Proof: For a randomly chosen $\boldsymbol{x}$ (according to p) and a fixed unit length vector $\mathbf{v}$,

$$\begin{aligned}
E\left[(\mathbf{v}^T\mathbf{x})^2\right] &= E\left[\left(\mathbf{v}^T\left(\mathbf{x}-\boldsymbol{\mu}\right)+\mathbf{v}^T\boldsymbol{\mu}\right)^2\right] \\
&= E\left[\left(\mathbf{v}^T\left(\mathbf{x}-\boldsymbol{\mu}\right)\right)^2 + 2\left(\mathbf{v}^T\boldsymbol{\mu}\right)\left(\mathbf{v}^T\left(\mathbf{x}-\boldsymbol{\mu}\right)\right) + \left(\mathbf{v}^T\boldsymbol{\mu}\right)^2\right] \\
&= E\left[\left(\mathbf{v}^T\left(\mathbf{x}-\boldsymbol{\mu}\right)\right)^2\right] + 2\left(\mathbf{v}^T\boldsymbol{\mu}\right)E\left[\mathbf{v}^T\left(\mathbf{x}-\boldsymbol{\mu}\right)\right] + \left(\mathbf{v}^T\boldsymbol{\mu}\right)^2 \\
&= E\left[\left(\mathbf{v}^T\left(\mathbf{x}-\boldsymbol{\mu}\right)\right)^2\right] + \left(\mathbf{v}^T\boldsymbol{\mu}\right)^2 \\
&= \sigma^2 + \left(\mathbf{v}^T\boldsymbol{\mu}\right)^2
\end{aligned}$$

since $E\left[\left(\mathbf{v}^T(\mathbf{x}-\boldsymbol{\mu})\right)^2\right]$ is the variance in the direction $\mathbf{v}$ and $E\left(\mathbf{v}^T(\mathbf{x}-\boldsymbol{\mu})\right)=0$. The lemma follows from the fact that the best fit line $\mathbf{v}$ is the one that maximizes $\left(\mathbf{v}^T\mathbf{u}\right)^2$ which is maximized when $\mathbf{v}$ is aligned with the center $\boldsymbol{\mu}$. ∎

Lemma 1.14 *For a spherical Gaussian with center $\boldsymbol{\mu}$, a k-dimensional subspace is a best fit subspace if and only if it contains $\boldsymbol{\mu}$.*

Proof: By symmetry, every k-dimensional subspace through $\boldsymbol{\mu}$ has the same sum of distances squared to the density. Now by the SVD procedure, we know that the best-fit k-dimensional subspace contains the best fit line, i.e., contains $\boldsymbol{\mu}$. Thus, the lemma follows. ∎

This immediately leads to the following theorem.

Theorem 1.15 *If p is a mixture of k spherical Gaussians whose centers span a k-dimensional subspace, then the best fit k-dimensional subspace is the one containing the centers.*

Proof: Let p be the mixture $w_1p_1+w_2p_2+\cdots+w_kp_k$. Let V be any subspace of dimension k or less. Then, the expected squared perpendicular distance of V to p is

$$\begin{aligned} f(V,p) &= \int \text{dist}^2(\mathbf{x},V)p(\mathbf{x})d\mathbf{x} \\ &= \sum_{i=1}^{k} w_i \int \text{dist}^2(\mathbf{x},V)p_i(\mathbf{x})d\mathbf{x} \\ &\geq \sum_{i=1}^{k} w_i(\text{ distance squared of } p_i \text{ to its best fit } k\text{-dimensional subspace}). \end{aligned}$$

Choose V to be the space spanned by the centers of the densities p_i. By Lemma **??** the last inequality becomes an equality proving the theorem. ∎

For an infinite set of points drawn according to the mixture, the k-dimensional SVD subspace gives exactly the space of the centers. In reality, we have only a large number of samples drawn according to the mixture. However, it is intuitively clear that as the number of samples increases, the set of sample points approximates the probability density and so the SVD subspace of the sample is close to the space spanned by the centers. The details of how close it gets as a function of the number of samples are technical and we do not carry this out here.

1.5.3 An Application of SVD to a Discrete Optimization Problem

In the last example, SVD was used as a dimension reduction technique. It found a k-dimensional subspace (the space of centers) of a d-dimensional space and made the Gaussian clustering problem easier by projecting the data to the subspace. Here, instead of fitting a model to data, we have an optimization problem. Again applying dimension reduction to the data makes the problem easier. The use of SVD to solve discrete optimization problems is a relatively new subject with many applications. We start with an important NP-hard problem, the Maximum Cut Problem for a directed graph $G(V, E)$.

The Maximum Cut Problem is to partition the node set V of a directed graph into two subsets S and $\bar{S}$ so that the number of edges from S to $\bar{S}$ is maximized. Let A be the adjacency matrix of the graph. With each vertex i, associate an indicator variable x_i. The variable x_i will be set to 1 for $i \in S$ and 0 for $i \in \bar{S}$. The vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is unknown and we are trying to find it (or equivalently the cut), so as to maximize the number of edges across the cut. The number of edges across the cut is precisely

$$\sum_{i,j} x_i(1 - x_j)a_{ij}.$$

Thus, the Maximum Cut Problem can be posed as the optimization problem

$$\text{Maximize} \sum_{i,j} x_i(1 - x_j)a_{ij} \quad \text{subject to } x_i \in \{0, 1\}.$$

In matrix notation,

$$\sum_{i,j} x_i(1 - x_j)a_{ij} = \mathbf{x}^T A(\mathbf{1} - \mathbf{x}),$$

where $\mathbf{1}$ denotes the vector of all 1's . So, the problem can be restated as

$$\text{Maximize } \mathbf{x}^T A(\mathbf{1} - \mathbf{x}) \quad \text{subject to } x_i \in \{0, 1\}. \tag{1.1}$$

The SVD is used to solve this problem approximately by computing the SVD of A and replacing A by $A_k = \sum_{i=1}^{k} \sigma_i \mathbf{u_i} \mathbf{v_i}^T$ in (1.1) to get

$$\text{Maximize } \mathbf{x}^T A_k(\mathbf{1} - \mathbf{x}) \quad \text{subject to } x_i \in \{0, 1\}. \tag{1.2}$$

Note that the matrix A_k is no longer a 0-1 adjacency matrix.

We will show that:

1. For each 0-1 vector $\mathbf{x}$, $\mathbf{x}^T A_k(\mathbf{1} - \mathbf{x})$ and $\mathbf{x}^T A(\mathbf{1} - \mathbf{x})$ differ by at most $\frac{n^2}{\sqrt{k+1}}$. Thus, the maxima in (1.1) and (1.2) differ by at most this amount.

2. A near optimal $\mathbf{x}$ for (1.2) can be found by exploiting the low rank of A_k, which by Item 1 is near optimal for (1.1) where near optimal means with additive error of at most $\frac{n^2}{\sqrt{k+1}}$.

First, we prove Item 1. Since $\mathbf{x}$ and $\mathbf{1}-\mathbf{x}$ are 0-1 n-vectors, each has length at most $\sqrt{n}$. By the definition of the 2-norm, $|(A-A_k)(\mathbf{1}-\mathbf{x})| \leq \sqrt{n}||A-A_k||_2$. Now since $\mathbf{x}^T(A-A_k)(\mathbf{1}-\mathbf{x})$ is the dot product of the vector $\mathbf{x}$ with the vector $(A-A_k)(\mathbf{1}-\mathbf{x})$,

$$|\mathbf{x}^T(A-A_k)(\mathbf{1}-\mathbf{x})| \leq n||A-A_k||_2.$$

By Lemma 1.8, $||A-A_k||_2 = \sigma_{k+1}(A)$. The inequalities,

$$(k+1)\sigma_{k+1}^2 \leq \sigma_1^2+\sigma_2^2+\cdots\sigma_{k+1}^2 \leq ||A||_F^2 = \sum_{i,j} a_{ij}^2 \leq n^2$$

imply that $\sigma_{k+1}^2 \leq \frac{n^2}{k+1}$ and hence $||A-A_k||_2 \leq \frac{n}{\sqrt{k+1}}$ proving Item 1.

Next we focus on Item 2. It is instructive to look at the special case when k=1 and A is approximated by the rank one matrix A_1. An even more special case when the left and right singular vectors $\mathbf{u}$ and $\mathbf{v}$ are required to be identical is already NP-hard to solve exactly because it subsumes the problem of whether for a set of n integers, $\{a_1, a_2, \ldots, a_n\}$, there is a partition into two subsets whose sums are equal. So, we look for algorithms that solve the Maximum Cut Problem approximately.

For Item 2, we want to maximize $\sum_{i=1}^k \sigma_i(\mathbf{x}^T\mathbf{u_i})(\mathbf{v_i}^T(\mathbf{1}-\mathbf{x}))$ over 0-1 vectors $\mathbf{x}$. A piece of notation will be useful. For any $S \subseteq \{1,2,\ldots n\}$, write $\mathbf{u_i}(S)$ for the sum of coordinates of the vector $\mathbf{u_i}$ corresponding to elements in the set S and also for $\mathbf{v_i}$. That is, $\mathbf{u}_i(S) = \sum_{j\in S} u_{ij}$. We will maximize $\sum_{i=1}^k \sigma_i \mathbf{u_i}(S)\mathbf{v_i}(\bar{S})$ using dynamic programming.

For a subset S of $\{1,2,\ldots,n\}$, define the $2k$-dimensional vector $\mathbf{w}(S) = (\mathbf{u_1}(S), \mathbf{v_1}(\bar{S}), \mathbf{u_2}(S), \mathbf{v_2}(\bar{S}), \ldots,$ If we had the list of all such vectors, we could find $\sum_{i=1}^k \sigma_i \mathbf{u_i}(S)\mathbf{v_i}(\bar{S})$ for each of them and take the maximum. There are 2^n subsets S, but several S could have the same $\mathbf{w}(S)$ and in that case it suffices to list just one of them. Round each coordinate of each $\mathbf{u_i}$ to the nearest integer multiple of $\frac{1}{nk^2}$. Call the rounded vector $\tilde{\mathbf{u}}_\mathbf{i}$. Similarly obtain $\tilde{\mathbf{v}}_\mathbf{i}$. Let $\tilde{\mathbf{w}}(S)$ denote the vector $(\tilde{\mathbf{u}}_\mathbf{1}(S), \tilde{\mathbf{v}}_\mathbf{1}(\bar{S}), \tilde{\mathbf{u}}_\mathbf{2}(S), \tilde{\mathbf{v}}_\mathbf{2}(\bar{S}), \ldots, \tilde{\mathbf{u}}_\mathbf{k}(S), \tilde{\mathbf{v}}_\mathbf{k}(\bar{S}))$. We will construct a list of all possible values of the vector $\tilde{\mathbf{w}}(S)$. [Again, if several different S 's lead to the same vector $\tilde{\mathbf{w}}(S)$, we will keep only one copy on the list.] The list will be constructed by Dynamic Programming. For the recursive step of Dynamic Programming, assume we already have a list of all such vectors for $S \subseteq \{1,2,\ldots,i\}$ and wish to construct the list for $S \subseteq \{1,2,\ldots,i+1\}$. Each $S \subseteq \{1,2,\ldots,i\}$ leads to two possible $S' \subseteq \{1,2,\ldots,i+1\}$, namely, S and $S \cup \{i+1\}$. In the first case, the vector $\tilde{\mathbf{w}}(S') = (\tilde{\mathbf{u}}_\mathbf{1}(S) + \tilde{u}_{1,i+1}, \tilde{\mathbf{v}}_\mathbf{1}(S), \tilde{\mathbf{u}}_\mathbf{2}(S) + \tilde{u}_{2,i+1}, \tilde{\mathbf{v}}_\mathbf{2}(S), \ldots, ...)$. In the second case, $\tilde{\mathbf{w}}(S') = (\tilde{\mathbf{u}}_\mathbf{1}(S), \tilde{\mathbf{v}}_\mathbf{1}(S) + \tilde{v}_{1,i+1}, \tilde{\mathbf{u}}_\mathbf{2}(S), \tilde{\mathbf{v}}_\mathbf{2}(S) + \tilde{v}_{2,i+1}, \ldots, ...)$. We put in these two vectors for each vector in the previous list. Then, crucially, we prune - i.e., eliminate duplicates.

Assume that k is constant. Now, we show that the error is at most $\frac{n^2}{\sqrt{k+1}}$ as claimed. Since $\mathbf{u_i}, \mathbf{v_i}$ are unit length vectors, $|\mathbf{u_i}(S)|, |\mathbf{v_i}(S)| \leq \sqrt{n}$. Also $|\tilde{\mathbf{u}}_\mathbf{i}(S) - \mathbf{u_i}(S)| \leq \frac{n}{nk^2} = \frac{1}{k^2}$ and similarly for $\mathbf{v_i}$. To bound the error, we use an elementary fact: if a, b are reals with $|a|, |b| \leq M$ and we estimate a by a' and b by b' so that $|a-a'|, |b-b'| \leq \delta \leq M$, then

$$|ab - a'b'| = |a(b-b') + b'(a-a')| \leq |a||b-b'| + (|b| + |b-b'|)|a-a'| \leq 3M\delta.$$

Using this, we get that

$$\left| \sum_{i=1}^{k} \sigma_i \tilde{\mathbf{u}}_\mathbf{i}(S) \tilde{\mathbf{v}}_\mathbf{i}(\bar{S}) \quad - \quad \sum_{i=1}^{k} \sigma_i \mathbf{u}_\mathbf{i}(S) \mathbf{v}_\mathbf{i}(S) \right| \leq 3k\sigma_1 \sqrt{n}/k^2 \leq 3n^{3/2}/k,$$

and this meets the claimed error bound.

Next, we show that the running time is polynomially bounded. $|\tilde{\mathbf{u}}_\mathbf{i}(S)|, |\tilde{\mathbf{v}}_\mathbf{i}(S)| \leq 2\sqrt{n}$. Since $\tilde{\mathbf{u}}_\mathbf{i}(S), \tilde{\mathbf{v}}_\mathbf{i}(S)$ are all integer multiples of $1/(nk^2)$, there are at most $2/\sqrt{n}k^2$ possible values of $\tilde{\mathbf{u}}_\mathbf{i}(S), \tilde{\mathbf{v}}_\mathbf{i}(S)$ from which it follows that the list of $\tilde{\mathbf{w}}(S)$ never gets larger than $(1/\sqrt{n}k^2)^{2k}$ which for fixed k is polynomially bounded.

We summarize what we have accomplished.

Theorem 1.16 *Given a directed graph $G(V, E)$, a cut of size at least the maximum cut minus $O\left(\frac{n^2}{\sqrt{k}}\right)$ can be computed in polynomial time n for any fixed k.*

It would be quite a surprise to have an algorithm that actually achieves the same accuracy in time polynomial in n and k because this would give an exact max cut in polynomial time.

1.5.4 SVD as a Compression Algorithm

Suppose A is the pixel intensity matrix of a large image. The entry a_{ij} gives the intensity of the ij^{th} pixel. If A is $n \times n$, the transmission of A requires transmitting $O(n^2)$ real numbers. Instead, one could send A_k, that is, the top k singular values $\sigma_1, \sigma_2, \ldots, \sigma_k$ along with the left and right singular vectors $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_k}$, and $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}$. This would require sending $O(kn)$ real numbers instead of $O(n^2)$ real numbers. If k is much smaller than n, this results in savings. For many images, a k much smaller than n can be used to reconstruct the image provided that a very low resolution version of the image is sufficient. Thus, one could use SVD as a compression method.

It turns out that in a more sophisticated approach, for certain classes of pictures one could use a fixed basis so that the top (say) hundred singular vectors are sufficient to represent any picture approximately. This means that the space spanned by the top hundred singular vectors is not too different from the space spanned by the top two hundred singular vectors of a given matrix in that class. Compressing these matrices by this standard basis can save substantially since the standard basis is transmitted only once and a matrix is transmitted by sending the top several hundred singular values for the standard basis.

1.5.5 Spectral Decomposition

Let B be a square matrix. If the vector $\mathbf{x}$ and scalar λ are such that $B\mathbf{x} = \lambda\mathbf{x}$, then $\mathbf{x}$ is an *eigenvector* of the matrix B and λ is the corresponding *eigenvalue*. We present here a spectral decomposition theorem for the special case where B is of the form

$B = AA^T$ for some (possibly rectangular) matrix A. If A is a real valued matrix, then B is symmetric and positive definite. That is, $\mathbf{x}^T B\mathbf{x} > 0$ for all nonzero vectors $\mathbf{x}$. The spectral decomposition theorem holds more generally and the interested reader should consult a linear algebra book.

Theorem 1.17 (Spectral Decomposition) *If $B = AA^T$ then $B = \sum_i \sigma_i^2 u_i u_i^T$ where $A = \sum_i \sigma_i u_i v_i^T$ is the singular valued decomposition of A.*

Proof:

$$\begin{aligned} B = AA^T &= \left(\sum_i \sigma_i \mathbf{u_i} \mathbf{v_i}^T\right)\left(\sum_j \sigma_j \mathbf{u_j} v_j^T\right)^T \\ &= \sum_i \sum_j \sigma_i \sigma_j \mathbf{u_i} \mathbf{v_i}^T \mathbf{v_j} \mathbf{u_j}^T \\ &= \sum_i \sigma_i^2 \mathbf{u_i} \mathbf{u_i}^T. \end{aligned}$$

■

When the σ_i are all distinct, the $\mathbf{u_i}$ are the eigenvectors of B and the σ_i^2 are the corresponding eigenvalues. If the σ_i are not distinct, then any vector that is a linear combination of those $\mathbf{u_i}$ with the same eigenvalue is an eigenvector of B.

1.5.6 Singular Vectors and ranking documents

An important task for a document collection is to *rank* the documents. Recall the term-document vector representation from Chapter 2. A näive method would be to rank in order of the total length of the document (which is the sum of the components of its term-document vector). Clearly, this is not a good measure in many cases. This näive method attaches equal weight to each term and takes the projection (dot product) term-document vector in the direction of the all 1 vector. Is there a better weighting of terms, i.e., a better projection direction which would measure the intrinsic relevance of the document to the collection? A good candidate is the best-fit direction for the collection of term-document vectors, namely the top (left) singular vector of the term-document matrix. An intuitive reason for this is that this direction has the maximum sum of squared projections of the collection and so can be thought of as a synthetic term-document vector best representing the document collection.

Ranking in order of the projection of each document's term vector along the best fit direction has a nice interpretation in terms of the power method. For this, we consider a different example - that of web with hypertext links. The World Wide Web can be represented by a directed graph whose nodes correspond to web pages and directed edges to hypertext links between pages. Some web pages, called *authorities*, are the most

prominent sources for information on a given topic. Other pages called *hubs*, are ones that identify the authorities on a topic. Authority pages are pointed to by many hub pages and hub pages point to many authorities. One is led to what seems like a circular definition: a hub is a page that points to many authorities and an authority is a page that is pointed to by many hubs.

One would like to assign hub weights and authority weights to each node of the web. If there are n nodes, the hub weights form a n-dimensional vector $\mathbf{u}$ and the authority weights form a n-dimensional vector $\mathbf{v}$. Suppose A is the adjacency matrix representing the directed graph : a_{ij} is 1 if there is a hypertext link from page i to page j and 0 otherwise. Given hub vector $\mathbf{u}$, the authority vector $\mathbf{v}$ could be computed by the formula

$$\mathbf{v}_j = \sum_{i=1}^{d} u_i a_{ij}$$

since the right hand side is the sum of the hub weights of all the nodes that point to node j. In matrix terms,

$$\mathbf{v} = A^T \mathbf{u}.$$

Similarly, given an authority vector $\mathbf{v}$, the hub vector $\mathbf{u}$ could be computed by $\mathbf{u} = A\mathbf{v}$. Of course, at the start, we have neither vector. But the above suggests a power iteration. Start with any $\mathbf{v}$. Set $\mathbf{u} = A\mathbf{v}$; then set $\mathbf{v} = A^T\mathbf{u}$ and repeat the process. We know from the power method that this converges to the left and right singular vectors. So after sufficiently many iterations, we may use the left vector $\mathbf{u}$ as hub weights vector and project each column of A onto this direction and rank columns (authorities) in order of this projection. But the projections just form the vector $A^T\mathbf{u}$ which equals $\mathbf{v}$. So we can just rank by order of v_j.

This is the basis of an algorithm called the HITS algorithm which was one of the early proposals for ranking web pages.

A different ranking called *page rank* is widely used. It is based on a random walk on the grap described above. (We will study Random Walks in detail in Chapter 5 and the reader may postpone reading this application until then.)

A random walk on the web goes from web page i to a randomly chosen neighbor of it. so if p_{ij} is the probability of going from i to j, then p_{ij} is just 1/ (number of hypertext links from i). Represent the p_{ij} in a matrix P. This matrix is called the *transition probability matrix* of the random walk. Represent the probabilities of being in each state at time t by the components of a row vector $\mathbf{p}(t)$. The probability of being in state j at time t is given by the equation

$$p_j(t) = \sum_i p_i(t-1)\, p_{ij}.$$

Then

$$\mathbf{p}(t) = \mathbf{p}(t-1)\, P$$

and thus

$$\mathbf{p}(t) = \mathbf{p}(0) P^t.$$

The probability vector $\mathbf{p}(t)$ is computed by computing P to the power t. It turns out that under some conditions, the random walk has a steady state probability vector that we can think of as $\mathbf{p}(\infty)$. It has turned out to be very useful to rank pages in decreasing order of $p_j(\infty)$ in essence saying that the web pages with the highest steady state probabilities are the most important.

In the above explanation, the random walk goes from page i to one of the web pages pointed to by i, picked uniformly at random. Modern technics for ranking pages are more complex. A more sophisticated random walk is used for several reasons. First, a web page might not contain any links and thus there is nowhere for the walk to go. Second, a page that has no in links will never be reached. Even if every node had at least one in link and one out link, the graph might not be strongly connected and the walk would eventually end up in some strongly connected component of the graph. Another difficulty occurs when the graph is periodic, that is, the greatest common divisor of all cycle lengths of the graph is greater than one. In this case, the random walk does not converge to a stationary probability distribution but rather oscillates between some set of probability distributions. We will consider this topic further in Chapter 5.

1.6 Bibliographic Notes

Singular value decomposition is fundamental to numerical analysis and linear algebra. There are many texts on these subjects and the interested reader may want to study these. A good reference is [**?**]. The material on clustering a mixture of Gaussians in Section 1.5.2 is from [**?**]. Modeling data with a mixture of Gaussians is a standard tool in statistics. Several well-known heuristics like the expectation-minimization algorithm are used to learn (fit) the mixture model to data. Recently, in theoretical computer science, there has been modest progress on provable polynomial-time algorithms for learning mixtures. Some references are [**?**], [**?**], [**?**], [**?**]. The application to the discrete optimization problem is from [**?**]. The section on ranking documents/webpages is from two influential papers, one on hubs and authorities by Jon Kleinberg [**?**] and the other on pagerank by Page, Brin, Motwani and Winograd [**?**].

1.7 Exercises

Exercise 1.1 (Best fit functions versus best least squares fit) *In many experiments one collects the value of a parameter at various instances of time. Let y_i be the value of the parameter y at time x_i. Suppose we wish to construct the best linear approximation to the data in the sense that we wish to minimize the mean square error. Here error is measured vertically rather than perpendicular to the line. Develop formulas for m and b to minimize the mean square error of the points $\{(x_i, y_i) | 1 \leq i \leq n\}$ to the line $y = mx + b$.*

Exercise 1.2 *Given five observed parameters, height, weight, age, income, and blood pressure of n people, how would one find the best least squares fit subspace of the form*

$$a_1 (height) + a_2 (weight) + a_3 (age) + a_4 (income) + a_5 (blood\ pressure) = 0$$

Here $a_1, a_2, \ldots, a_5$ are the unknown parameters. If there is a good best fit 4-dimensional subspace, then one can think of the points as lying close to a 4-dimensional sheet rather than points lying in 5-dimensions. Why is it better to use the perpendicular distance to the subspace rather than vertical distance where vertical distance to the subspace is measured along the coordinate axis corresponding to one of the unknowns?

Exercise 1.3 *What is the best fit line for each of the following set of points?*

1. $\{(0,1), (1,0)\}$
2. $\{(0,1), (2,0)\}$
3. *The rows of the matrix*

$$\begin{pmatrix} 17 & 4 \\ -2 & 26 \\ 11 & 7 \end{pmatrix}$$

Solution: (1) and (2) are easy to do from scratch. (1) $y = x$ and (2) $y = 2x$. For (3), there is no simple method. We will describe a general method later and this can be applied. But the best fit line is $v_1 = \frac{1}{\sqrt{5}} \begin{pmatrix} 1 \\ 2 \end{pmatrix}$. **FIX** Convince yourself that this is correct. ∎

Exercise 1.4 *Let A be a square $n \times n$ matrix whose rows are orthonormal. Prove that the columns of A are orthonormal.*

Solution: Since the rows of A are orthonormal $AA^T = I$ and hence $A^T AA^T = A^T$. Since A^T is nonsingular it has an inverse $\left(A^T\right)^{-1}$. Thus $A^T AA^T \left(A^T\right)^{-1} = A^T \left(A^T\right)^{-1}$ implying that $A^T A = I$, i.e., the columns of A are orthonormal. ∎

Exercise 1.5 *Suppose A is a $n \times n$ matrix with block diagonal structure with k equal size blocks where all entries of the i^{th} block are a_i with $a_1 > a_2 > \cdots > a_k > 0$. Show that A has exactly k nonzero singular vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}$ where $\mathbf{v_i}$ has the value $(\frac{k}{n})^{1/2}$ in the coordinates corresponding to the i^{th} block and 0 elsewhere. In other words, the singular vectors exactly identify the blocks of the diagonal. What happens if $a_1 = a_2 = \cdots = a_k$? In the case where the a_i are equal, what is the structure of the set of all possible singular vectors?*
Hint: *By symmetry, the top singular vector's components must be constant in each block.*

Exercise 1.6 *Prove that the left singular vectors of A are the right singular vectors of A^T.*

Solution: $A = UDV^T$, thus $A^T = VDU^T$. ∎

Exercise 1.7 *Interpret the right and left singular vectors for the document term matrix.*

Solution: The first right singular vector is a synthetic document that best matches the collection of documents. The first left singular vector is a synthetic word that best matches the collection of terms appearing in the documents. ∎

Exercise 1.8 *Verify that the sum of rank one matrices $\sum_{i=1}^{r} \sigma_i \mathbf{u_i} \mathbf{v_i}^T$ can be written as UDV^T, where the $\mathbf{u_i}$ are the columns of U and $\mathbf{v_i}$ are the columns of V. To do this, first verify that for any two matrices P and Q, we have*

$$PQ = \sum_i \mathbf{p_i} \mathbf{q_i}^T$$

where $\mathbf{p_i}$ is the i^{th} column of P and $\mathbf{q_i}$ is the i^{th} column of Q.

Exercise 1.9

1. *Show that the rank of A is r where r is the miminum i such that $\max_{\substack{\mathbf{v} \perp \mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v}_i \\ |\mathbf{v}|=1}} |A\,\mathbf{v}| = 0$.*
2. *Show that $\left|\mathbf{u_1}^T A\right| = \max_{|\mathbf{u}|=1} \left|\mathbf{u}^T A\right| = \sigma_1$.*

Hint: *Use SVD.*

Exercise 1.10 *If $\sigma_1, \sigma_2, \ldots, \sigma_r$ are the singular values of A and $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r$ are the corresponding right singular vectors, show that*

1. $A^T A = \sum_{i=1}^{r} \sigma_i^2 \mathbf{v_i} \mathbf{v_i}^T$

2. $\mathbf{v}_1, \mathbf{v}_2, \ldots \mathbf{v}_r$ *are eigenvectors of*$A^T A$.

3. *Assuming that the set of eigenvectors of a matrix is unique, conclude that the set of singular values of the matrix is unique.*

See the appendix for the definition of eigenvectors.

Exercise 1.11 *Let A be a matrix. Given an algorithm for finding*

$$\mathbf{v_1} = \arg\max_{|\mathbf{v}|=1} |A\mathbf{v}|$$

describe an algorithm to find the SVD of A.

Exercise 1.12 *Compute the singular valued decomposition of the matrix*

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Exercise 1.13 *Write a program to implement the power method for computing the first singular vector of a matrix. Apply your program to the matrix*

$$A = \begin{pmatrix} 1 & 2 & 3 & \cdots & 9 & 10 \\ 2 & 3 & 4 & \cdots & 10 & 0 \\ \vdots & \vdots & \vdots & & & \vdots \\ 9 & 10 & 0 & \cdots & 0 & 0 \\ 10 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

Exercise 1.14 *Modify the power method to find the first four singular vectors of a matrix A as follows. Randomly select four vectors and find an orthonormal basis for the space spanned by the four vectors. Then multiple each of the basis vectors times A and find a new orthonormal basis for the space spanned by the resulting four vectors. Apply your method to find the first four singular vectors of matrix A of Exercise 1.13*

Exercise 1.15 *Let A be a real valued matrix. Prove that $B = AA^T$ is positive definite.*

Exercise 1.16 *Prove that the eigenvalues of a symmetric real valued matrix are real.*

Exercise 1.17 *Suppose A is a square invertible matrix and the SVD of A is $A = \sum_i \sigma_i u_i v_i^T$. Prove that the inverse of A is $\sum_i \frac{1}{\sigma_i} v_i u_i^T$.*

Exercise 1.18 *Suppose A is square, but not necessarily invertible and has SVD $A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$. Let $B = \sum_{i=1}^{r} \frac{1}{\sigma_i} v_i u_i^T$. Show that $B\mathbf{x} = \mathbf{x}$ for all $\mathbf{x}$ in the span of the right singular vectors of A. For this reason B is sometimes called the pseudo inverse of A and can play the role of A^{-1} in many applications.*

Exercise 1.19

1. *For any matrix A, show that $\sigma_k \leq \frac{||A||_F}{\sqrt{k}}$.*

2. *Prove that there exists a matrix B of rank at most k such that $||A - B||_2 \leq \frac{||A||_F}{\sqrt{k}}$.*

3. *Can the 2-norm on the left hand side in (b) be replaced by Frobenius norm?*

Exercise 1.20 *Suppose an $n \times d$ matrix A is given and you are allowed to preprocess A. Then you are given a number of d-dimensional vectors $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}$ and for each of these vectors you must find the vector $A\mathbf{x_i}$ approximately, in the sense that you must find a vector $\mathbf{u_i}$ satisfying $|\mathbf{u_i} - A\mathbf{x_i}| \leq \varepsilon ||A||_F |\mathbf{x_i}|$. Here $\varepsilon > 0$ is a given error bound. Describe an algorithm that accomplishes this in time $O\left(\frac{d+n}{\varepsilon^2}\right)$ per $\mathbf{x_i}$ not counting the preprocessing time.*

Exercise 1.21 ***(Constrained Least Squares Problem using SVD)*** *Given A, $\mathbf{b}$, and m, use the SVD algorithm to find a vector $\mathbf{x}$ with $|\mathbf{x}| < m$ minimizing $|A\mathbf{x} - \mathbf{b}|$. This problem is a learning exercise for the advanced student. For hints/solution consult Golub and van Loan, Chapter 12.*

Exercise 1.22 (Document-Term Matrices): *Suppose we have a $m \times n$ document-term matrix where each row corresponds to a document where the rows have been normalized to length one. Define the "similarity" between two such documents by their dot product.*

1. *Consider a "synthetic" document whose sum of squared similarities with all documents in the matrix is as high as possible. What is this synthetic document and how would you find it?*

2. *How does the synthetic document in (1) differ from the center of gravity?*

3. *Building on (1), given a positive integer k, find a set of k synthetic documents such that the sum of squares of the mk similarities between each document in the matrix and each synthetic document is maximized. To avoid the trivial solution of selecting k copies of the document in (1), require the k synthetic documents to be orthogonal to each other. Relate these synthetic documents to singular vectors.*

4. *Suppose that the documents can be partitioned into k subsets (often called clusters), where documents in the same cluster are similar and documents in different clusters are not very similar. Consider the computational problem of isolating the clusters. This is a hard problem in general. But assume that the terms can also be partitioned into k clusters so that for $i \neq j$, no term in the i^{th} cluster occurs in a document in the j^{th} cluster. If we knew the clusters and arranged the rows and columns in them to be contiguous, then the matrix would be a block-diagonal matrix. Of course*

the clusters are not known. By a "block" of the document-term matrix, we mean a submatrix with rows corresponding to the i^{th} cluster of documents and columns corresponding to the i^{th} cluster of terms . We can also partition any n vector into blocks. Show that any right singular vector of the matrix must have the property that each of its blocks is a right singular vector of the corresponding block of the document-term matrix.

5. *Suppose now that the singular values of all the blocks are distinct (also across blocks). Show how to solve the clustering problem.*

Hint: *(4) Use the fact that the right singular vectors must be eigenvectors of A^TA. Show that A^TA is also block-diagonal and use properties of eigenvectors.*

Solution: (1)
(2)
(3): It is obvious that A^TA is block diagonal. We claim that for any block-diagonal symmetric matrix B, each eigenvector must be composed of eigenvectors of blocks. To see this, just note that since for an eigenvector $\mathbf{v}$ of B, $B\mathbf{v}$ is $\lambda\mathbf{v}$ for a real λ, for a block B_i of B, $B_i\mathbf{v}$ is also λ times the corresponding block of $\mathbf{v}$.
(4): By the above, it is easy to see that each eigenvector of A^TA has nonzero entries in just one block.
(e) ∎

Exercise 1.23 *Generate a number of samples according to a mixture of 1-dimensional Gaussians. See what happens as the centers get closer. Alternatively, see what happens when the centers are fixed and the standard deviation is increased.*

Exercise 1.24 *Show that maximizing $\mathbf{x}^T\mathbf{u}\mathbf{u}^T(\mathbf{1}-\mathbf{x})$ subject to $x_i \in \{0,1\}$ is equivalent to partitioning the coordinates of $\mathbf{u}$ into two subsets where the sum of the elements in both subsets are equal.*

Solution: $\mathbf{x}^T\mathbf{u}\mathbf{u}^T(\mathbf{1}-\mathbf{x})$ can be written as the product of two scalars $\left(\mathbf{x}^T\mathbf{u}\right)\left(\mathbf{u}^T(\mathbf{1}-\mathbf{x})\right)$. The first scalar is the sum of the coordinates of $\mathbf{u}$ corresponding to the subset S and the second scalar is the sum of the complementary coordinates of $\mathbf{u}$. To maximize the product, one partitions the coordinates of $\mathbf{u}$ so that the two sums are as equally as possible. Given the subset determined by the maximization, check if $\mathbf{x}^T\mathbf{u} = \mathbf{u}^T(\mathbf{1}-\mathbf{x})$. ∎

Exercise 1.25 *Read in a photo and convert to a matrix. Perform a singular value decomposition of the matrix. Reconstruct the photo using only 10%, 25%, 50% of the singular values.*

1. *Print the reconstructed photo. How good is the quality of the reconstructed photo?*

2. *What percent of the Forbenius norm is captured in each case?*

Hint: *If you use Matlab, the command to read a photo is imread. The types of files that can be read are given by imformats. To print the file use imwrite. Print using jpeg format. To access the file afterwards you may need to add the file extension .jpg. The command imread will read the file in uint8 and you will need to convert to double for the SVD code. Afterwards you will need to convert back to uint8 to write the file. If the photo is a color photo you will get three matrices for the three colors used.*

Exercise 1.26 *Find a collection of something such as photgraphs, drawings, or charts and try the SVD compression technique on it. How well does the reconstruction work?*

Exercise 1.27 *Create a set of 100,* 100×100 *matrices of random numbers between 0 and 1 such that each entry is highly correlated with the adjacency entries. Find the SVD of A. What fraction of the Frobenius norm of A is captured by the top 100 singular vectors? How many singular vectors are required to capture 95% of the Frobenius norm?*

Exercise 1.28 *Create a* 100×100 *matrix A of random numbers between 0 and 1 such that each entry is highly correlated with the adjacency entries and find the first 100 vectors for a single basis that is reasonably good for all 100 matrices. How does one do this? What fraction of the Frobenius norm of a new matrix is captured by the basis?*

Solution: If $\mathbf{v_1}, \mathbf{v_2}, \cdots, \mathbf{v_{100}}$ is the basis, then $A = A\mathbf{v_1}\mathbf{v_1}^T + A\mathbf{v_2}\mathbf{v_2}^T + \cdots$. ∎

Exercise 1.29 *Show that the running time for the maximum cut algorithm in Section* **??** *can be carried out in time* $O(n^3 + poly(n)k^k)$*, where poly is some polynomial.*

Exercise 1.30 *Let* $x_1, x_2, \ldots, x_n$ *be n points in d-dimensional space and let X be the* $n \times d$ *matrix whose rows are the n points. Suppose we know only the matrix D of pairwise distances between points and not the coordinates of the points themselves. The* x_{ij} *are not unique since any translation, rotation, or reflection of the coordinate system leaves the distances invariant. Fix the origin of the coordinate system so that the centroid of the set of points is at the origin.*

1. *Show that the elements of* $X^T X$ *are given by*

$$x_i^T x_j = -\frac{1}{2}\left[d_{ij}^2 - \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 + \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2\right].$$

2. *Describe an algorithm for determining the matrix X whose rows are the* x_i*.*

Solution: (1) Since the centroid of the set of points is at the origin of the coordinate axes, $\sum_{i=1}^{n} x_{ij} = 0$. Write

$$d_{ij}^2 = (x_i - x_j)^T(x_i - x_j) = x_i^T x_i + x_j^T x_j - 2x_i^T x_j \tag{1.3}$$

Then

$$\frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 = \frac{1}{n}\sum_{i=1}^{n} x_i^T x_i + x_j^T x_j \tag{1.4}$$

Since $\frac{1}{n}\sum_{i=1}^{n} x_j^T x_j = x_j^T x_j$ and $\frac{1}{n}\left(\sum_{i=1}^{n} x_i^T\right) x_j = 0$.
Similarly

$$\frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 = \frac{1}{n}\sum_{j=1}^{n} x_j^T x_j + x_i^T x_i \tag{1.5}$$

Summing (1.4) over j gives

$$\frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{n} d_{ij}^2 = \sum_{i=1}^{n} x_i^T x_i + \sum_{j=1}^{n} x_j^T x_j = 2\sum_{i=1}^{n} x_i^T x_i \tag{1.6}$$

Rearranging (1.3) and substituting for $x_i^T x_i$ and $x_j^T x_j$ from (1.3) and (1.4) yields

$$x_i^T x_j = -\frac{1}{2}\left(d_{ij}^2 - x_i^T x_i - x_j^T x_j\right) = -\frac{1}{2}\left(d_{ij}^2 - \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 + \frac{2}{n}\sum_{i=1}^{n} x_i^T x_i\right)$$

Finally substituting (1.6) yields

$$x_i^T x_j = -\frac{1}{2}\left(d_{ij}^2 - x_i^T x_i - x_j^T x_j\right) = -\frac{1}{2}\left(d_{ij}^2 - \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 - \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 + \frac{1}{n^2}\sum_{j=1}^{n}\sum_{i=1}^{n} d_{ij}^2\right)$$

Note that is D is the matrix of pairwise squared distances, then $\frac{1}{n}\sum_{k=1}^{n} d_{ij}^2$, $\frac{1}{n}\sum_{i=1}^{n} d_{ij}^2$, and $\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2$ are the averages of the square of the elements of the i^{th} row, the square of the elements of the j^{th} column and all squared distances respectively.
(2) Having constructed $X^T X$ we can use an eigenvalue decomposition to determine the coordinate matrix X. Clearly $X^T X$ is symmetric and if the distances come from a set of n points in a d-dimensional space $X^T X$ will be positive definite and of rank d. Thus we can decompose $X^T X$ as$X^T X = V^T \sigma V$ where the first d eigenvalues are positive and the remainder are zero. Since the $X^T X = V^T \sigma^{\frac{1}{2}} \sigma^{\frac{1}{2}} V$ and thus the coordinates are given by $X = V^T \sigma^{\frac{1}{2}}$

Exercise 1.31

1. *Consider the pairwise distance matrix for twenty US cities given below. Use the algorithm of Exercise 2 to place the cities on a map of the US.*

2. *Suppose you had airline distances for 50 cities around the world. Could you use these distances to construct a world map?*

	BOS	BUF	CHI	DAL	DEN	HOU	LA	MEM	MIA	MIM
Boston	-	400	851	1551	1769	1605	2596	1137	1255	1123
Buffalo	400	-	454	1198	1370	1286	2198	803	1181	731
Chicago	851	454	-	803	920	940	1745	482	1188	355
Dallas	1551	1198	803	-	663	225	1240	420	1111	862
Denver	1769	1370	920	663	-	879	831	879	1726	700
Houston	1605	1286	940	225	879	-	1374	484	968	1056
Los Angeles	2596	2198	1745	1240	831	1374	-	1603	2339	1524
Memphis	1137	803	482	420	879	484	1603	-	872	699
Miami	1255	1181	1188	1111	1726	968	2339	872	-	1511
Minneapolis	1123	731	355	862	700	1056	1524	699	1511	-
New York	188	292	713	1374	1631	1420	2451	957	1092	1018
Omaha	1282	883	432	586	488	794	1315	529	1397	290
Philadelphia	271	279	666	1299	1579	1341	2394	881	1019	985
Phoenix	2300	1906	1453	887	586	1017	357	1263	1982	1280
Pittsburgh	483	178	410	1070	1320	1137	2136	660	1010	743
Saint Louis	1038	662	262	547	796	679	1589	240	1061	466
Salt Lake City	2099	1699	1260	999	371	1200	579	1250	2089	987
San Francisco	2699	2300	1858	1483	949	1645	347	1802	2594	1584
Seattle	2493	2117	1737	1681	1021	1891	959	1867	2734	1395
Washington D.C.	393	292	597	1185	1494	1220	2300	765	923	934

	NY	OMA	PHI	PHO	PIT	Stl	SLC	SF	SEA	DC
Boston	188	1282	271	2300	483	1038	2099	2699	2493	393
Buffalo	292	883	279	1906	178	662	1699	2300	2117	292
Chicago	713	432	666	1453	410	262	1260	1858	1737	597
Dallas	1374	586	1299	887	1070	547	999	1483	1681	1185
Denver	1631	488	1579	586	1320	796	371	949	1021	1494
Houston	1420	794	1341	1017	1137	679	1200	1645	1891	1220
Los Angeles	2451	1315	2394	357	2136	1589	579	347	959	2300
Memphis	957	529	881	1263	660	240	1250	1802	1867	765
Miami	1092	1397	1019	1982	1010	1061	2089	2594	2734	923
Minneapolis	1018	290	985	1280	743	466	987	1584	1395	934
New York	-	1144	83	2145	317	875	1972	2571	2408	205
Omaha	1144	-	1094	1036	836	354	833	1429	1369	1014
Philadelphia	83	1094	-	2083	259	811	1925	2523	2380	123
Phoenix	2145	1036	2083	-	1828	1272	504	653	1114	1963
Pittsburgh	317	836	259	1828	-	559	1668	2264	2138	192
Saint Louis	875	354	811	1272	559	-	1162	1744	1724	712
Salt Lake City	1972	833	1925	504	1668	1162	-	600	701	1848
San Francisco	2571	1429	2523	653	2264	1744	600	-	678	2442
Seattle	2408	1369	2380	1114	2138	1724	701	678	-	2329
Washington D.C.	250	1014	123	1983	192	712	1848	2442	2329	-

References

Hubs and Authorities
Golub and Van Loan
Clustering Gaussians