An Introduction to Digital Video Data Compression in Java

Fore June

Chapter 15 Principal Component Analysis

15.1 Introduction

We have mentioned in Chapter 14 that people use the techniques of Principal Component Analysis (PCA) to model face features. Actually, PCA is a common statistical technique used in finding patterns in high-dimension data. It has applications in various fields such as image processing and face recognition. Before discussing PCA, we need to have a basic knowledge about the tools and techniques involved in PCA. Therefore, this chapter first introduces the mathematical tools that will be used in PCA, starting from a discussion on matrix operations. It then discusses standard deviation, covariance, eigenvectors and eigenvalues. If you are already familiar with these basic tools and concepts, you can skip them and go directly to the PCA section.

Some of the examples of this chapter are adopted from the college textbook *Linear Algebra with Applications* by *Steven J. Leon*.

15.2 Matrix Algebra

A matrix is a rectangular array of numbers. An $m \times n$ matrix, read "m by n matrix", has m rows and n columns. Let A be an $m \times n$ matrix with a_{ij} denoting the element in the ith row and jth column. Then

$$A = (a_{ij}) = \begin{pmatrix} a_{11} & a_{12} & ... & a_{1n} \\ a_{21} & a_{22} & ... & a_{2n} \\ . & . & ... & . \\ a_{m1} & a_{m2} & ... & a_{mn} \end{pmatrix} \tag{15.1}$$

The elements $a_{11}, a_{22}, a_{33}...$ are called the elements of the *main diagonal* of A. Two matrices $A = (a_{ij})$ and $B = (b_{ij})$ are equal if they have the same number of rows and columns and the elements are equal, i.e. $a_{ij} = b_{ij}$. The following are examples of a 3×1 matrix, a 2×3 matrix, and a 3×3 matrix respectively.

$$\begin{pmatrix} 3.1 \\ 4.1 \\ 5.9 \end{pmatrix}, \quad \begin{pmatrix} 1 & -2 & 3 \\ 9 & 8 & 5 \end{pmatrix}, \quad \begin{pmatrix} 3 & 1 & 4 \\ 1 & 5 & 9 \\ 2 & 6 & 5 \end{pmatrix}$$

The transpose of A in (15.1) is obtained by interchanging rows and columns, and is denoted by A^T. That is,

$$A^T = \begin{pmatrix} a_{11} & a_{21} & ... & a_{m1} \\ a_{12} & a_{22} & ... & a_{m2} \\ . & . & ... & . \\ a_{1n} & a_{2n} & ... & a_{mn} \end{pmatrix} \tag{15.2}$$

and A^T has n rows and m columns. For example,

$$\begin{pmatrix} 1 & -2 & 3 \\ 9 & 8 & 5 \end{pmatrix}^T = \begin{pmatrix} 1 & 9 \\ -2 & 8 \\ 3 & 5 \end{pmatrix}$$

Maxtrix $X = (x_1, x_2, ..., x_n)$ is a $1 \times n$ matrix; it has one row and n columns, and we call it an nth order row vector. Its transpose

$$X^T = \begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{pmatrix}$$

is an $n \times 1$ matrix, consisting of n rows and one column, and will be called a column vector in contrast to X itself.

We note that, if A is any matrix and A^T its transpose, then A is the transpose of A^T so that $(A^T)^T = A$. We call a matrix that has the same number of rows as columns a *square matrix.* The transpose of a square matrix is also a square matrix. A square matrix with n rows and n columns is referred to as a matrix of *order* n. If the transpose of a square matrix is equal to itself (i.e. $A^T = A$), then it is called a *symmetric* matrix. An important property of symmetric matrix is that we can diagonalize any symmetric matrix A, which means that A is similar to a diagonal matrix. That is, there exists an invertible matrix P such that $P^{-1}AP$ is a diagonal matrix, where P^{-1} is the inverse of P.

The *row rank* of a matrix A is the maximum number of linearly independent row vectors of A and the *column rank* is the maximum number of linearly independent column vectors. A matrix A has full column rank if its column vectors are independet and it has full row rank if its row vectors are independent. One can show that the row rank is always equal to the column rank of a matrix. Therefore, we define that the *rank* of a matrix A is the maximum number of linearly independent row (or column) vectors of A and is denoted by $rank(A)$. The rank of an $m \times n$ matrix cannot be greater than m nor n. That is $rank(A) \leq min(m, n)$. A matrix that has a rank as large as possible is said to have *full rank*, otherwise the matrix is *rank deficient.* If A is a square matrix (i.e., $m = n$), then A is invertible (i.e., its inverse exists) if and only if A has rank n (i.e., A has full rank).

An identity matrix I is an $n \times n$ square matrix with $I = (\delta_{ij})$, where

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

is the Kronecker delta. An identity matrix has 1's on the main diagonal and 0's elsewhere.

Matrix Multiplication

If A is an $m \times n$ matrix and B is an $n \times r$ matrix, we can form the product of A and B, which is an $m \times r$ matrix; if C is the product of A and B, we write $C = AB$ and

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} \qquad 1 \leq i \leq m, 1 \leq j \leq r \tag{15.3}$$

One can easily show that matrix multiplication is *associative*. That is, if A, B, C are any three matrices of types (m, n), (n, r) and (r, s), respectively, then

$$(AB)C = A(BC) \tag{15.4}$$

One can also show that the transpose of the product of two matrices A and B is equal to the product of their transposes in *reverse order*. That is,

$$(AB)^T = B^T A^T \tag{15.5}$$

For any $n \times n$ matrix A,

$$IA = AI = A$$

Matrix Addition

If two matrices A and B are of the same type (m, n), their sum $C = A + B$ is obtained by adding corresponding elements of A and B:

$$c_{ij} = a_{ij} + b_{ij} \tag{15.6}$$

Clearly, addition of matrices is commutative, $A+B = B+A$, and associative, $(A+B)+C = A + (B + C)$. Moreover, the zero matrix O satisfies the law $A + O = A + O = A$ for every matrix A. For every matrix $A = (a_{ij})$, there corresponds a *negative* $-A = (-a_{ij})$ with property $A + (-A) = O$.

One can also easily prove that matrix multiplication is distributive with respect to matrix addition. That is, if matrices A and B are of type (m, n), C of type (r, m) and D of type (n, s), then

$$C(A + B) = CA + CB \tag{15.7}$$

and

$$(A + B)D = AD + BD \tag{15.8}$$

Determinant

The determinant of a square matrix is a value associated with the matrix and is computed from the entries of the matrix. It provides valuable information for the matrix operations such as computing the inverse. The determinant of a matrix A is denoted det (A) or $|A|$.

Let $A = (a_{ij})$ be a square matrix of order n. We can form a product of elements of A by multiplying together one and only one element from each row and each column, which will be in the form,

$$a_{1i_1} a_{2i_2} \cdots a_{ni_n} \tag{15.9}$$

where $i_1, i_2, \cdots, i_n$ is a permutation of the numbers $1, 2, \cdots, n$. The numbering and permutation ensure that one and only one element is chosen from each row and each column. The determinant of A is the sum of some of products in the form of (15.9). We say that an inversion occurs in the permutation $i_1, i_2, \cdots, i_n$ whenever a larger subscript precedes a smaller one. For example, consider $n = 4$; the product

$$a_{14}a_{22}a_{31}a_{43}$$

has a total of 4 inversions because $i_1i_2i_3i_4 = 4213$, so 4 precedes $1, 2$, and 3 (3 inversions), and 2 precedes 1 (1 inversion); the product

$$a_{11}a_{22}a_{33}a_{44}$$

has zero inversion. We say that a permutation $i_1i_2\cdots i_n$ of the numbers $1, 2, \cdots, n$ is even if the number of inversions is even and it is odd if the number of inversions is odd. We can now define the determinant of a square matrix.

We denote the determinant associated with the square matrix A by $|A|$ or by

$$\begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ . & . & \dots & . \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{vmatrix} \tag{15.10}$$

The determinant is a polynomial of the elements of A defined by

$$|A| = \sum \pm a_{1i_1}a_{2i_2}\cdots a_{ni_n} \tag{15.11}$$

where we sum over all $n!$ permutations $i_1, i_2, \cdots, i_n$ of $1, 2, \cdots, n$; the sign before a term is $+$ if the permutation is even, and $-$ for odd permutations. One can prove that the determinant $|A|$ of an $n \times n$ square matrix A has the following properties.

1. $|A^T| = |A|$.
2. If the elements of two rows (or two columns) are identical, $|A| = 0$.
3. If matrix B is obtained by interchanging two rows or two columns of A, then $|B| = -|A|$.
4. If matrix B is obtained by multiplying all the elements of a row or a column of A by a constant c, then $|B| = c|A|$.
5. If c is a constant, then $|cA| = c^n|A|$.
6. If B is an $n \times n$ matrix, then $|AB| = |A||B|$.
7. If all the elements of a row or a column of A are 0, then $|A| = 0$.
8. If matrix B is obtained by multiplying a row (or column) vector of $|A|$ by a constant c and adding the result to another row (or column) vector, then $|B| = |A|$.
9. If I is the identity matrix, then $1 = |I| = |AA^{-1}| = |A||A^{-1}|$, where A^{-1} is the the inverse of A, which is discussed below. Therefore

$$|A^{-1}| = \frac{1}{|A|}$$

10. The determinant of the similarity transformation of A is equal to $|A|$:

$$|BAB^{-1}| = |B||A||B^{-1}| = |B||A|\frac{1}{|B|} = |A|$$

For example, a 2×2 matrix can be calculated as,

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc \tag{15.12}$$

The calculation can be represented graphically as shown in Figure 15-1 below, where an arrow crosses the elements of a diagonal of the matrix, giving rise to a term of (15.11). We put a '$-$' sign in front of the term if the arrow points upward, and a '$+$' sign for the arrow pointing downward. The determinant is given by the sum of the terms.

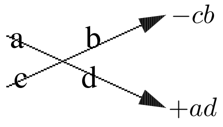

Figure 15-1 Calculating 2×2 Matrix Determinant

We can similarly use this graphical method to calculate the determinant of a 3×3 matrix $A = (a_{ij})$. In this case, we copy the first two rows of the matrix and put them beneath the last row. We then cross out the diagonal elements as shown in Figure 15-2. Again, a '$-$' sign is added for an 'upward' term and a '$+$' sign is added for a 'downward' term. The determinant is the sum of all the terms thus obtained.

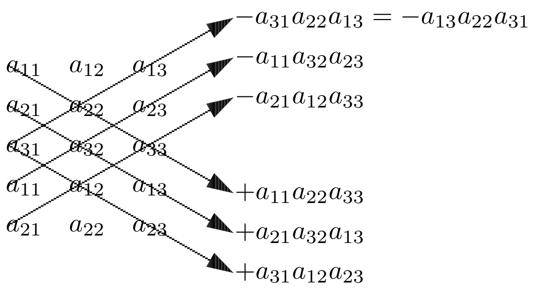
$-a_{31}a_{22}a_{13} = -a_{13}a_{22}a_{31}$
$-a_{11}a_{32}a_{23}$
$-a_{21}a_{12}a_{33}$
$+a_{11}a_{22}a_{33}$
$+a_{21}a_{32}a_{13}$
$+a_{31}a_{12}a_{23}$
a_{11} a_{12} a_{13}
a_{21} a_{22} a_{23}
a_{31} a_{32} a_{33}
a_{11} a_{12} a_{13}
a_{21} a_{22} a_{23}

Figure 15-2 Calculating 3×3 Matrix Determinant

If we delete some rows and/or columns of a matrix A, the matrix of the remaining elements is referred to as a *submatrix* of A. In particular, if A is a square matrix, and we delete the i-th row and j-th column of it, then we denote the remaining submatrix by A_{ij}. The determinant $|A_{ij}|$ is called the *minor* of of the element a_{ij} in A, and $(-1)^{i+j}|A_{ij}|$ is called the *signed minor* or *cofactor* of a_{ij} in A. With these notations, one can express the determinant of an $n \times n$ matrix A as an expansion of determinants of minors as follows:

$$\begin{aligned} |A| &= (-1)^{i+1}a_{i1}|A_{i1}| + (-1)^{i+2}a_{i2}|A_{i2}| + \cdots + (-1)^{i+n}a_{in}|A_{in}| \\ &= (-1)^{1+j}a_{1j}|A_{1j}| + (-1)^{2+j}a_{2j}|A_{2j}| + \cdots + (-1)^{n+j}a_{nj}|A_{nj}| \end{aligned} \qquad (15.13)$$

where $i, j = 1, 2, \cdots, n$. For example, we can express the expansion along the first row as

$$\begin{vmatrix} a_{11} & a_{12} & \cdot\cdot & a_{1n} \\ a_{21} & a_{22} & \cdot\cdot & a_{2n} \\ \cdot & \cdot & \cdot\cdot & \cdot \\ a_{n1} & a_{n2} & \cdot\cdot & a_{nn} \end{vmatrix} = a_{11} \begin{vmatrix} a_{22} & a_{23} & \cdot\cdot & a_{2n} \\ \cdot & \cdot & \cdot\cdot & \cdot \\ a_{n2} & a_{n3} & \cdot\cdot & a_{nn} \end{vmatrix} - a_{12} \begin{vmatrix} a_{21} & a_{23} & \cdot\cdot & a_{2n} \\ \cdot & \cdot & \cdot\cdot & \cdot \\ a_{n1} & a_{n3} & \cdot\cdot & a_{nn} \end{vmatrix}$$
$$+ \cdots + (-1)^{1+n} \begin{vmatrix} a_{21} & a_{22} & \cdot\cdot & a_{2(n-1)} \\ \cdot & \cdot & \cdot\cdot & \cdot \\ a_{n1} & a_{n3} & \cdot\cdot & a_{n(n-1)} \end{vmatrix} \tag{15.14}$$

In general, we can express the determinant of A as

$$\boxed{|A| = \sum_{i=1}^{n} a_{ij} C_{ij}} \tag{15.15}$$

where C_{ij} is the cofactor of a_{ij}, which is

$$C_{ij} = (-1)^{i+j} |A_{ij}| \tag{15.16}$$

and A_{ij} is the submatrix formed by deleting row i and column j from A. One can also show that if $h \neq k$, then

$$\sum_{j=1}^{n} a_{hj} C_{kj} = 0 \quad \text{and} \quad \sum_{i=1}^{n} a_{ih} C_{ik} = 0 \tag{15.17}$$

The left equation says that the sum of the elements from the h-th row times the cofactors from the k-th row is zero. The right equation is about columns.

We can combine equations (15.15) and (15.17) in the form

$$\sum_{j=1}^{n} a_{hj} C_{kj} = |A| \delta_{hk} \quad \text{and} \quad \sum_{i=1}^{n} a_{ih} C_{ik} = |A| \delta_{hk} \tag{15.18}$$

The following is an example of evaluating the determinant of a 3×3 matrix.

$$\begin{aligned} \begin{vmatrix} 2 & -1 & 6 \\ 4 & 1 & 2 \\ 3 & 5 & 7 \end{vmatrix} &= 2 \times \begin{vmatrix} 1 & 2 \\ 5 & 7 \end{vmatrix} - 4 \times \begin{vmatrix} -1 & 6 \\ 5 & 7 \end{vmatrix} + 3 \times \begin{vmatrix} -1 & 6 \\ 1 & 2 \end{vmatrix} \\ &= 2(1 \times 7 - 5 \times 2) - 4(-1 \times 7 - 5 \times 6) + 3(-1 \times 2 - 1 \times 6) \\ &= -6 + 148 - 24 \\ &= 118 \end{aligned}$$

Matrix Inverse

We denote $I = (\delta_{ij})$ as an $n \times n$ identity matrix, where δ_{ij} is a Kronecker delta. That is,

$$I = \begin{pmatrix} 1 & 0 & ... & 0 \\ 0 & 1 & 0.. & 0 \\ . & . & ... & . \\ 0 & 0 & ..0 & 1 \end{pmatrix} \tag{15.19}$$

We define the inverse of an $n \times n$ matrix A as a square matrix A^{-1} such that

$$AA^{-1} = I \tag{15.20}$$

A square matrix A has an inverse iff its determinant $|A| \neq 0$. We say that a matrix is *nonsingular* or *invertible* if its inverse exists.

We can obtain the inverse of A from the *adjoint* (matrix) of A, denoted as A^{adj}, which is defined as the transpose of the cofactor matrix C_{ij}. That is,

$$A^{adj} = (C_{ij})^T = \begin{pmatrix} C_{11} & C_{21} & ... & C_{n1} \\ C_{12} & C_{22} & .. & C_{n2} \\ . & . & ... & . \\ C_{1n} & C_{2n} & .. & C_{nn} \end{pmatrix} \tag{15.21}$$

From (15.18), we have

$$AA^{adj} = |A|I \tag{15.22}$$

Therefore, if $|A| \neq 0$, the inverse of A is given by

$$\boxed{A^{-1} = \frac{1}{|A|}A^{adj}} \tag{15.23}$$

For a 2×2 matrix

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \tag{15.24}$$

its adjoint is

$$A^{adj} = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \tag{15.25}$$

So the inverse is

$$\begin{aligned} A^{-1} &= \frac{1}{|A|}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \\ &= \frac{1}{ad - bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \end{aligned} \tag{15.26}$$

As an example, let us find the inverse of the 3×3 matrix

$$A = \begin{pmatrix} 2 & 1 & 3 \\ 2 & 4 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

In this example, $|A| = 6$, and

$$C_{11} = \begin{vmatrix} 4 & 0 \\ 2 & 1 \end{vmatrix} = 4, \quad C_{12} = -\begin{vmatrix} 2 & 0 \\ 1 & 1 \end{vmatrix} = -2, \quad C_{13} = \begin{vmatrix} 2 & 4 \\ 1 & 2 \end{vmatrix} = 0, \quad \cdots$$

and so on. So we have

$$A^{-1} = \frac{1}{|A|} A^{adj} = \frac{1}{6} \begin{pmatrix} 4 & 5 & -12 \\ -2 & -1 & 6 \\ 0 & -3 & 6 \end{pmatrix} = \begin{pmatrix} \frac{2}{3} & \frac{5}{6} & -2 \\ -\frac{1}{3} & -\frac{1}{6} & 1 \\ 0 & -\frac{1}{2} & 1 \end{pmatrix}$$

The following are some properties of matrix inverses:

1. If A and B are $n \times n$ matrices, we say that B is **similar** to A if there exists a nonsingular matrix S such that $B = S^{-1}AS$. Obviously, if B is similar to A, then A is similar to B as $A = U^{-1}BU$, where $U = S^{-1}$.

2. If matrices A and B are invertible, then their product AB is also invertible and is

$$(AB)^{-1} = B^{-1}A^{-1}$$

3. A diagonal matrix has an inverse if no diagonal element is zero:

$$\text{If } \Lambda = \begin{pmatrix} \lambda_1 & 0 & \cdot\cdot & 0 \\ 0 & \lambda_2 & \cdot\cdot & 0 \\ \cdot & \cdot\cdot & \cdot & \cdot \\ 0 & \cdot\cdot & 0 & \lambda_n \end{pmatrix} \text{ then } \Lambda^{-1} = \begin{pmatrix} \frac{1}{\lambda_1} & 0 & \cdot\cdot & 0 \\ 0 & \frac{1}{\lambda_2} & \cdot\cdot & 0 \\ \cdot & \cdot\cdot & \cdot & \cdot \\ 0 & \cdot\cdot & 0 & \frac{1}{\lambda_n} \end{pmatrix}$$

where $\lambda_i \neq 0$.

4. If A is an $n \times m$ matrix, then AA^T is $n \times n$ and is a symmetric square matrix.

5. An **orthogonal** matrix is a square matrix with orthogonal unit column and row vectors $\mathbf{v_i}$ (i.e., orthonormal vectors where $\mathbf{v_i} \cdot \mathbf{v_j} = \delta_{ij}$). One can show that a matrix A is orthogonal if and only if its transpose is equal to its inverse:

$$A^T = A^{-1}$$

or

$$AA^T = A^TA = I$$

Left inverse

If A is a nonsingular square matrix, it has a *2-sided inverse*, A^{-1} for which $AA^{-1} = I = A^{-1}A$. This is what we have called the *inverse* of A. If A is $m \times n$ and $m \neq n$, then it does not have a 2-sided inverse, but it can have a left inverse or a right inverse which are referred to as *pseudoinverse*.

If A is $m \times n$ with full column rank (i.e. $r = rank(A) = n, m > n$), the matrix A^TA is an invertible $n \times n$ symmetric matrix. Therefore, $(A^TA)^{-1}A^TA = I$. We can define the *left inverse* of A to be

$$A_{left}^{-1} = (A^TA)^{-1}A^T$$

as $A_{left}^{-1}A = I$.

Note that AA_{left}^{-1} is an $m \times m$ matrix which equals the identity matrix only if $m = n$. Actually,

$$P = AA_{left}^{-1} = A(A^TA)^{-1}A^T$$

is the matrix that projects $\Re^m$ onto the column space of A. This is the closest we can get to the matrix product $AB = I$.

Right inverse

Similarly, if A is $m \times n$ with full row rank (i.e. $r = rank(A) = m, m < n$), we can define its right inverse. In this case, AA^T is an invertible $m \times m$ symmetric matrix. So, $AA^T(A^TA)^{-1} = I$. The *right inverse* of A is

$$A_{right}^{-1} = A^T(AA^T)^{-1}$$

as $AA_{right}^{-1} = I$.

Also note that $A_{right}^{-1}A$ is an $n \times n$ matrix which equals the identity matrix only if $m = n$. The matrix

$$P = A_{right}^{-1}A = A^T(AA^T)^{-1}A$$

projects $\Re^n$ onto the row space of A. It is as close as we can get to the matrix product $BA = I$.

Pseudoinverse

An $m \times n$ matrix A has a left inverse if it is with full column rank ($r = rank(A) = n$) and has a right inverse if it is with full row rank ($r = rank(A) = m$). The left inverse or the right inverse of a matrix is a pseudoinverse. If A has rank $r < min(m, n)$, then we need to consider the general **pseudoinverse**.

We can define the pseudoinverse A^+ of A as the matrix for which A^+A gives an identity operation on any row vector $\mathbf{x}$, i.e. $\mathbf{x} = A^+A\mathbf{x}$.

To find the pseudoinverse of A, we can start from the singular value decomposition,

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

where Σ is an $m \times n$ matrix with zero entry values except the first r row diagonal entries, which have nonzero values denoted by $\sigma_1, \sigma_2, \cdot\cdot\cdot, \sigma_r$. It is easy to find the inverses for

U and V as they are orthonormal. So we only need to find the pseudoinverse for Σ. The best we can get to an inverse for Σ is an $n \times m$ matrix Σ^+ which has nonzero elements $\sigma_1, \sigma_2, \cdots, \sigma_r$ along the diagonal in the first r rows. The pseudoinverse for A is

$$A^+ = (U\Sigma V^T)^+ = (V^T)^{-1}\Sigma^+ U^{-1} = V\Sigma^+ U^T$$

as U and V are orthogonal matrices and thus $U^{-1} = U^T$ and $V^{-1} = V^T$.

15.3 Discrete Data Sets

In science and engineering, we often encounter problems involving a large set of data. We would like to know whether the data set could be characterized by a few parameters and whether there are any correlations among the data. These can be analyzed using tools of discrete probability theory, which is a branch of statistics and deals with events that occur in countable sample spaces.

15.3.1 Standard Deviation

Standard deviation of a data set shows the degree of variation (or dispersion) from the average (mean, or expected value) of the data. A low standard deviation indicates that the data points tend to cluster near the mean, whereas a high standard deviation indicates that the data spread out over a wide range of values.

Consider a set of data samples X with n elements:

$$X = \{x_1, x_2, \cdots, x_n\} \tag{15.27}$$

The mean (average) μ of X is given by

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{1}{n}(x_1 + x_2 + \cdots + x_n) \tag{15.28}$$

The standard deviation s of the data set is defined as

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2} \tag{15.29}$$

The value $var(X) = s^2$ is referred to as the *variance*. Note that the denominator in (15.29) is $n-1$ rather than n. People found that using $n-1$ in the formula of finding the standard deviation of a data sample set gives results closer to our intuition of of the dispersion of the data. However, if one calculates the standard deviation of the whole population of the data, one should use n in the denominator of the formula and the mean is usually denoted as σ:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2} \tag{15.30}$$

and the variance is σ^2.

15.3.2 Covariance and Correlation

Standard deviation is useful for analyzing data sets that are ‘one dimensional’ such as the heights or the ages of individuals of a nation. In many situations, we want to look at the correlation between two ‘one dimensional’ data sets such as the relation between cancer rate and the body weights of individuals. Covariance provides a good measure of this kind of correlation.

Consider two sample data sets, X and Y. The size of each set is n:

$$\begin{aligned} X &= \{x_1, x_2, \cdots, x_n\} \\ Y &= \{y_1, y_2, \cdots, y_n\} \end{aligned}$$

Suppose μ_X and μ_Y are the means of X and Y respectively. We can define the covariance for these two sample data sets as

$$cov(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu_X)(y_i - \mu_Y) \tag{15.31}$$

Note that if $X = Y$, the covariance is reduced to the variance. Also, $cov(X,Y) = cov(Y,X)$.

Another quantity that closely relates to covariance is *correlation*. Suppose we consider the two data sets X', and Y' obtained by subtracting the means μ_X and μ_Y from the elements of X and Y respectively so that their means are 0. We can imagine that X' and Y' represent two vectors, $\mathbf{X}$ and $\mathbf{Y}$. The correlation between these two vectors is the cosine of the ‘angle’ θ between these two vectors:

$$cor(X,Y) = \cos\theta = \frac{\mathbf{X}\cdot\mathbf{Y}}{||\mathbf{X}||\,||\mathbf{Y}||} \tag{15.32}$$

where $||\mathbf{X}||$, and $||\mathbf{Y}||$ are the norms (magnitudes) of the vectors, and

$$\mathbf{X}\cdot\mathbf{Y} = \sum_{i=1}^{n} x_i y_i$$

If $\cos\theta = 1$, the two vectors are ‘pointing in the same direction’, which means that the two data sets are perfectly correlated. If $\cos\theta = 0$, the two vectors are perpendicular, meaning that they are totally uncorrelated. Equation (15.32) can be expressed in the statistical form:

$$cor(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \mu_X)(y_i - \mu_Y)}{(n-1)s_X s_Y} \tag{15.33}$$

where s_X and s_Y are the standard deviations of X and Y respectively.

15.3.3 Correlation and Covariance Matrices

Covariance is a measure for two one-dimensional data sets. (Unless otherwise stated, in this section we refer to a set of data as a one-dimensional data set.) If we have more than two data sets, we can calculate the covariance values of different data set pairs and put them in a matrix, which is referred to as a *covariance matrix*. For example, if we have 3 data sets, X, Y, and Z, we could calculate $cov(X, Y)$, $cov(Y, Z)$, $cov(Z, X)$. For n data sets, there are $\binom{n}{2}$ covariance values. Another statistical quantity that closely relates to covariance matrix is the *correlation matrix*.

Let us consider a simple example to illustrate these concepts. Suppose we want to find out how closely blood sugar and cholesterol levels for a group of obese kids correlate with body weights. The data measured in relative units are shown in Table 15-1 below.

Table 15-1 Health Data of Kids

Person	**Weight** $X = (x_i)$	**Cholesterol** $Y = (y_i)$	**Blood Sugar** $Z = (z_i)$
P1	198	200	196
P2	160	165	165
P3	158	158	133
P4	150	165	91
P5	175	182	151
P6	134	135	101
P7	152	136	80
Mean	$\mu_X = 161$	$\mu_Y = 163$	$\mu_Z = 131$

We would like to measure obesity compared between each set of blood sugar level data or cholesterol level. We compute the deviations of each set of data from the means,and put the values in a matrix:

$$D = \begin{pmatrix} 37 & 37 & 65 \\ -1 & 2 & 34 \\ -3 & -5 & 2 \\ -11 & 2 & -40 \\ 14 & 19 & 20 \\ -27 & -28 & -30 \\ -9 & -27 & -51 \end{pmatrix} \tag{15.34}$$

The column vectors of D represent the deviations from the mean for each of the three sets of data. The mean for each column vector of D is 0; the first column represents $(x_i - \mu_X)$, the second column is $(y_i - \mu_Y)$, and the third column is $(z_i - \mu_Z)$. We can now easily calculate the covariance between two data sets. For example, $cov(X, Y)$, the covariance of X and Y is the dot product of the first two colum vectors divided by $n - 1$, n being the number of rows. The square of the first column vector is essentially the variance of X, which is equal to $cov(X, X)$. The covariance matrix C of this problem is a matrix in the form:

$$C = \begin{pmatrix} cov(X,X) & cov(X,Y) & cov(X,Z) \\ cov(Y,X) & cov(Y,Y) & cov(Y,Z) \\ cov(Z,X) & cov(Z,Y) & cov(Z,Z) \end{pmatrix} \tag{15.35}$$

From the definition of covariance of (15.31) and the properties of matrix multiplication, we can express the covariance matrix as

$$C = \frac{1}{n-1} D^T D \tag{15.36}$$

Evaluating all the covariance values using the data of (15.34), we obtain the covariance matrix:

$$\begin{aligned} C &= \frac{1}{6}\begin{pmatrix} 37 & -1 & -3 & -11 & 14 & -27 & -9 \\ 37 & 2 & -5 & 2 & 19 & -28 & -27 \\ 65 & 34 & 2 & -40 & 20 & -30 & -51 \end{pmatrix}\begin{pmatrix} 37 & 37 & 65 \\ -1 & 2 & 34 \\ -3 & -5 & 2 \\ -11 & 2 & -40 \\ 14 & 19 & 20 \\ -27 & -28 & -30 \\ -9 & -27 & -51 \end{pmatrix} \\ &= \begin{pmatrix} 417.7 & 437.5 & 725.7 \\ 437.5 & 546.0 & 830.0 \\ 725.7 & 830.0 & 1814.3 \end{pmatrix} \end{aligned} \tag{15.37}$$

The diagonal entries of C are the variances of the three data sets, X, Y, and Z. The off-diagonal entries are the covariances.

We can also calculate the corresponding correlation matrix $R = (r_{ij})$ from D. Suppose d_i is the i-th column vector of D. From (15.31) or (15.32), the (i, j)-th entry of R is given by

$$r_{ij} = \frac{\mathbf{d_i} \cdot \mathbf{d_j}}{||\mathbf{d_i}||\ ||\mathbf{d_j}||} \quad i, j = 1, 2, 3 \tag{15.38}$$

In this example,

$$R = \begin{pmatrix} 1.000 & 0.916 & 0.834 \\ 0.916 & 1.000 & 0.834 \\ 0.834 & 0.834 & 1.000 \end{pmatrix} \tag{15.39}$$

The three sets of data in our example are all positively correlated, because all entries in R are positive. This means that obesity, cholesterol level and blood sugar level of a kid are all correlated. A value of 0 would mean that the vectors are orthogonal and are uncorrelated. A negative value would mean that the two data sets are negatively correlated. In image and speech problems, the data are usually highly correlated. Note that both C and R are symmetric.

In this example, we have considered three sample data sets X, Y, and Z. Each of them can be regarded a 1-dimensional data set. We can also group the data together to form one data set, say S. This newly formed data set S, consisting of X, Y, and Z, is 3-dimensional. If there is strong correlation between X, Y, and Z, we may be able to predict one from the other, and the dimension of the data set S can be reduced.

15.4 Eigenvectors and Eigenvalues

To understand eigenvectors, we first consider operations in 2D Euclidean space. A transformation is represented by a 2×2 matrix and a vector is a 2×1 matrix. A transformation of a vector can be described by the multiplication of a transformation matrix and the vector, which gives us a new vector. Consider the example,

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 7 \end{pmatrix} \tag{15.40}$$

In this example, like most transformations, we cannot express the resulted vector $\begin{pmatrix} 3 \\ 7 \end{pmatrix}$, as a scalar multiple of the original vector $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$.

Now consider another example with a 2×2 transformation matrix A and a 2×1 vector $\mathbf{v}$:

$$A\mathbf{v} = \begin{pmatrix} 7 & 2 \\ 3 & 8 \end{pmatrix} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -5 \\ 5 \end{pmatrix} = 5 \begin{pmatrix} -1 \\ 1 \end{pmatrix} \tag{15.41}$$

In this example, the resulted vector $\begin{pmatrix} -5 \\ 5 \end{pmatrix}$ can be expressed as 5 times the original vector $\mathbf{v} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$. This means that the transformed vector $A\mathbf{v}$ points in the same direction as the original vector $\mathbf{v}$. The vector $\mathbf{v}$ (and any scalar multiple of it, $\lambda\mathbf{v}$) is an *eigenvector* of the transformation matrix A; the scalar multiple λ is an *eigenvalue* of the transformation. We can formally define these quantities as follow.

> Suppose A is an $n \times n$ matrix. We call a scalar λ an *eigenvalue* or a *characteristic value* of A if there exists a nonzero vector $\mathbf{v}$ such that $A\mathbf{v} = \lambda\mathbf{v}$. We call the vector $\mathbf{v}$ an *eigenvector* or a *characteristic vector* belonging to λ.

The equation $A\mathbf{v} = \lambda\mathbf{v}$ can be expressed in the form

$$(A - \lambda I)\mathbf{v} = \mathbf{0} \tag{15.42}$$

where **0** is the zero vector whose entries are all equal to zero. Therefore, λ is an eigenvalue of A if and only if (15.42) has a nontrivial solution, which is true if and only if $A - \lambda I$ is singular (otherwise we can multiply (15.42) by the inverse and get $\mathbf{v} = \mathbf{0}$), or equivalently, its determinant is zero:

$$|A - \lambda I| = 0 \tag{15.43}$$

If we expand $|A - \lambda I|$, we obtain an nth degree polynomial in λ:

$$p(\lambda) = |A - \lambda I| = c_0 + c_1\lambda + \cdots + c_n\lambda^n \tag{15.44}$$

We call this polynomial the *characteristic polynomial*, and equation (15.43) the *characteristic equation*, for the matrix A. The eigenvalues of A are the roots of the characteristic equation. An *eigenspace* of A is the set of all eigenvectors with the same eigenvalue, together with the zero vector.

Here is a summary of the properties of eigenvectors and eigenvalues:

1. Eigenvectors can only be found for square matrices.
2. Not every square matrix has eigenvectors.
3. Given an $n \times n$ matrix that does have eigenvectors, there are n of them; some of the eigenvalues may be complex numbers. So a 3×3 matrix has 3 eigenvectors.
4. The eigenvectors of a symmetric matrix (i.e. $A = A^T$) are 'perpendicular' to each other. That is, they are at right angles to each other. The mathematical term for 'perpendicular' is *orthogonal*.
5. People are more interested to find eigenvectors with unit lengths. The unit eigenvectors of a symmetric matrix may form an orthonormal basis of a coordinate system.

Example 15.1

Find the eigenvalues and corresponding eigenvectors of the matrix

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

Solution

The characteristic equation is

$$\begin{vmatrix} 4-\lambda & 3 \\ -2 & -1-\lambda \end{vmatrix} = 0 \quad \text{or} \quad \lambda^2 - 3\lambda + 2 = 0 \tag{15.45}$$

The eigenvalues of A are the roots of equation (15.45), which are $\lambda_1 = 1$ and $\lambda_2 = 2$. To find the eigenvector belonging to $\lambda_1 = 1$, we need to solve for $\mathbf{v}$ of equation (15.42). That is,

$$(A - \lambda_1 I)\mathbf{v} = \mathbf{0} \quad \textit{or} \quad \begin{pmatrix} 3 & 3 \\ -2 & -2 \end{pmatrix}\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{15.46}$$

From this, we obtain the duplicate equations:

$$\begin{aligned} 3v_1 + 3v_2 &= 0 \\ -2v_1 - 2v_2 &= 0 \end{aligned} \tag{15.47}$$

which can be reduced to

$$v_1 + v_2 = 0 \tag{15.48}$$

If we let $v_2 = t$, then $v_1 = -t$. Therefore $\mathbf{e_1} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$ is an eigenvector of A belonging to $\lambda_1 = 1$. Actually, all multiples of $\mathbf{e_1}$ are eigenvectors of A for λ_1. We can claim that $\mathbf{e_1}$ is the basis of the eigenspace corresponding to $\lambda_1 = 1$.
Similarly, with $\lambda_2 = 2$, we have the duplicate equations,

$$\begin{aligned} 2v_1 + 3v_2 &= 0 \\ -2v_1 - 3v_2 &= 0 \end{aligned} \tag{15.49}$$

An eigenvector of A for λ_2 is $\mathbf{e_s} = \begin{pmatrix} -3 \\ 2 \end{pmatrix}$. The corresponding eigenspace for $\lambda_2 = 2$ is given by the span of $\mathbf{e_2}$.

For matrices with higher dimensions, we can solve for the eigenvectors using Gaussian elimination.

Consider another example, where A is a 3×3 symmetric matrix:

$$A = \begin{pmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{pmatrix}$$

The roots for $|A - \lambda I| = 0$ are $\lambda_1 = -1, \lambda_2 = -1$, and $\lambda_3 = 8$. The corresponding eigenvectors are:

$$\mathbf{v_1} = \begin{pmatrix} 1 \\ -2 \\ 0 \end{pmatrix}, \quad \mathbf{v_2} = \begin{pmatrix} 4 \\ 2 \\ -5 \end{pmatrix}, \quad \mathbf{v_3} = \begin{pmatrix} 2 \\ 1 \\ 2 \end{pmatrix}$$

The eigenvectors $\mathbf{v_1}, \mathbf{v_2}$, and $\mathbf{v_3}$ are **orthogonal** to each other, which is a consequence of the property of a symmetric matrix mentioned above. For example, $\mathbf{v_2} \cdot \mathbf{v_3} = 4 \times 2 + 2 \times 1 + (-5) \times 2 = 0$.

Suppose we express the eigenvectors as row vectors $\mathbf{e_i}$'s and normalize them to unit vectors, i.e., $\mathbf{e_i} = \mathbf{v_i^T}/|\mathbf{v_i}|$. Then we have

$$\mathbf{e_1} = \frac{1}{\sqrt{5}}(1, -2, 0), \quad \mathbf{e_2} = \frac{1}{3\sqrt{5}}(4, 2, -5), \quad \mathbf{e_3} = \tfrac{1}{3}(2, 1, 2)$$

which form an orhtonormal basis (i.e., $\mathbf{e_i} \cdot \mathbf{e_j} = \delta_{ij}$). We can form an orthogonal matrix (see definition above) P using the basis vectors:

$$P = \begin{pmatrix} \mathbf{e_1} \\ \mathbf{e_2} \\ \mathbf{e_3} \end{pmatrix} = \frac{1}{3\sqrt{5}} \begin{pmatrix} 3 & -6 & 0 \\ 4 & 2 & -5 \\ 2\sqrt{5} & \sqrt{5} & 2\sqrt{5} \end{pmatrix} = \begin{pmatrix} 0.45 & -0.89 & 0.00 \\ 0.60 & 0.30 & -0.75 \\ 0.67 & 0.33 & 0.67 \end{pmatrix}$$

As P is an orthogonal matrix, its inverse is given by its transpose:

$$P^{-1} = P^T = \begin{pmatrix} 0.45 & 0.60 & 0.67 \\ -0.89 & 0.30 & 0.33 \\ 0.00 & -0.75 & 0.67 \end{pmatrix}$$

We will see in the next section that P can be viewed as a projection matrix. If B is a $3 \times n$ matrix, the operation PB is to 'project' the n column vectors of B onto the new orthonormal basis, $\mathbf{e_1}$, $\mathbf{e_2}$, and $\mathbf{e_3}$.

One can regard that such an operation is a rotation of a 3D coordinate system. The operation of P^{-1} is to rotate the coordinate system to the new basis. Imagine that a solid object consists of n vertices and their original coordinates are given by the column vectors of B. The values of vertices in the rotated coordinate system are given by the columns of PB. Figure 15-3 below shows the original basis (**x, y, z**), the new basis $(\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3})$ and the 8 vertices of a cube centered at the origin with length 0.2. By applying appropriate rotations about the axes **x**, **y**, and **z**, we can rotate (**x, y, z**) onto $(\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3})$. The columns of matrix B, which is now 3×8, are the coordinates of the 8 cube vertices:

$$B = \begin{pmatrix} -0.1 & 0.1 & 0.1 & -0.1 & 0.1 & 0.1 & -0.1 & -0.1 \\ -0.1 & -0.1 & 0.1 & 0.1 & -0.1 & 0.1 & 0.1 & -0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & -0.1 & -0.1 & -0.1 & -0.1 \end{pmatrix}$$

Each column of the matrix product PB represents a vertex of the cube in the new coordinate system. Another interpretation of PB is that we rotate the cube and the PB columns are the vertex values of the rotated cube in the original coordinate system. This is shown in Figure 15-4. The rotated cube vertices are given by:

$$B' = PB = \begin{pmatrix} 0.05 & 0.13 & -0.05 & -0.13 & 0.13 & -0.05 & -0.13 & 0.05 \\ -0.16 & -0.05 & 0.02 & -0.10 & 0.10 & 0.16 & 0.05 & -0.02 \\ -0.03 & 0.10 & 0.17 & 0.03 & -0.03 & 0.03 & -0.10 & -0.17 \end{pmatrix}$$

Each column is a vertex.

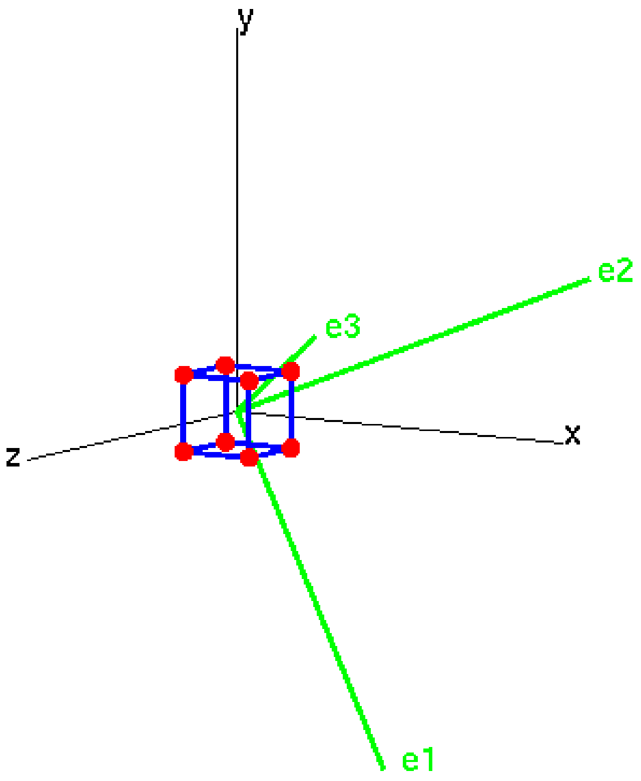

Figure 15-3 Eigenvectors Forming New Basis

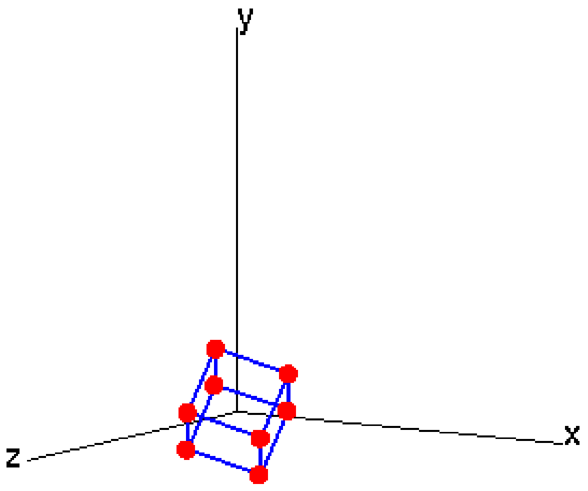

Figure 15-4 Rotated Cube with Vertex Values Given by PB

15.5 Principal Component Analysis

Principal Component Analysis (PCA) was invented by Karl Pearson in 1901, and was later popularized by Harold Hotelling. PCA is a useful tool for analyzing high dimension data sets. It is effective for pattern search, dimensionality reduction, lossy data compression, feature extraction, and data visualization. Actually, PCA is a simple case of the eigenvector-based multivariate analysis. It also closely relates to factor analysis, which is a statistical method useful for reducing the number of variables in gathering data.

The goal of PCA is to identify the most meaningful basis to re-express a data set, hoping that the new basis will filter out the noise and reveal hidden structures. The technique uses an orthogonal transformation to convert a multi-dimension data set to a set of values of linearly uncorrelated variables called *principal components*. The number of principal components is less than or equal to the dimension of the original data set. We setup the transformation in such a way that the first principal component has the largest possible variance and each succeeding component is orthogonal to (i.e. uncorrelated with) the preceding components and in turn has the largest subsequent variance. In the process, we want to answer the question: *Can we find another basis, which can be expressed as a linear combination of the original basis, that best re-expresses our data set*?

Suppose A is the original data set arranged as an $m \times n$ matrix. Suppose P is an $m \times m$ matrix that transforms A to another $m \times n$ matrix B. That is,

$$PA = B \tag{15.50}$$

Suppose $\mathbf{a_i}$ and $\mathbf{b_i}$ are the i-th **column** vectors of A and B, and $\mathbf{p_i}$ is i-th **row** vector of P. We can interpret $\{\mathbf{p_1}, \cdots, \mathbf{p_m}\}$ as a set of new basis vectors for expressing the column vectors of A:

$$PA = \begin{pmatrix} \mathbf{p_1} \\ \cdot \\ \cdot \\ \mathbf{p_m} \end{pmatrix} \begin{pmatrix} \mathbf{a_1}, & \cdots, & \mathbf{a_n} \end{pmatrix} = \begin{pmatrix} \mathbf{p_1} \cdot \mathbf{a_1}, & \cdots, & \mathbf{p_n} \cdot \mathbf{a_n}, \\ \cdot & \cdots & \cdot \\ \cdot & \cdots & \cdot \\ \mathbf{p_m} \cdot \mathbf{a_1}, & \cdots, & \mathbf{p_m} \cdot \mathbf{a_n}, \end{pmatrix} = B \tag{15.51}$$

We can see that a column vector $\mathbf{b_i}$ of B has the form:

$$\mathbf{b_i} = \begin{pmatrix} \mathbf{p_1} \cdot \mathbf{a_i} \\ \cdot \\ \cdot \\ \mathbf{p_m} \cdot \mathbf{a_i} \end{pmatrix} \tag{15.52}$$

We recognize that each component of the vector $\mathbf{b_i}$ is a dot-product of $\mathbf{a_i}$ with the corresponding row in P. In another perspective, the k-th component of $\mathbf{b_i}$ is a projection on to the k-th row of P. We can therefore interpret the rows of P as a new set of basis vectors for representing the column vectors of A. This concept has been illustrated in the example of the previous section. The row vectors $\mathbf{p_i}$'s become the *principal components* of A. The question that remains is how to make a good choice of P to best re-express A? In a 2 variable case, we may use the least-square fitting method to find the line that best-fits the data. *How do we quantify and generalize these notions to arbitrarily higher dimensions?* The PCA technique makes use of covariance matrix and eigenvectors to address this question. The PCA technique is discussed below. We only present the steps of using PCA to find the best basis but we omit the proofs of some statements we claim to be true based on some simple examples.

15.5.1 PCA Procedures

Calculate the Deviation Matrix

The first step of the PCA procedures is to subtract the mean from the data for each of the data dimensions. Suppose we use the 3 dimensional health data for a group of kids of Table 15-1 above as example. After the subtraction, we obtain the deviation matrix D of equation (15.34). The new mean for each column vector (data of each dimension) of D is 0.

Calculate the Covariance Matrix

The second step is to calculate the covariance matrix, which is given by equation (15.36). (Alternatively, one can use the correlation matrix in the process.) In our example, $m = 3, n = 7$, and the covariance matrix is

$$C = \frac{1}{n-1} D^T D = \begin{pmatrix} 417.7 & 437.5 & 725.7 \\ 437.5 & 546.0 & 830.0 \\ 725.7 & 830.0 & 1814.3 \end{pmatrix} \tag{15.53}$$

Note that a covariance matrix is always **symmetric** (i.e. $C^T = C$). Therefore, the eigenvectors of a covariance matrix are always **orthogonal**.

Calculate Eigenvectors

The next step is to calculate the eigenvalues and eigenvectors of the covariance matrix. We have discussed in the previous section how to find eigenvalues and eigenvectors of low-rank matrices. For the symmetric covariance matrix C of (15.53), the eigenvalues are:

$$\lambda_1 = 39.57, \quad \lambda_2 = 180.29, \quad \lambda_3 = 2558.14$$

The corresponding normalized eigenvectors are:

$$\begin{aligned} \mathbf{e_1} &= (0.770, & -0.638, & \quad -0.016) \\ \mathbf{e_2} &= (0.522, & 0.644, & \quad -0.559) \\ \mathbf{e_3} &= (0.367, & 0.422, & \quad 0.829) \end{aligned} \tag{15.54}$$

The eigenvectors of (15.54) have been normalized (i.e. length = 1) and are orthogonal. Therefore, the vectors $\{\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3}\}$ are orthonormal. The projection matrix P and data matrix A of (15.50) are given by

$$P = \begin{pmatrix} \mathbf{e_1} \\ \mathbf{e_2} \\ \mathbf{e_3} \end{pmatrix} \quad \text{and} \quad A = D^T \tag{15.55}$$

So

$$B = PA = \begin{pmatrix} \mathbf{e_1} \\ \mathbf{e_2} \\ \mathbf{e_3} \end{pmatrix} \begin{pmatrix} \mathbf{a_1}, & \cdots, & \mathbf{a_7} \end{pmatrix} \tag{15.56}$$

The square projection matrix of (15.55) is composed of unit row vectors which are orthogonal to each other. Therefore, it is an *orthogonal matrix*. As mentioned above, an important property of an orthogonal matrix is that its inverse is equal to its transpose (i.e. $P^{-1} = P^T$).

Figure 15-5 below shows a plot of the adjusted data (mean subtracted) of X, Y, and Z of Table 15-1 and the eigenvectors. The axes labeled 1, 2, and 3 correspond to the eigenvectors for eigenvalues λ_1, λ_2, and λ_3 respectively.

From the figure, we see that most data points cluster around the axis of λ_3, which is the largest eigenvalue. When we project the data onto this axis, we'll get large values. So one should choose this axis to be the principal axis of the new basis. The axis of λ_2 has the second largest eigenvalue and it has less data points cluster around it. The axis of λ_1 has a much smaller eigenvalue and there is hardly any data point cluster around it. When we project the data onto this axis, we get values close to zero. From this simple example, we can see that the larger the eigenvalue, the more important the corresponding eigenvector is and the ones with the largest eigenvalues should be chosen as the principal components. It turns out that one can prove that this is generally true. The eigenvectors of the highest eigenvalues are the principal components of the data set.

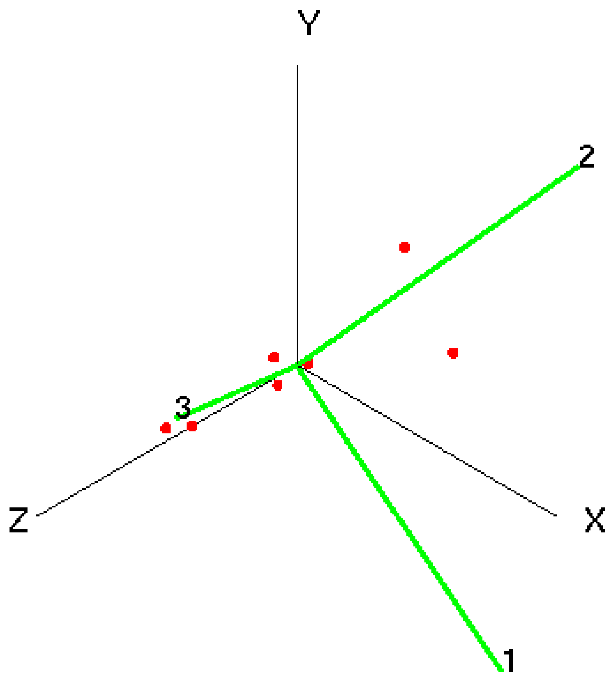

Figure 15-5 A Plot of Data of D of (15.34) and the Eigenvectors

Choosing Components

In general, once eigenvectors have been found from the covariance matrix, we order them according to their eigenvalues, from highest to lowest. This gives us the components in order of significance. At this point, if necessary we can discard the components of lesser significance, which may result in loss of some information, but if the eigenvalues are small, the loss is not significant. If we discard some components, the final data set is of lower dimension as compared to the original one. (As a consequence, PCA can be used in lossy data compression.) We obtain a lower dimension projection matrix by keeping the remaining eigenvectors. In our example, we can form a matrix F, for example, by throwing away $\mathbf{e_1}$ and keeping only $\mathbf{e_2}$ and $\mathbf{e_3}$:

$$F = \begin{pmatrix} \mathbf{e_3} \\ \mathbf{e_2} \end{pmatrix}$$

Each vector $\mathbf{e_i}$ of F is sometimes called a feature vector because it is chosen to represent some characteristics or attributes of the original data set while still only partially describing it.

Deriving New Data Set

A new data set with reduced dimension is obtained by projecting the original data onto the principal axes. This is achieved by multiplying the feature vector matrix F by the matrix A that contains the deviation data ($A = D^T$). Suppose the new data matrix is B, then

$$\begin{aligned} B = FA &= \begin{pmatrix} 0.367 & 0.422 & 0.829 \\ 0.522 & 0.644 & -0.559 \end{pmatrix} \begin{pmatrix} 37 & -1 & -3 & -11 & 14 & -27 & -9 \\ 37 & 2 & -5 & 2 & 19 & -28 & -27 \\ 65 & 34 & 2 & -40 & 20 & -30 & -51 \end{pmatrix} \\ &= \begin{pmatrix} 83.078 & 28.663 & -1.553 & -36.353 & 29.736 & -46.595 & -56.976 \\ 6.807 & -18.240 & -5.904 & 17.906 & 8.364 & -15.356 & 6.423 \end{pmatrix} \end{aligned} \tag{15.57}$$

The projection of (15.57) basically transforms our data so that they are expressed in terms of the lines where data tend to cluster around them.

Recovering Data Set

The transformation of (15.57) is a lossy transformation as we have thrown away one eigenvector. We cannot recover the exact original data from the transformed data. On the other hand, the transformation using P of (15.55) is lossless and reversible. If $B = PA$, then $A = P^{-1}B$. Since P is an orthogonal matrix, $P^{-1} = P^T$. So

$$A = P^T B \tag{15.58}$$

We can use this equation to recover the exact data transformed by P. However, if we have thrown away some components and the feature matrix F of (15.57) is not the same as the projection matrix P, we cannot recover the exact data. Also, in this case F is not a square matrix and does not have an inverse. Of course, the 'recovered' data will not be identical to that of the original set.

Without going into details of proving, we claim that the 'recovered' data can be approximated by $F^T B$. In this example, F is 2×3. So F^T is 3×2 and the 'recovered' data set is

$$\begin{aligned} A' = F^T B &= \begin{pmatrix} \mathbf{e_3^T}, \mathbf{e_2^T} \end{pmatrix} B \\ &= \begin{pmatrix} 0.367 & 0.522 \\ 0.422 & 0.644 \\ 0.829 & -0.559 \end{pmatrix} \begin{pmatrix} 83.08 & 28.66 & -1.55 & -36.35 & 29.74 & -46.60 & -56.96 \\ 6.80 & -18.24 & -5.904 & 17.91 & 8.36 & -15.36 & 6.42 \end{pmatrix} \\ &= \begin{pmatrix} 34.81 & 1.00 & -3.65 & -3.99 & 16.05 & -25.12 & -17.56 \\ 38.80 & 0.35 & -4.46 & -3.81 & 17.30 & -29.55 & -19.91 \\ 65.05 & 33.96 & 2.01 & -40.15 & 19.96 & -30.04 & -50.82 \end{pmatrix} \end{aligned} \tag{15.59}$$

Figure 15-6 shows a plot of the recovered data of (15.59) along with the original data and eigenvectors of Figure 15-5; the recovered data points are shown as black square dots. The recovered data here are the deviation data. If we want to get back the very original data of Table 15-1, we need to add the means μ_X, μ_Y, and μ_Z accordingly to the recovered data of (15.59).

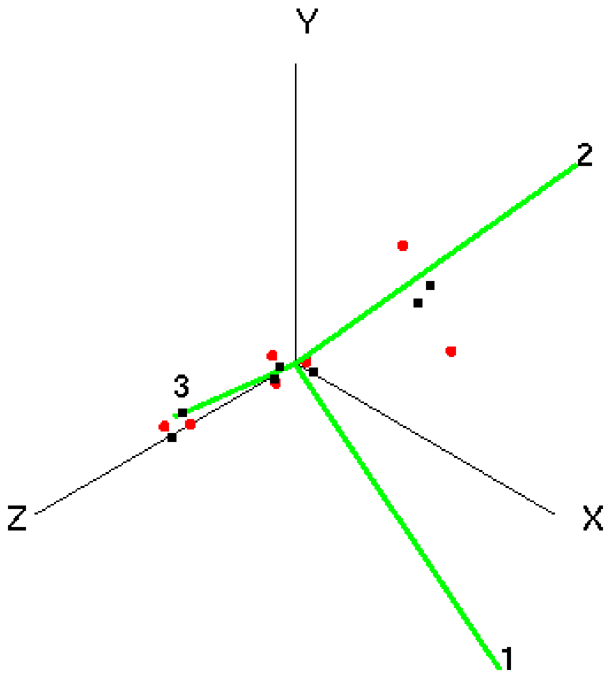

Figure 15-6 Recovered Data of Figure 15-5

15.6 Eigenvectors by Jacobi Method

We have discussed how to find egeinvalues and eigenvectors of a square matrix by solving a characteristic equation. This method works well for low-rank matrices as illustrated in the examples of section 15.4. However, for a large data set that involves large-size matrices, it is impractical to find eignevectors by solving characteristic equations. Instead, numerical methods are employed to find the eigenvectors and eigenvalues. There are a few popular numerical methods that can be used to give a general solution to the problem. However, here we are not interested in the general solutions of finding eigenvectors of a square matrix. We are interested in the problem of finding eignevectors of a **symmetric** matrix as a covariance matrix is always symmetric. The solution to such a problem is a lot simpler as compared to solving the general problem. Again various numerical methods exist. The one that we will discuss is called the *Jacobi Method*, which diagonalizes a square symmetric matrix to obtain eigenvalues and eigenvectors.

We have discussed that a symmetric matrix is diagonalizable. Finding eigenvalues and eigenvectors of a diagonal matrix is trivial. For example, if A is a 3×3 diagonal matrix, it is in the form

$$A = \begin{pmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & a_{33} \end{pmatrix} \tag{15.60}$$

The eigenvalues of this matrix are $\lambda_1 = a_{11}, \lambda_2 = a_{22}$, and $\lambda_3 = a_{33}$, and the eigenvectors are

$$\mathbf{e_1} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{e_2} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad \mathbf{e_3} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{15.61}$$

The eigenvectors obviously form an orthonormal basis. Note that here we express eigenvectors as column vectors rather than row vectors as we did in the previous section.

The idea of the Jacobi method is to transform iteratively a symmetric matrix A to a diagonal form through a sequence of 'rotations', which transform the original basis to the orthonormal basis formed by the eigenvectors of A. This concept can be visualized using an example of a 2D matrix as shown in Figure 15-7 below.

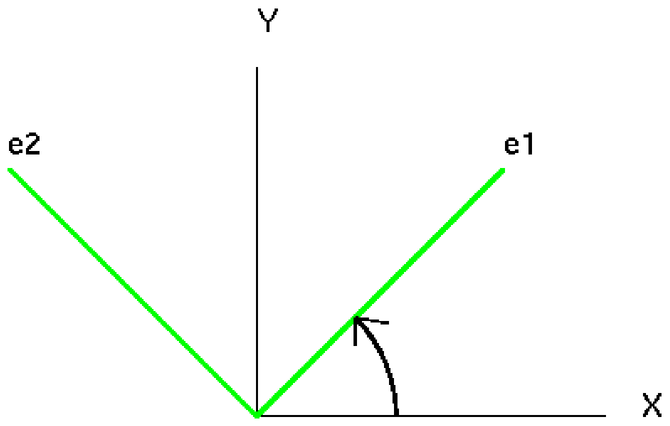

Figure 15-7 Rotating Orthonormal X-Y Basis to Orthonormal e1-e2 Basis

The 'rotations' are elementary orthogonal transformations that are often called *Jacobi rotations*, and have the form

$$U(p, q, \phi) = \begin{pmatrix} 1 & 0 & \cdots & \cdot & & \cdot & 0 \\ 0 & 1 & 0 \cdot\cdot & \cdot & & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & & \cdot & \cdot \\ \cdot & \cdot & \cos\phi & & \sin\phi \cdot\cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & 1 & & \cdot & \cdot \\ \cdot & \cdot & -\sin\phi & & \cos\phi \cdot\cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & & \cdot & \cdot \\ 0 & 0 & \cdots & \cdot & & \cdot & 1 \end{pmatrix} \tag{15.62}$$

The diagonal elements of $U(p, q, \phi) = (u_{ij})$ are all 1 except the two elements at rows and columns p and q, which are equal to cos ϕ (i.e. $u_{pp} = u_{qq} =$ cos ϕ). All off-diagonal elements are 0 except $u_{pq} =$ sin ϕ, and $u_{qp} = -$sin ϕ. Obviously, this matrix is orthogonal as $\sin^2\phi + \cos^2\phi = 1$.

Starting with a symmetric matrix A_0, in the k-th step of Jacobi rotations, a rotation matrix $U_k = U_k(p, q, \phi)$ of the form (15.62) is used to transform the matrix A_k to A_{k+1}:

$$A_{k+1} = U_k^T A_k U_k \tag{15.63}$$

The composite Jacobi rotations approximate the operation

$$A \to D = V^T AV \tag{15.64}$$

where D is a diagonal matrix and V is orthogonal, D being the limit of A_k when $k \to \infty$; the matrix V is the product of all Jacobi rotations:

$$V = U_0 U_1 U_2 \cdots \tag{15.65}$$

Suppose we let $A = A_k$, $A' = A_{k+1}$, and $U = U_k(p, q, \phi)$. We can then express (15.63) as

$$A' = U^T AU \tag{15.66}$$

The operation $U^T A$ only changes rows p and q of A, while AU only changes columns p and q. Therefore, only the elements of rows p and q, and columns of p and q of A will be changed in (15.66). The new matrix A' is of the form

$$A' = \begin{pmatrix} a_{11} & \cdots & a'_{1p} & \cdot\cdot & a'_{1q} & \cdot\cdot & a_{1n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a'_{p1} & \cdots & a'_{pp} & \cdot\cdot & a'_{pq} & \cdot\cdot & a'_{pn} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a'_{q1} & \cdots & a'_{qp} & \cdot\cdot & a'_{qq} & \cdot\cdot & a'_{qn} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{n1} & \cdots & a'_{np} & \cdot\cdot & a'_{nq} & \cdot\cdot & a_{nn} \end{pmatrix} \tag{15.67}$$

Multiplying out (15.66) and using the special form of U, we obtain the explicit formulas for the elements of A':

$$\begin{aligned} a'_{rp} &= a_{rp}\cos\phi - a_{rq}\sin\phi & r \neq p, r \neq q && (15.68a) \\ a'_{rq} &= a_{rq}\cos\phi + a_{rp}\sin\phi & r \neq p, r \neq q && (15.68b) \\ a'_{pp} &= a_{pp}\cos^2\phi + a_{qq}\sin^2\phi - 2a_{pq}\sin\phi\cos\phi & && (15.68c) \\ a'_{qq} &= a_{pp}\sin^2\phi + a_{qq}\cos^2\phi + 2a_{pq}\sin\phi\cos\phi & && (15.68d) \\ a'_{pq} &= a_{pq}(\cos^2\phi - \sin^2\phi) + (a_{pp} - a_{qq})\sin\phi\cos\phi & && (15.68e) \end{aligned}$$

The idea of the Jacobi method is to make the off-diagonal elements of A to become 0 through rotations. That is, we want the term a'_{pq} in (15.68e) to be 0. As a consequence, we obtain the equation

$$0 = a_{pq}\cos 2\phi + (a_{pp} - a_{qq})\frac{1}{2}\sin 2\phi \tag{15.69}$$

from which we can solve for the rotation angle ϕ:

$$\cot 2\phi = \frac{\cos 2\phi}{\sin 2\phi} = \frac{a_{qq} - a_{pp}}{2a_{pq}} \tag{15.70}$$

where we have used the trigonometric identities $cos\ 2\phi = cos^2\phi - sin^2\phi$, and $sin\ 2\phi = 2sin\ \phi cos\ \phi$. We can obtain ϕ from cot 2ϕ and calculate other trigonometric quantities in (15.68). However, a simpler way is to solve for sin ϕ and cos ϕ directly from cot 2ϕ. If we let $t = \tan\phi$, then

$$\cot 2\phi = \frac{\cos 2\phi}{\sin 2\phi} = \frac{\cos^2\phi - \sin^2\phi}{2\sin\phi\cos\phi} = \frac{1}{2t} - \frac{t}{2} \tag{15.71}$$

We can rewrite (15.71) as

$$t^2 + 2(\cot 2\phi)t - 1 = 0 \tag{15.72}$$

The roots of the quadratic equation (15.72) are

$$t = -\cot 2\phi \pm \sqrt{\cot^2 2\phi + 1} \tag{15.73}$$

We should choose the smaller root which corresponds to a rotation angle less than $\pi/4$ in magnitude. Such a choice at each iteration gives a stable reduction. This can be implemented by the following java code segment:

```
double cot_2phi =   ( A[q][q] - A[p][p] ) / (2 * A[p][q]);
double tan_phi;
double d = Math.sqrt ( cot_2phi * cot_2phi + 1 );
if ( cot_2phi > 0 )
   tan_phi = -cot_2phi + d;
else
   tan_phi = -cot_2phi - d;
```

Other trigonometric quantities in (15.68) can be then obtained from $tan\ \phi$. By iterating the equation of (15.63), we eventually obtain a diagonal matrix $D = V^TAV$ as shown in equation (15.64). The diagonal elements of D give the eigenvalues of the original matrix A. The column vectors of V are the eigenvectors of A as $AV = D(V^T)^{-1} = DV$. They can be computed by carrying out the same rotation operation as that on matrix A at each iterative stage:

$$V_{k+1} = U_k^T V_k U_k \tag{15.74}$$

where initially, V_0 is the identity matrix.

The following java class, *JacobiCyclic* shown in Listing 15-1 shows a full implementation of this method. The function **eigenVs** of this class takes an $n \times n$ symmetric matrix A as a two dimensional array input. It returns n eigenvalues in the array *evalues*. The function also returns n eigenvectors, each with dimension n in the 2D array *evectors*; each eigenvector is a column vector of the array.

Program Listing 15-1: An Implementation of Finding Eigenvectors by Jacobi Method

```
import java.io.*;

class JacobiCyclic {
  private final static double eps = 1.0E-8;
  private static double threshold;
  private static double thresholdNorm;
  private static double max;

  // calculate eigenvalues and egienvectors
  // returns n eigenvectors as column vectors in evectors[][]
  // n eigenvalues are returned in evalues
```

```
static boolean eigenVs (int n, double [][] A,
                double [] evalues, double [][]  evectors )
{
  if ( n < 1 ) return false;
  if ( n == 1 ) {
    evalues[0] = A[0][0];
    evectors[0][0] = 1.0;
    return true;
  }
  //start with identity matrix
  for ( int i = 0; i < n; i++ )
    for ( int j = 0; j < n; j++ )
      if ( i == j )
        evectors[i][j] = 1.0;
      else
        evectors[i][j] = 0.0;
  //calculate threshold and thresholdNorm
  threshold = 0.0;
  for ( int i = 0; i < n - 1; i++ )
    for ( int j = i + 1; j < n; j++ )  //consider upper triangle only
      threshold += A[i][j] * A[i][j];
  threshold = Math.sqrt ( threshold + threshold );
  thresholdNorm = threshold * eps;
  max = threshold + 1.0;
  while ( threshold > thresholdNorm ) {
    threshold /= 10.0;
    if (max < threshold) continue;
    max = 0.0;
    for ( int k = 0; k < n - 1; k++ ) {
      for ( int m = k + 1; m < n; m++ ) {
        if ( Math.abs ( A[k][m] ) < threshold ) continue;
        //calculate angle of rotation to make A[k][m] 0
        double cot_2phi =   ( A[k][k] - A[m][m] ) / (2 * A[k][m]);
        double tan_phi;
        double t1, t2, t3;
        double d = Math.sqrt ( cot_2phi * cot_2phi + 1 );
        if ( cot_2phi > 0 )
          tan_phi = -cot_2phi + d;
        else
          tan_phi = -cot_2phi - d;
        double tan2_phi = tan_phi * tan_phi;
        double  sin2_phi = tan2_phi / (1.0 + tan2_phi);
        double  cos2_phi = 1.0 - sin2_phi;
        double  sin_phi = Math.sqrt(sin2_phi);
        if (tan_phi < 0.0) sin_phi = - sin_phi;
        double cos_phi = Math.sqrt(cos2_phi);
        double sin_2phi = 2.0 * sin_phi * cos_phi;
        double cos_2phi = cos2_phi - sin2_phi;
        t1 = A[k][k];
        t2 = A[m][m];
        t3 = A[k][m];
        A[k][k] = t1 * cos2_phi + t2 * sin2_phi + t3 * sin_2phi;
        A[m][m] = t1 * sin2_phi + t2 * cos2_phi - t3 * sin_2phi;
        A[k][m] = A[m][k] = 0;
        for ( int i = 0; i < n; i++ ){
          if ( i == k  ||  i == m  ) continue;
          if ( i < k )
            t1 = A[i][k];
          else
            t1 = A[k][i];
            if ( i < m )
              t2 = A[i][m];
            else
              t2 = A[m][i];
            t3 = t1 * cos_phi + t2 * sin_phi;
            if ( i < k )
              A[i][k] = t3;
            else
              A[k][i] = t3;
            t3 = - t1 * sin_phi + t2 * cos_phi;
            if ( i < m )
              A[i][m] = t3;
            else
              A[m][i] = t3;
          } //for i

          for ( int i = 0; i < n; i++ ) {
            t1 = evectors[i][k];
            t2 = evectors[i][m];
            evectors[i][k] = t1 * cos_phi + t2 * sin_phi;
            evectors[i][m] = -t1 * sin_phi + t2 * cos_phi;
          }
        } //for m
        for ( int i = 0; i < n; i++ ) {
          if ( i == k ) continue;
          else if ( max < Math.abs ( A[k][i] ) )
            max = Math.abs ( A[k][i] );
        }
      }  // for k
    } //while
    for ( int i = 0; i < n; i++ )
      evalues[i] = A[i][i];

    return true;
  }
}
```