Dimensionality Reduction

Dr. Muhammad Nouman Durrani

List of associated tutorials and Chapters to Lecture 4

- CHAPTER 11. DIMENSIONALITY REDUCTION Page 405-424

- Singular Value Decomposition (SVD): A Fast Track Tutorial

for SVD Matrices (U, Σ and V^T calculations of a 2x2 Matrix M)

- Singular Value Decomposition Tutorial

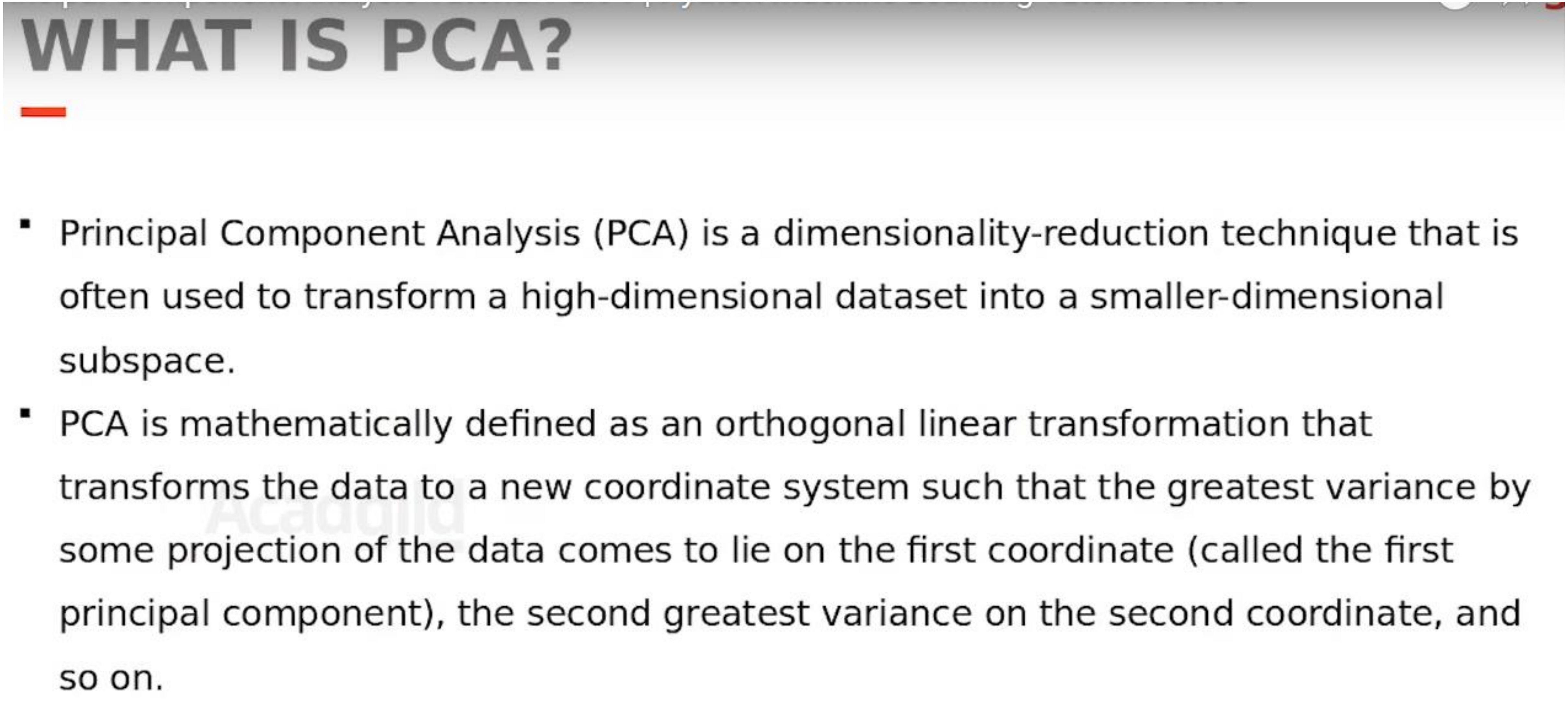
for Matrices related concepts, Eigen Values, Eigen Vectors, and SVD (3x3) matrix

- Finding eigenvalues and eigenvectors

for 3x3 matrix. Here row echelon form for calculating Eigen Vectors has also covered

Dimensionality reduction

- A matrix can be summarized by finding “narrower” matrices that in some sense close to the original matrix
- These narrow matrices have only a small number of rows or columns, but can be used much more efficiently than the original large matrix
- The process of finding these narrow matrices is called dimensionality reduction



WHAT IS PCA?

- Principal Component Analysis (PCA) is a dimensionality-reduction technique that is often used to transform a high-dimensional dataset into a smaller-dimensional subspace.
- PCA is mathematically defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.

Principal-component analysis (PCA)

- Principal-component analysis (PCA) is a technique for taking a dataset consisting of a set of tuples representing points in a high-dimensional space and finding the directions along which the tuples line up best
- The idea is to treat the set of tuples as a matrix M and find the eigenvectors for MM^T
- The matrix of these eigenvectors can be thought of as a rigid rotation in a high-dimensional space
- When you apply this transformation to the original data, the axis corresponding to the principal eigenvector is the one along which the points are most “spread out”

Principal-component analysis (PCA)

- On this axes, the variance of the data is maximized.
  - The points can best be viewed as lying along this axis, with small deviations from this axis.
- Likewise, the axis corresponding to the second eigenvector (the eigenvector corresponding to the second-largest eigenvalue) is the axis along which the variance of distances from the first axis is greatest, and so on.

Principal-component analysis (PCA)

- The high-dimensional data can be replaced by its projection onto the most important axes.
  - These axes are the ones corresponding to the largest eigenvalues.
- Thus, the original data is approximated by data that has many fewer dimensions and that summarizes well the original data.

An Illustrative Example

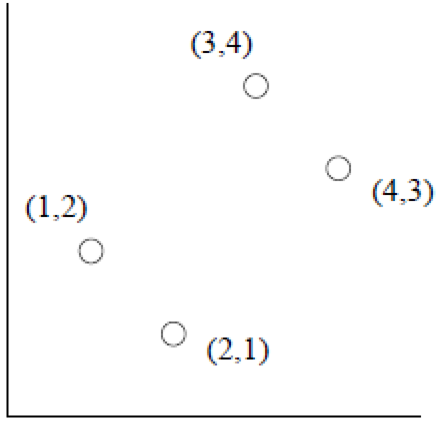
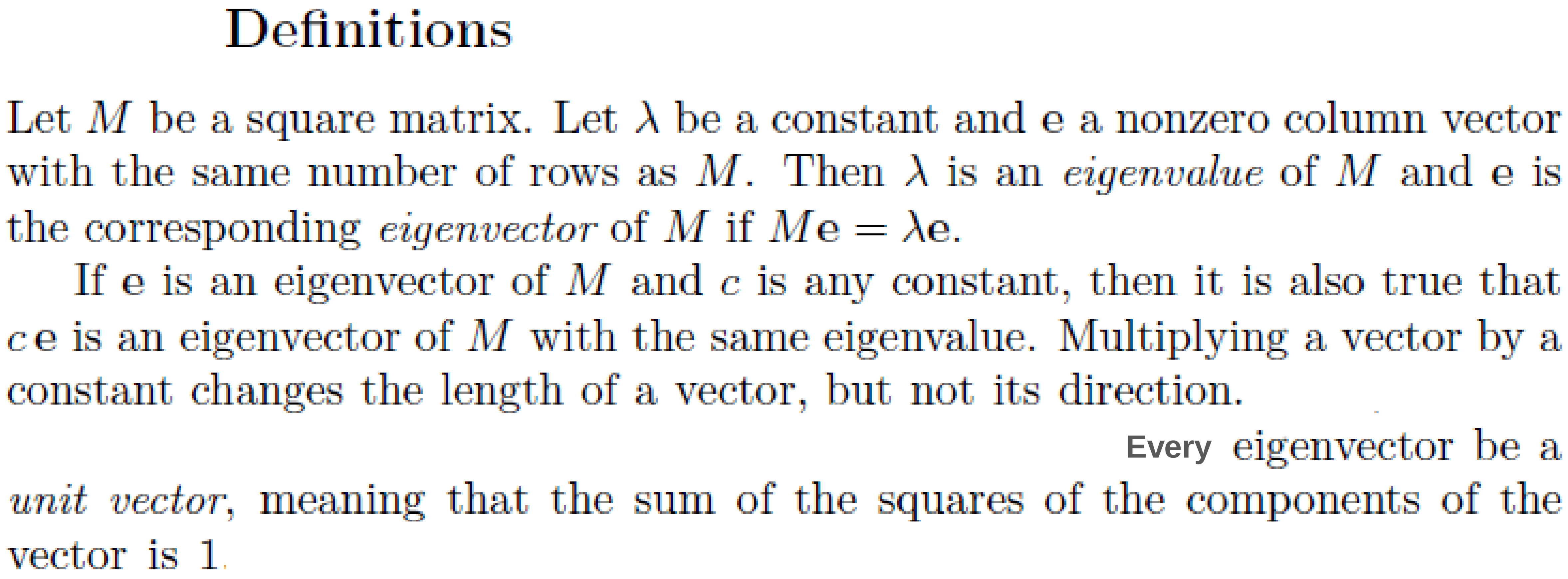


Figure 11.1: Four points in a two-dimensional space

- The data is two-dimensional, a number of dimensions that is too small to make PCA really useful.
- Moreover, the data, shown in Fig. 11.1 has only four points, and they are arranged in a simple pattern along the 45-degree line to make our calculations easy to follow.
- The points can best be viewed as lying along the axis that is at a 45-degree angle, with small deviations in the perpendicular direction.

Eigenvalue and Eigenvector

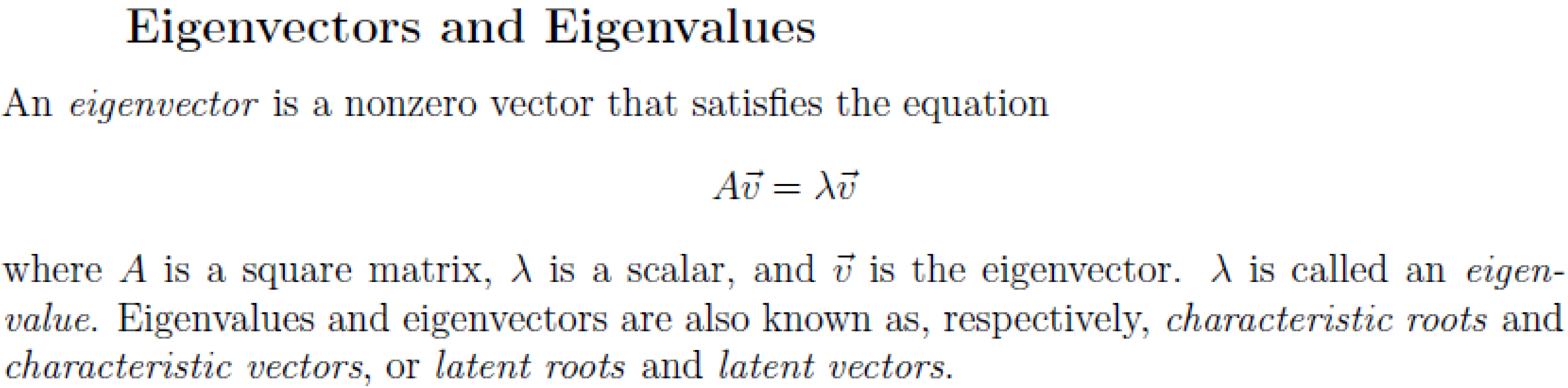


Definitions

Let M be a square matrix. Let λ be a constant and $\mathbf{e}$ a nonzero column vector with the same number of rows as M. Then λ is an *eigenvalue* of M and $\mathbf{e}$ is the corresponding *eigenvector* of M if $M\mathbf{e} = \lambda\mathbf{e}$.

If $\mathbf{e}$ is an eigenvector of M and c is any constant, then it is also true that $c\,\mathbf{e}$ is an eigenvector of M with the same eigenvalue. Multiplying a vector by a constant changes the length of a vector, but not its direction.

Every eigenvector be a *unit vector*, meaning that the sum of the squares of the components of the vector is 1.



Eigenvectors and Eigenvalues

An *eigenvector* is a nonzero vector that satisfies the equation

$$A\vec{v} = \lambda\vec{v}$$

where A is a square matrix, λ is a scalar, and $\vec{v}$ is the eigenvector. λ is called an *eigenvalue*. Eigenvalues and eigenvectors are also known as, respectively, *characteristic roots* and *characteristic vectors*, or *latent roots* and *latent vectors*.

- We can find eigenvalues and eigenvectors by treating a matrix as a system of linear equations, and
- Solving for the values of the variables that make up the components of the eigenvector

For example, finding the eigenvalues and corresponding eigenvectors of the matrix

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

means applying the above formula to get

$$A\vec{v} = \lambda\vec{v} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

in order to solve for λ, x_1 and x_2. This statement is equivalent to the system of equations

$$2x_1 + x_2 = \lambda x_1$$

$$x_1 + 2x_2 = \lambda x_2$$

which can be rearranged as

$$(2 - \lambda)x_1 + x_2 = 0$$

$$x_1 + (2 - \lambda)x_2 = 0$$

A necessary and sufficient condition for this system to have a nonzero vector $[x_1, x_2]$ is that the determinant of the coefficient matrix

$$\begin{bmatrix} (2-\lambda) & 1 \\ 1 & (2-\lambda) \end{bmatrix}$$

be equal to zero. Accordingly,

$$\begin{vmatrix} (2-\lambda) & 1 \\ 1 & (2-\lambda) \end{vmatrix} = 0$$

$$(2-\lambda)(2-\lambda) - 1 \cdot 1 = 0$$

$$\lambda^2 - 4\lambda + 3 = 0$$

$$(\lambda - 3)(\lambda - 1) = 0$$

There are two values of λ that satisfy the last equation; thus there are two eigenvalues of the original matrix A and these are $\lambda_1 = 3, \lambda_2 = 1$.

We can find eigenvectors which correspond to these eigenvalues by plugging λ back in to the equations above and solving for x_1 and x_2. To find an eigenvector corresponding to $\lambda = 3$, start with

$$(2 - \lambda)x_1 + x_2 = 0$$

and substitute to get

$$(2 - 3)x_1 + x_2 = 0$$

which reduces and rearranges to

$$x_1 = x_2$$

There are an infinite number of values for x_1 which satisfy this equation; the only restriction is that not all the components in an eigenvector can equal zero. So if $x_1 = 1$, then $x_2 = 1$ and an eigenvector corresponding to $\lambda = 3$ is $[1, 1]$.

Finding an eigenvector for $\lambda = 1$ works the same way.

$$(2 - 1)x_1 + x_2 = 0$$

$$x_1 = -x_2$$

So an eigenvector for $\lambda = 1$ is $[1, -1]$.

Example 11.2: Let us find the eigenpairs for the 2×2 matrix M from Example 11.1. Recall $M =$

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$$

Then $M - \lambda I$ is

$$\begin{bmatrix} 3-\lambda & 2 \\ 2 & 6-\lambda \end{bmatrix}$$

The determinant of this matrix is $(3-\lambda)(6-\lambda) - 4$, which we must set to 0. The equation in λ to solve is thus $\lambda^2 - 9\lambda + 14 = 0$. The roots of this equation are $\lambda = 7$ and $\lambda = 2$; the first is the principal eigenvalue, since it is the larger. Let $\mathbf{e}$ be the vector of unknowns

$$\begin{bmatrix} x \\ y \end{bmatrix}$$

We must solve

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 7 \begin{bmatrix} x \\ y \end{bmatrix}$$

When we multiply the matrix and vector we get two equations

$$\begin{array}{rcl} 3\text{x}+2\text{y} & = & 7\text{x} \\ 2\text{x}+6\text{y} & = & 7\text{y} \end{array}$$

Notice that both of these equations really say the same thing: $y = 2x$. Thus, a possible eigenvector is

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

But that vector is not a unit vector, since the sum of the squares of its components is 5, not 1. Thus to get the unit vector in the same direction, we divide each component by $\sqrt{5}$. That is, the principal eigenvector is

$$\begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$$

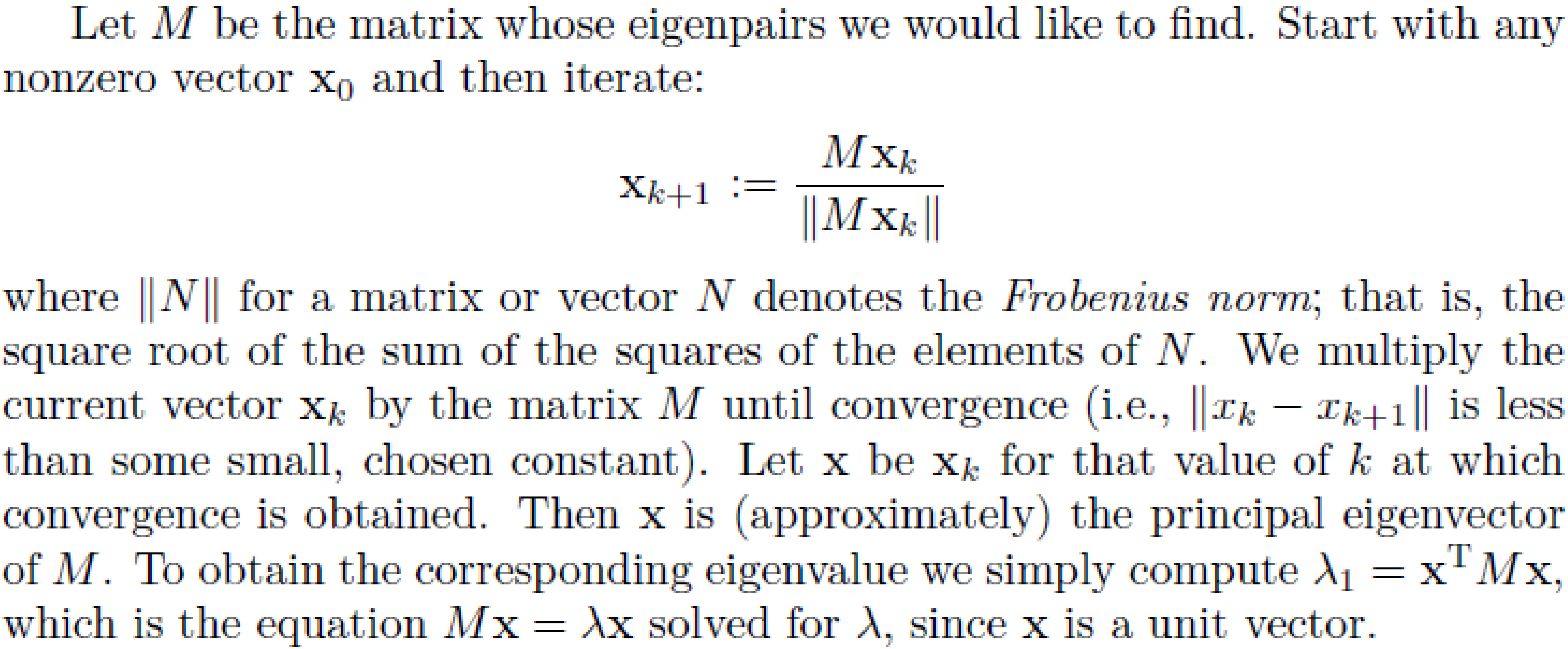
and its eigenvalue is 7. Note that this was the eigenpair we explored in Example 11.1.

For the second eigenpair, we repeat the above with eigenvalue 2 in place of 7. The equation involving the components of $\mathbf{e}$ is $x = -2y$, and the second eigenvector is

$$\begin{bmatrix} 2/\sqrt{5} \\ -1/\sqrt{5} \end{bmatrix}$$

Its corresponding eigenvalue is 2, of course. $\square$

Finding Eigenpairs by Power Iteration



Let M be the matrix whose eigenpairs we would like to find. Start with any nonzero vector $\mathbf{x}_0$ and then iterate:

$$\mathbf{x}_{k+1} := \frac{M\mathbf{x}_k}{\|M\mathbf{x}_k\|}$$

where $\|N\|$ for a matrix or vector N denotes the *Frobenius norm*; that is, the square root of the sum of the squares of the elements of N. We multiply the current vector $\mathbf{x}_k$ by the matrix M until convergence (i.e., $\|x_k - x_{k+1}\|$ is less than some small, chosen constant). Let $\mathbf{x}$ be $\mathbf{x}_k$ for that value of k at which convergence is obtained. Then $\mathbf{x}$ is (approximately) the principal eigenvector of M. To obtain the corresponding eigenvalue we simply compute $\lambda_1 = \mathbf{x}^{\mathrm{T}}M\mathbf{x}$, which is the equation $M\mathbf{x} = \lambda\mathbf{x}$ solved for λ, since $\mathbf{x}$ is a unit vector.

Example 11.3: Take the matrix from Example 11.2:

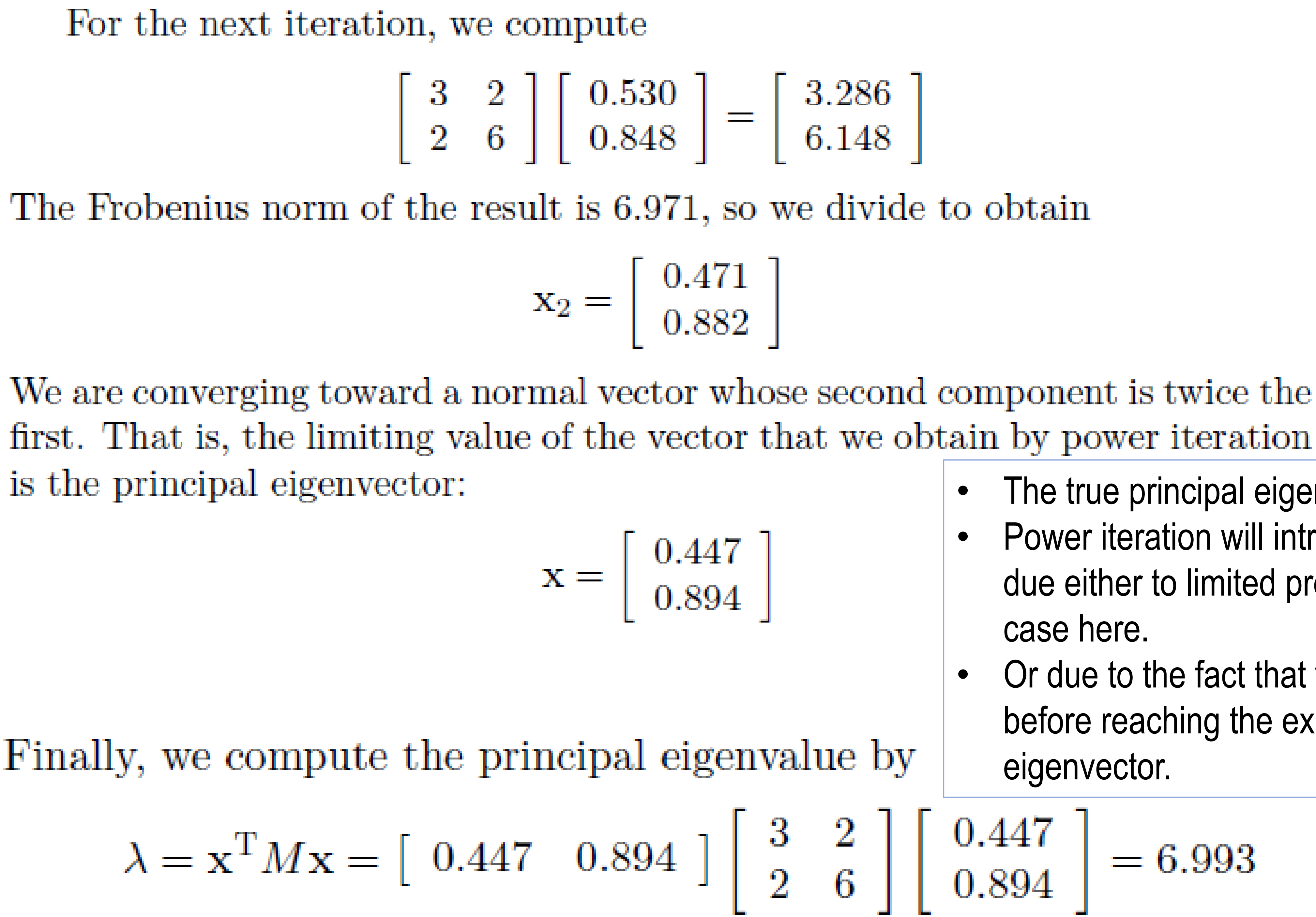
$$M = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix}$$

and let us start with $\mathbf{x}_0$ a vector with 1 for both components. To compute $\mathbf{x}_1$, we multiply $M\mathbf{x}_0$ to get

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}$$

The Frobenius norm of the result is $\sqrt{5^2 + 8^2} = \sqrt{89} = 9.434$. We obtain $\mathbf{x}_1$ by dividing 5 and 8 by 9.434; that is:

$$\mathbf{x}_1 = \begin{bmatrix} 0.530 \\ 0.848 \end{bmatrix}$$



For the next iteration, we compute

$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 0.530 \\ 0.848 \end{bmatrix} = \begin{bmatrix} 3.286 \\ 6.148 \end{bmatrix}$$

The Frobenius norm of the result is 6.971, so we divide to obtain

$$\mathbf{x}_2 = \begin{bmatrix} 0.471 \\ 0.882 \end{bmatrix}$$

We are converging toward a normal vector whose second component is twice the first. That is, the limiting value of the vector that we obtain by power iteration is the principal eigenvector:

$$\mathbf{x} = \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix}$$

Finally, we compute the principal eigenvalue by

$$\lambda = \mathbf{x}^{\mathrm{T}} M \mathbf{x} = \begin{bmatrix} 0.447 & 0.894 \end{bmatrix} \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} \begin{bmatrix} 0.447 \\ 0.894 \end{bmatrix} = 6.993$$

- The true principal eigenvalue is 7.
- Power iteration will introduce small errors due either to limited precision, as was the case here.
- Or due to the fact that we stop the iteration before reaching the exact value of the eigenvector.

11.1.4 The Matrix of Eigenvectors

Suppose we have an $n \times n$ symmetric matrix M whose eigenvectors, viewed as column vectors, are $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n$. Let E be the matrix whose ith column is $\mathbf{e}_i$. Then $EE^{\mathrm{T}} = E^{\mathrm{T}}E = I$. The explanation is that the eigenvectors of a symmetric matrix are *orthonormal*. That is, they are orthogonal unit vectors.

Example 11.5: For the matrix M of Example 11.2, the matrix E is

$$\begin{bmatrix} 2/\sqrt{5} & 1/\sqrt{5} \\ -1/\sqrt{5} & 2/\sqrt{5} \end{bmatrix}$$

E^{T} is therefore

$$\begin{bmatrix} 2/\sqrt{5} & -1/\sqrt{5} \\ 1/\sqrt{5} & 2/\sqrt{5} \end{bmatrix}$$

When we compute EE^{T} we get

$$\begin{bmatrix} 4/5 + 1/5 & -2/5 + 2/5 \\ -2/5 + 2/5 & 1/5 + 4/5 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

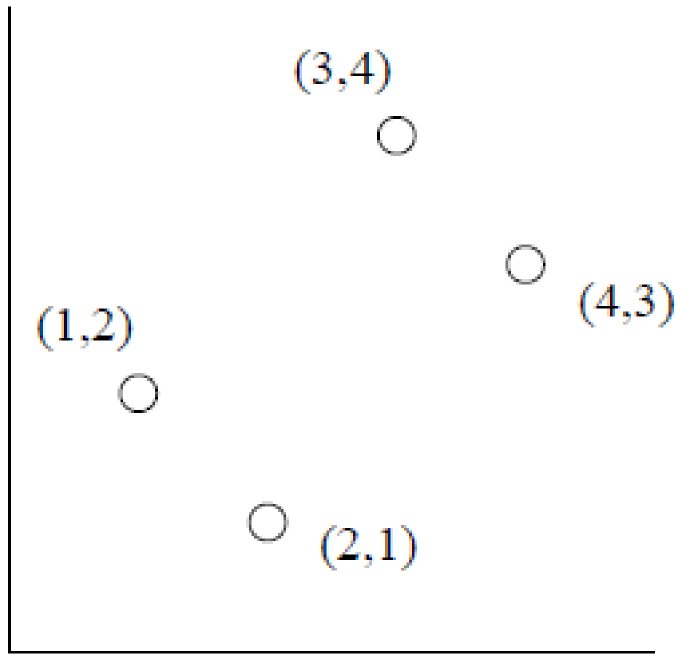


Figure 11.1: Four points in a two-dimensional space

To begin, let us represent the points by a matrix M with four rows – one for each point – and two columns, corresponding to the x-axis and y-axis. This matrix is

$$M = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix}$$

Compute $M^{\mathrm{T}}M$, which is

$$M^{\mathrm{T}}M = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} = \begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix}$$

We may find the eigenvalues of the matrix above by solving the equation

$$(30 - \lambda)(30 - \lambda) - 28 \times 28 = 0$$

as we did in Example 11.2. The solution is $\lambda = 58$ and $\lambda = 2$.

Following the same procedure as in Example 11.2, we must solve

$$\begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 58 \begin{bmatrix} x \\ y \end{bmatrix}$$

When we multiply out the matrix and vector we get two equations

$$\begin{array}{rcl} \text{30x+28y} & = & \text{58x} \\ \text{28x+30y} & = & \text{58y} \end{array}$$

Both equations tell us the same thing: $x = y$. Thus, the unit eigenvector corresponding to the principal eigenvalue 58 is

$$\begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

For the second eigenvalue, 2, we perform the same process. Multiply out

$$\begin{bmatrix} 30 & 28 \\ 28 & 30 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 2 \begin{bmatrix} x \\ y \end{bmatrix}$$

to get the two equations

$$\begin{aligned} \text{30x+28y} &= \text{2x} \\ \text{28x+30y} &= \text{2y} \end{aligned}$$

Both equations tell us the same thing: $x = -y$. Thus, the unit eigenvector corresponding to the principal eigenvalue 2 is

$$\begin{bmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

While we promised to write eigenvectors with their first component positive, we choose the opposite here because it makes the transformation of coordinates easier to follow in this case.

Now, let us construct E, the matrix of eigenvectors for the matrix $M^{\mathrm{T}}M$. Placing the principal eigenvector first, the matrix of eigenvectors is

$$E = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

Any matrix of *orthonormal vectors* (unit vectors that are orthogonal to one another) represents a rotation of the axes of a Euclidean space. The matrix above can be viewed as a rotation 45 degrees counterclockwise. For example, let us multiply the matrix M that represents each of the points of Fig. 11.1 by E. The product is

$$ME = \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 3/\sqrt{2} & 1/\sqrt{2} \\ 3/\sqrt{2} & -1/\sqrt{2} \\ 7/\sqrt{2} & 1/\sqrt{2} \\ 7/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

Figure 11.3 shows the four points in the rotated coordinate system.

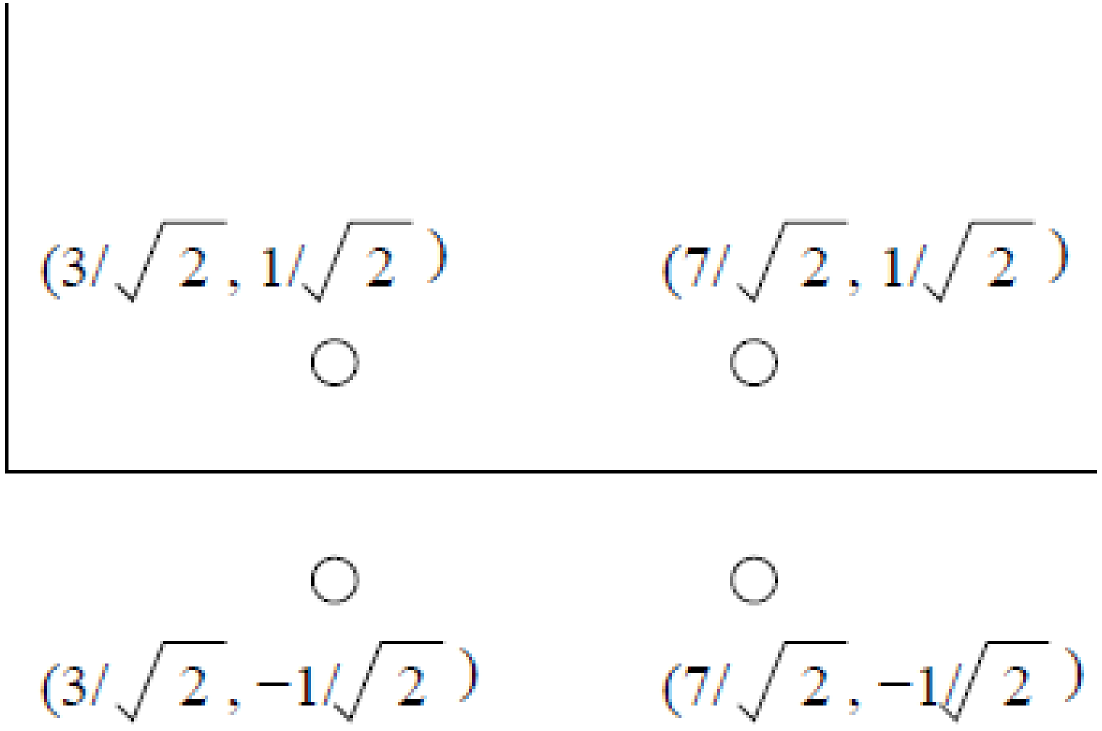


Figure 11.3: The points of Fig. 11.1 in the new coordinate system

11.2.2 Using Eigenvectors for Dimensionality Reduction

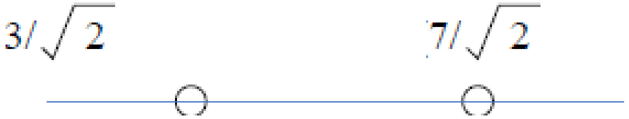
If M is a matrix whose rows each represent a point in a Euclidean space with any number of dimensions, we can compute $M^{\mathrm{T}}M$ and compute its eigenpairs. Let E be the matrix whose columns are the eigenvectors, ordered as largest eigenvalue first. Define the matrix L to have the eigenvalues of $M^{\mathrm{T}}M$ along the diagonal, largest first, and 0's in all other entries. Then, since $M^{\mathrm{T}}M\mathbf{e} = \lambda\mathbf{e} = \mathbf{e}\lambda$ for each eigenvector $\mathbf{e}$ and its corresponding eigenvalue λ, it follows that $M^{\mathrm{T}}ME = EL$.

We observed that ME is the points of M transformed into a new coordinate space. In this space, the first axis (the one corresponding to the largest eigenvalue) is the most significant; formally, the variance of points along that axis is the greatest. The second axis, corresponding to the second eigenpair, is next most significant in the same sense, and the pattern continues for each of the eigenpairs. If we want to transform M to a space with fewer dimensions, then the choice that preserves the most significance is the one that uses the eigenvectors associated with the largest eigenvalues and ignores the other eigenvalues.

That is, let E_k be the first k columns of E. Then ME_k is a k-dimensional representation of M.

Example 11.6: Let M be the matrix from Section 11.2.1. This data has only two dimensions, so the only dimensionality reduction we can do is to use $k = 1$; i.e., project the data onto a one dimensional space. That is, we compute ME_1 by

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 3 & 4 \\ 4 & 3 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 3/\sqrt{2} \\ 3/\sqrt{2} \\ 7/\sqrt{2} \\ 7/\sqrt{2} \end{bmatrix}$$



The effect of this transformation is to replace the points of M by their projections onto the x-axis of Fig. 11.3. While the first two points project to the same point, as do the third and fourth points, this representation makes the best possible one-dimensional distinctions among the points. □

Singular Value Decomposition (SVD)

- SVD can be looked at from three mutually compatible points of view.

- We can see it as a method for transforming correlated variables into a set of uncorrelated ones that better expose the various relationships among the original data items.
- At the same time, SVD is a method for identifying and ordering the dimensions along which data points exhibit the most variation.
- Once we have identified where the most variation is, it's possible to find the best approximation of the original data points using fewer dimensions.
- Hence, SVD can be seen as a method for data reduction.

Singular Value Decomposition (SVD)

- Singular value decomposition (SVD), allows an exact representation of any matrix, and also makes it easy to eliminate the less important parts of that representation to produce an approximate representation with any desired number of dimensions
- Of course the fewer the dimensions we choose, the less accurate will be the approximation

11.3.1 Definition of SVD

Let M be an $m \times n$ matrix, and let the rank of M be r. Recall that the *rank* of a matrix is the largest number of rows (or equivalently columns) we can choose for which no nonzero linear combination of the rows is the all-zero vector $\mathbf{0}$ (we say a set of such rows or columns is *independent*). Then we can find matrices U, Σ, and V as shown in Fig. 11.5 with the following properties:

1. U is an $m \times r$ *column-orthonormal matrix*; that is, each of its columns is a unit vector and the dot product of any two columns is 0.

2. V is an $n \times r$ column-orthonormal matrix. Note that we always use V in its transposed form, so it is the rows of V^{T} that are orthonormal.

3. Σ is a diagonal matrix; that is, all elements not on the main diagonal are 0. The elements of Σ are called the *singular values* of M.

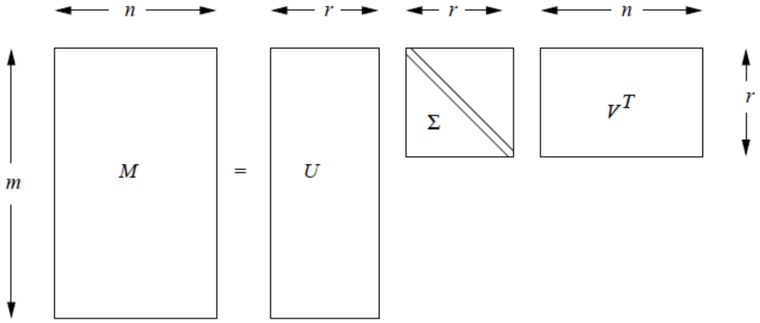


Figure 11.5: The form of a singular-value decomposition

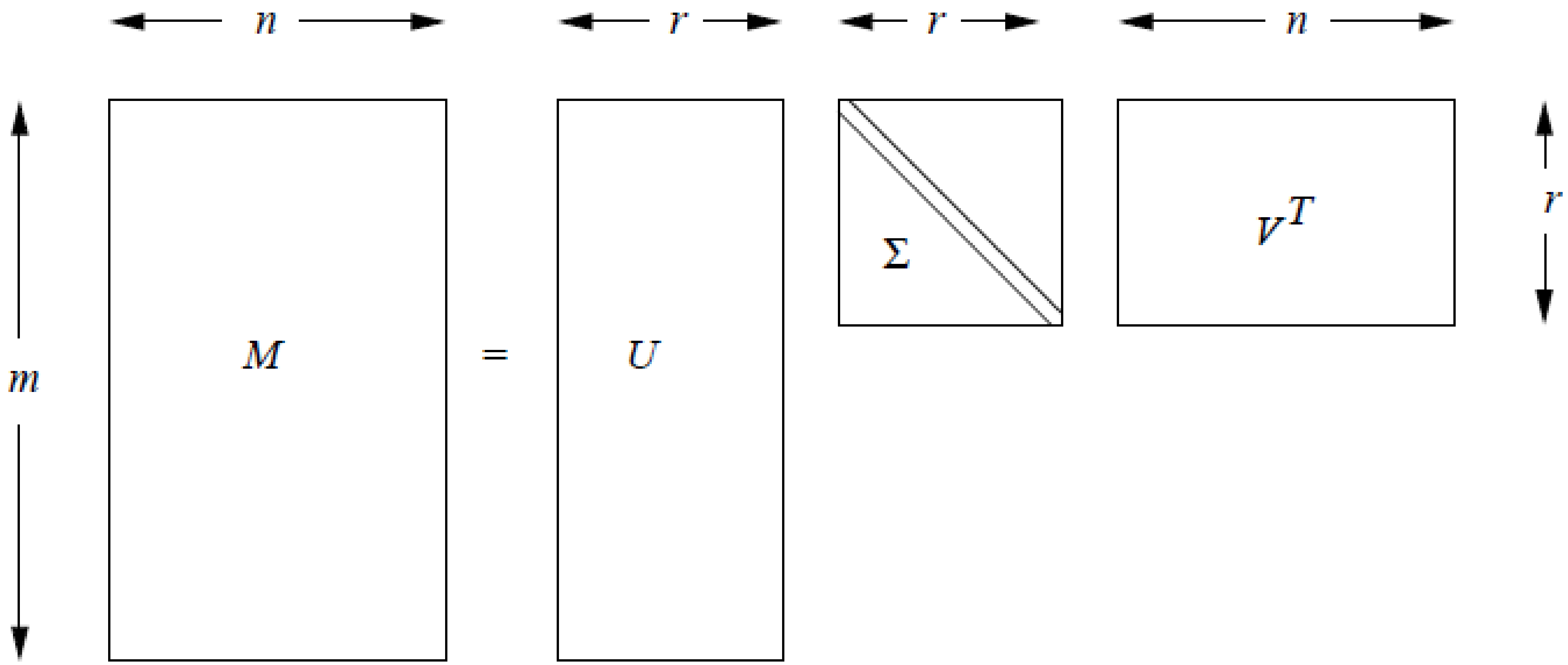


Figure 11.5: The form of a singular-value decomposition

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

Figure 11.6: Ratings of movies by users

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix}}_{M} = \underbrace{\begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}}_{V^{\mathrm{T}}}$$

Figure 11.7: SVD for the matrix M of Fig. 11.6

Interpretation of SVD

- SVD offers is in viewing the r columns of U, Σ, and V as representing concepts that are hidden in the original matrix M
- In Figure 11.6, these concepts are clear; one is “science fiction” and the other is “romance”
- Let us think of the rows of M as people and the columns of M as movies
  - Matrix U connects people to concepts
  - The matrix V relates movies to concepts
  - The matrix Σ gives the strength of each of the concepts.
- In our example, the strength of the science-fiction concept is 12.4, while the strengthof the romance concept is 9.5.

$$\underset{M'}{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}} =$$

How U, Σ and V^T are calculated, please see SVD tutorial 1 and 2 in my uploaded files at slate.

$$\underset{U}{\begin{bmatrix} .13 & .02 & -.01 \\ .41 & .07 & -.03 \\ .55 & .09 & -.04 \\ .68 & .11 & -.05 \\ .15 & -.59 & .65 \\ .07 & -.73 & -.67 \\ .07 & -.29 & .32 \end{bmatrix}} \underset{\Sigma}{\begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix}} \underset{V^{\mathrm{T}}}{\begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \\ .40 & -.80 & .40 & .09 & .09 \end{bmatrix}}$$

Dimensionality Reduction Using SVD

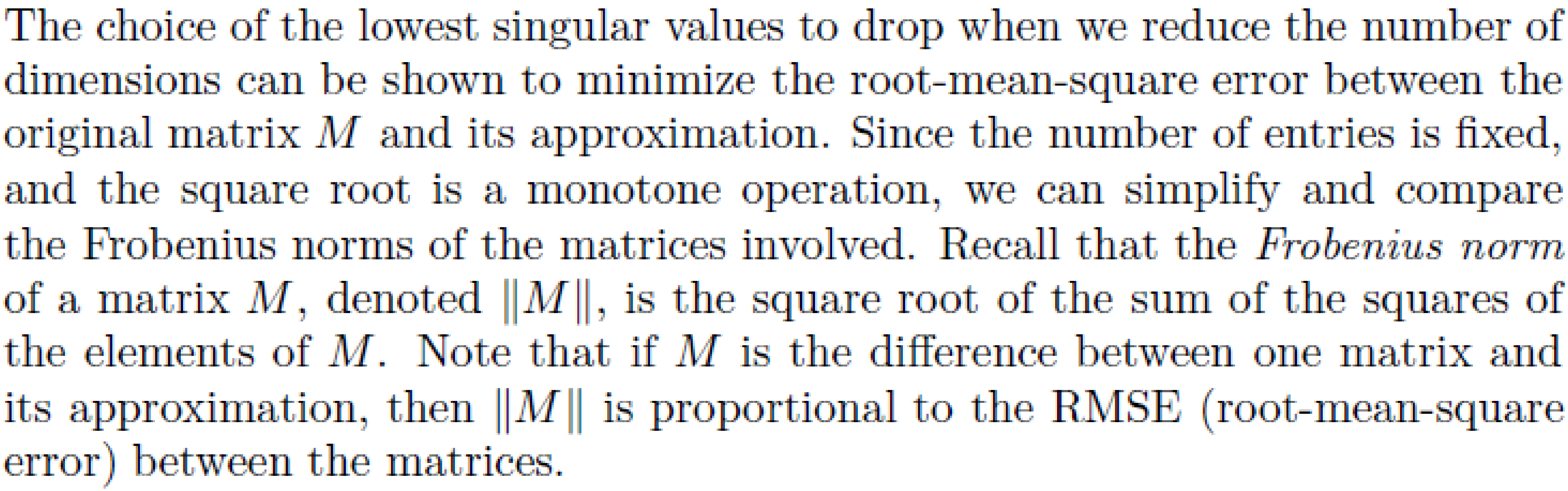
- Suppose we want to represent a very large matrix M by its SVD components U, Σ, and V , but these matrices are also too large to store conveniently
- The best way to reduce the dimensionality of the three matrices is to set the smallest of the singular values to zero
- If we set the s smallest singular values to 0, then we can also eliminate the corresponding s columns of U and V .
- Set the smallest of the singular values, which is 1.3, to zero in Example 11.9, the effect is:
- the third column of U and the third row of V^T are multiplied only by 0’s and the U’s row and V^T’s column may be eliminated, as shown in Fig. 11.10.

$$
\begin{bmatrix}
.13 & .02 \\
.41 & .07 \\
.55 & .09 \\
.68 & .11 \\
.15 & -.59 \\
.07 & -.73 \\
.07 & -.29
\end{bmatrix}
\begin{bmatrix}
12.4 & 0 \\
0 & 9.5
\end{bmatrix}
\begin{bmatrix}
.56 & .59 & .56 & .09 & .09 \\
.12 & -.02 & .12 & -.69 & -.69
\end{bmatrix}
$$

$$
= \begin{bmatrix}
0.93 & 0.95 & 0.93 & .014 & .014 \\
2.93 & 2.99 & 2.93 & .000 & .000 \\
3.92 & 4.01 & 3.92 & .026 & .026 \\
4.84 & 4.96 & 4.84 & .040 & .040 \\
0.37 & 1.21 & 0.37 & 4.04 & 4.04 \\
0.35 & 0.65 & 0.35 & 4.87 & 4.87 \\
0.16 & 0.57 & 0.16 & 1.98 & 1.98
\end{bmatrix}
$$

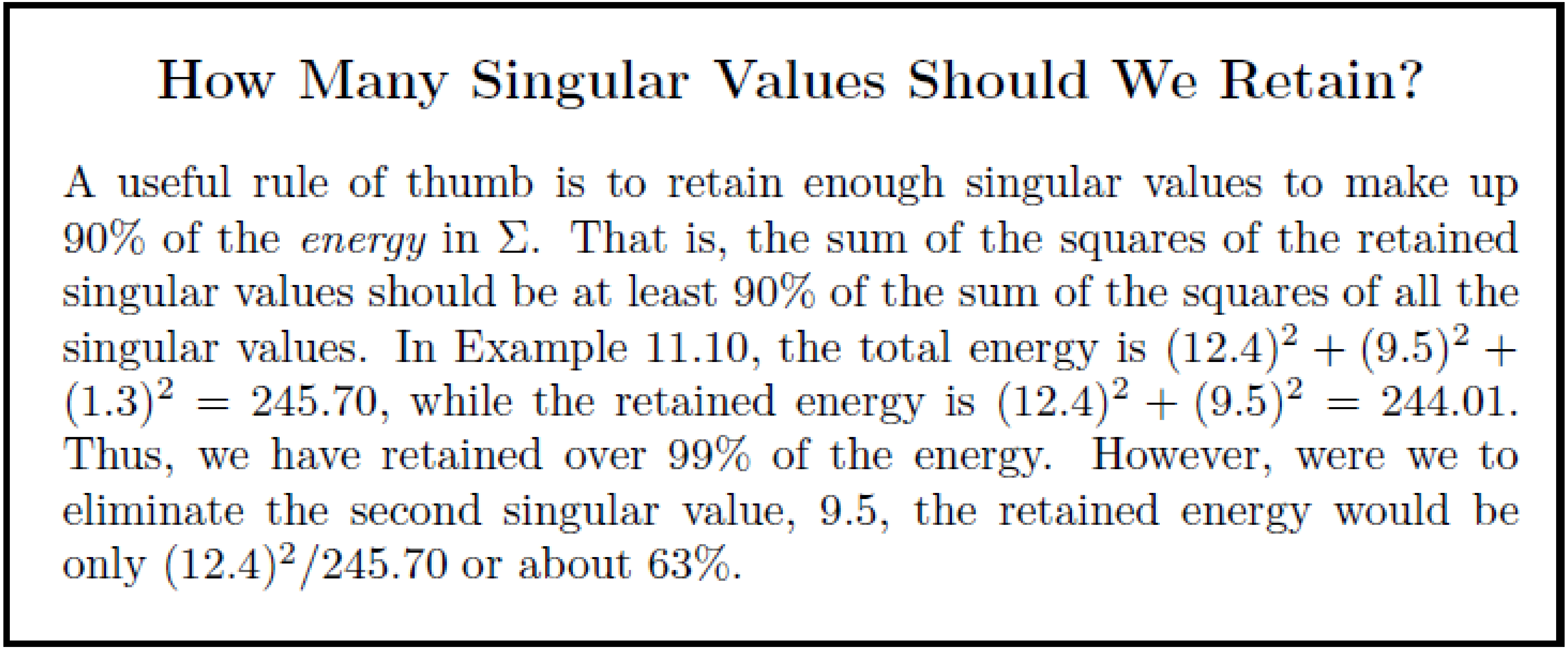
Figure 11.10: Dropping the lowest singular value from the decomposition of Fig. 11.7

Why Zeroing Low Singular Values Works



The choice of the lowest singular values to drop when we reduce the number of dimensions can be shown to minimize the root-mean-square error between the original matrix M and its approximation. Since the number of entries is fixed, and the square root is a monotone operation, we can simplify and compare the Frobenius norms of the matrices involved. Recall that the *Frobenius norm* of a matrix M, denoted $\|M\|$, is the square root of the sum of the squares of the elements of M. Note that if M is the difference between one matrix and its approximation, then $\|M\|$ is proportional to the RMSE (root-mean-square error) between the matrices.

$$\begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \longrightarrow \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}$$



How Many Singular Values Should We Retain?

A useful rule of thumb is to retain enough singular values to make up 90% of the *energy* in Σ. That is, the sum of the squares of the retained singular values should be at least 90% of the sum of the squares of all the singular values. In Example 11.10, the total energy is $(12.4)^2 + (9.5)^2 + (1.3)^2 = 245.70$, while the retained energy is $(12.4)^2 + (9.5)^2 = 244.01$. Thus, we have retained over 99% of the energy. However, were we to eliminate the second singular value, 9.5, the retained energy would be only $(12.4)^2/245.70$ or about 63%.

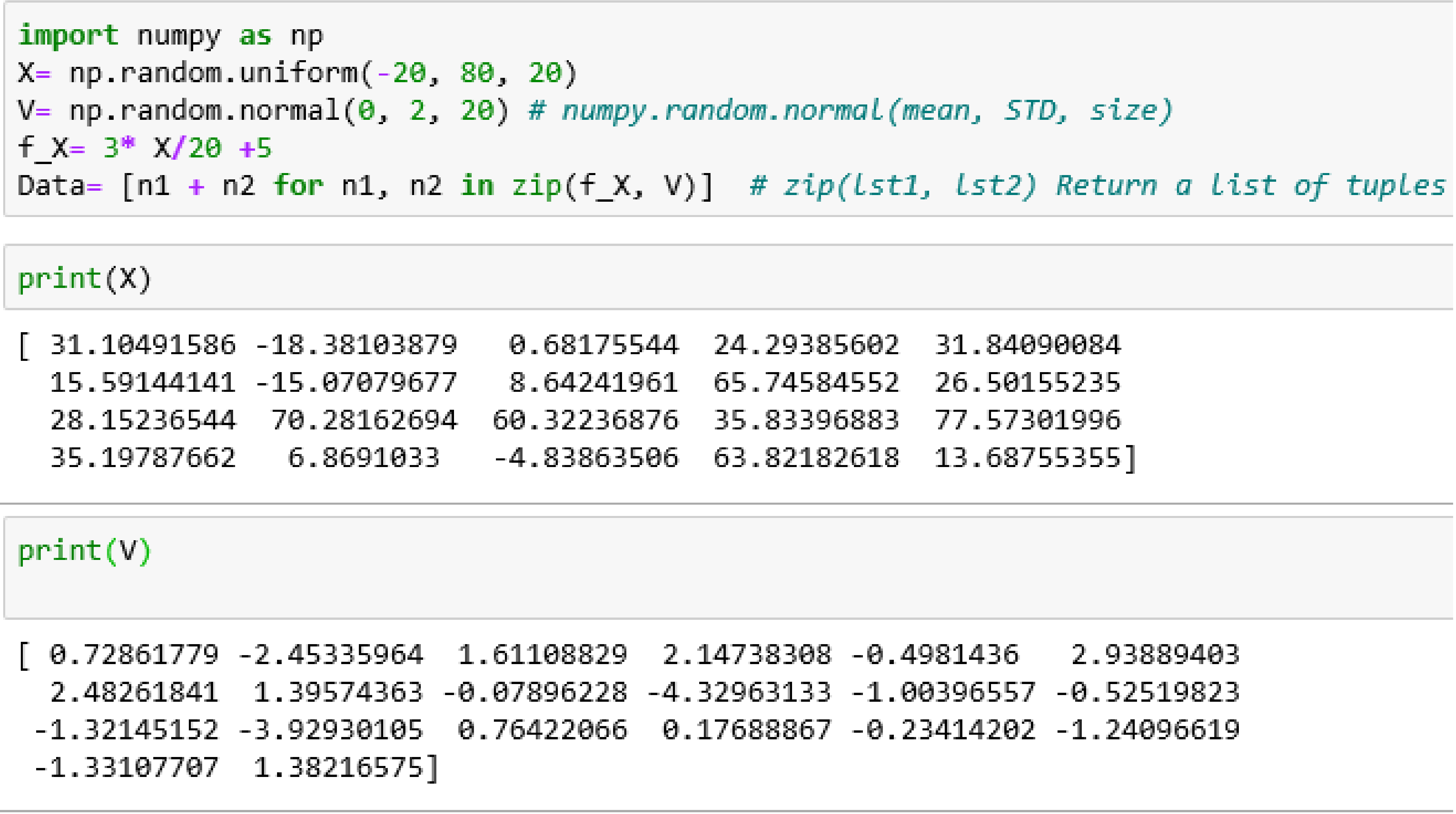
SVD Implementation code using linalg

```
U, s, VT = svd(A)
```

where A is a matrix

```
u, s, v = np.linalg.svd(lifeExp_AE_scaled, full_matrices=True)
```

- “full_matrices=True” to get all singular vectors
- The output of SVD is three matrices, u, s, and v
- The matrices u and v are singular vectors and s is singular values
- Singular values help us compute variance explained by each singular vectors



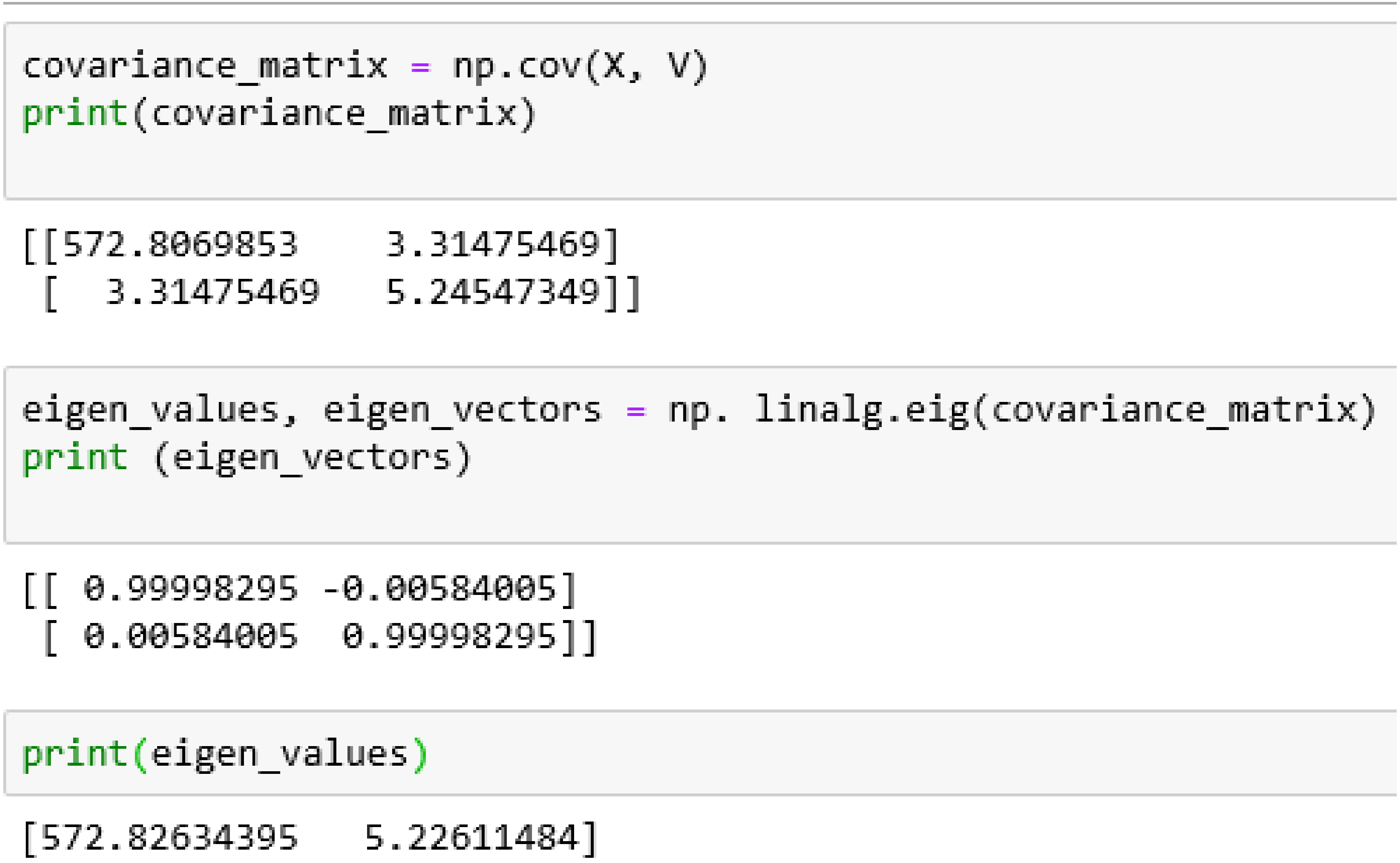
```
import numpy as np
X= np.random.uniform(-20, 80, 20)
V= np.random.normal(0, 2, 20) # numpy.random.normal(mean, STD, size)
f_X= 3* X/20 +5
Data= [n1 + n2 for n1, n2 in zip(f_X, V)]  # zip(lst1, lst2) Return a list of tuples
```

```
print(X)
```

```
[ 31.10491586 -18.38103879   0.68175544  24.29385602  31.84090084
  15.59144141 -15.07079677   8.64241961  65.74584552  26.50155235
  28.15236544  70.28162694  60.32236876  35.83396883  77.57301996
  35.19787662   6.8691033   -4.83863506  63.82182618  13.68755355]
```

```
print(V)
```

```
[ 0.72861779 -2.45335964  1.61108829  2.14738308 -0.4981436   2.93889403
  2.48261841  1.39574363 -0.07896228 -4.32963133 -1.00396557 -0.52519823
 -1.32145152 -3.92930105  0.76422066  0.17688867 -0.23414202 -1.24096619
 -1.33107707  1.38216575]
```



```
covariance_matrix = np.cov(X, V)
print(covariance_matrix)
```

```
[[572.8069853    3.31475469]
 [  3.31475469   5.24547349]]
```

```
eigen_values, eigen_vectors = np. linalg.eig(covariance_matrix)
print (eigen_vectors)
```

```
[[ 0.99998295 -0.00584005]
 [ 0.00584005  0.99998295]]
```

```
print(eigen_values)
```

```
[572.82634395   5.22611484]
```