One of the most useful results from linear algebra, is a matrix decomposition known as the singular value decomposition. It has many useful applications in almost every scientific discipline. For example, it is used as an important tool for data compression.

To get the singular value decomposition of a matrix $A$, we take advantage of the fact that both $A^*A$ and $AA^*$ are hermitian ($A^tA$ and $AA^t$ are symmetric) with non-negative eigenvalues. Hermitian and symmetric matrices have the nice property that their eigenvectors form an orthonormal basis.

For any $m \times n$ real or complex matrix $A = (a_{ij})$, there exist a factorization of the form

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

called the singular value decomposition (SVD). The $m \times m$ unitary matrix $U$ consists of orthonormal eigenvectors associated with the eigenvalues of $AA^*$ and the $n \times n$ unitary matrix $V$ consists of the orthonormal eigenvectors of $A^*A$. The diagonal elements of $\Sigma$ are called singular values of $A$. The matrix $\Sigma = (\sigma_{ij})$ is an $m \times n$ matrix, where all the entries are zero, except the diagonal entries $\sigma_{ii}$ (or just $\sigma_i$), which are the square roots of the eigenvalues of $AA^*$ and/or $A^*A$. Although there are many ways to choose $U$ and $V$ from the eigenvectors of $AA^*$ and/or $A^*A$, but it is customary to choose these two matrices in a way that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m > 0.$$ 

To solve a system of linear equations, where its matrix is either singular or rectangular, one must use the SVD in order to be able to find solutions, if any.

**Truncated SVD.** The sizes of the matrices in the SVD are as follows: $U$ is $m \times m$, $\Sigma$ is $m \times n$, and $V$ is $n \times n$. Thus, $\Sigma$ has the same size as $A$, while $U$ and $V$ are square. However, if $m > n$, the bottom $(m - n) \times n$ block of $\Sigma$ is zero, so that the last $m - n$ columns of $U$ are multiplied by zero. Similarly, if $m < n$, the rightmost $m \times (n - m)$ block of $\Sigma$ is zero, and this multiplies the last $n - m$ rows of $V$.

This suggests a "small," equivalent version of the SVD. If $p = \min(m, n)$, we can define

$$U_p = U(:, 1 : p), \quad \Sigma_p = \Sigma(1 : p, 1 : p), \quad \text{and} \quad V_p = V(:, 1 : p),$$

and write

$$A_p = U_p \Sigma_p V_p^*,$$

where $U_p$ is $m \times p$, $\Sigma_p$ is $p \times p$, and $V_p$ is $n \times p$. Moreover, if $p - r$ singular values are zero, we can let

$$U_r = U(:, 1 : r), \quad \Sigma_r = \Sigma(1 : r, 1 : r), \quad \text{and} \quad V_r = V(:, 1 : r),$$
and write

\[ A_r = U_r \Sigma_r V_r^*. \]

Then we have \( A_r = U_r \Sigma_r V_r^* \) which is an even smaller, SVD.

It is often the case that the matrix \( A \) has one dimension much bigger than the other. For instance, \( m = 3 \) and \( n = 10,000 \) might be such a case. For such examples, much of the computation and memory required for the standard SVD may not actually be needed. Instead, a truncated, or reduced version is appropriate \( ( A_r = U_r \Sigma_r V_r^* ) \). It will be computed faster, and require less memory to store the data.

**Definition 1.** The 2-norm and the Frobenius norm of an \( m \times n \) matrix \( A = (a_{ij}) \) can be easily computed from the SVD decomposition.

\[
\| A \|_2 = \sigma_1; \quad \| A \|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_{p}^2}; \quad p = \min (m, n). 
\]

**Theorem (Eckart-Young-Mirsky, 1936).** If \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m \) are the nonzero singular values of the \( m \times n \) matrix \( A \), then for each \( k \), the distance from \( A \) to the closest rank-\( k \) matrix \( B_k \) is

\[
\sigma_{k+1} = \min_{B_k} \| A - B_k \|_F. 
\]

Thus, \( \sigma_{k+1} \) is a measure of how well \( A_k \) approximates \( A \). In fact, for \( A \) of rank \( r \) \( (k < r) \), the truncated matrix \( A_k \) is the best rank-\( k \) matrix approximation to \( A \).

**Proof.** For any rank \( k \) matrix \( B_k \) of the same size as \( A \), define the matrix \( C = (c_{ij}) = U^T B_k V \), so \( B_k = U C V^T \). Then

\[
\| A - B \|_F^2 = \| U \Sigma V^T - U C V^T \|_F^2 = \| \Sigma - C \|_F^2 = \sum_i |\sigma_i - c_{ii}|^2 + \sum_{i \neq j} |c_{ij}|^2 \geq \sum_i |\sigma_i - c_{ii}|^2.
\]

Thus clearly the rank \( k \) choice of \( C \) that minimizes \( \| A - B \|_F^2 \) is when \( C \) is a diagonal matrix with \( (\sigma_1, \sigma_2, \cdots, \sigma_k) \) as its nonzero diagonal elements. The corresponding \( B_k \) matches exactly the truncated matrix \( A_k \).

**Definition 2.** The condition number associated with the linear equation \( Ax = b \) gives a bound on how inaccurate the solution \( x \) will be after approximation. Note that this is before the effects of round-off error are taken into account; conditioning is a property of the matrix, not the algorithm or floating point accuracy of the computer used to solve the corresponding system. In particular, one should think of the condition number as being (very roughly) the rate at which the solution, \( x \), will change with respect to a change in \( b \). Thus, if the condition number is large, even a small error in \( b \) may cause a large error in \( x \). On the other hand, if the condition number is small then the error in \( x \) will not be much bigger than the error in \( b \). The condition number is defined more precisely to be the maximum ratio of the relative error in \( x \) divided by the relative error in \( b \).

The condition number of a matrix is the ratio of its largest singular value to its smallest singular value.

If one of the \( \sigma \)'s is zero, or so small that its value is dominated by round-off error, then there is a problem!

**Definition 3.** A pseudo-inverse, also known as generalized inverse, or Moore-Penrose inverse, denoted by \( A^\dagger \) or \( A^+ \), is a generalization of the inverse matrix. All matrices, including rectangular or singular
matrices have unique pseudo-inverse.

For any \( m \times n \) real or complex matrix \( A \), a pseudo-inverse of \( A \) is defined as an \( n \times m \) matrix \( A^+ \) satisfying all of the following four criteria:

\[
AA^+ = A \quad \text{(} AA^+ \text{ maps all column vectors of } A \text{ to themselves);}
\]

\[
A^+ AA^+ = A^+ \quad \text{(} A^+ \text{ is a weak inverse for the multiplicative semigroup);}
\]

\[
(AA^+)^* = AA^+ \quad \text{(} AA^+ \text{ is Hermitian); and}
\]

\[
(A^+ A)^* = A^+ A \quad \text{(} A^+ A \text{ is also Hermitian).}
\]

A common use of the pseudo-inverse is to compute a 'best fit' (least squares) solution to a system of linear equations that lacks a unique solution. Another use is to find the minimum (Euclidean) norm solution to a system of linear equations with multiple solutions.

The singular value decomposition can be used for computing the pseudo-inverse of a rectangular or singular matrix. Indeed, the pseudo-inverse of the matrix \( M \) with singular value decomposition \( M = U \Sigma V^* \) is \( M^+ = V \Sigma^+ U^* \), where \( \Sigma^+ \) is the pseudo-inverse of \( \Sigma \), which is formed by replacing every non-zero diagonal entry by its reciprocal and transposing the resulting matrix. The pseudo-inverse is one way to solve linear least squares problems.

**Theorem (Gauss and Legendre).** Every linear system \( Ax = b \), where \( A \) is an \( m \times n \) matrix, has a solution \( \hat{x} \) minimizing

\[
\| A\hat{x} - b \|_2.
\]

These solutions are given by the equation \( A^tAx = A^tb \), called the normal equations. Furthermore, when the columns of \( A \) are linearly independent, it turns out that \( A^tA \) is invertible, and so \( \hat{x} \) is unique and given by

\[
\hat{x} = (A^tA)^{-1}Ab.
\]

**Solving a System of Linear Equations Using Generalized Inverse.**

Consider the following systems of linear equations:

\[
S_1 : \begin{cases}
x_1 + x_2 - x_3 = 4 \\
x_1 - x_2 + x_3 = 2
\end{cases}
\quad \text{and} \quad
S_2 : \begin{cases}
y_1 + y_2 = 4 \\
y_1 - y_2 = 2 \\
-y_1 + y_2 = -2
\end{cases}
\]

Notice that \( S_1 \) has more unknowns than equations, but \( S_2 \) has more equations than unknowns.

The matrix equations for \( S_1 \) and \( S_2 \) are as follows:

\[
S_1 : A_1X = b_1, \quad \text{where} \quad A_1 = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix}, \quad X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad \text{and} \quad b_1 = \begin{bmatrix} 4 \\ 2 \end{bmatrix};
\]

\[
S_2 : A_2Y = b_2, \quad \text{where} \quad A_2 = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}, \quad Y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \quad \text{and} \quad b_2 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}.
\]

To solve these two systems, first we need to find the SVD of both \( A_1 \) and \( A_2 \), then from SVD, obtain \( A_1^+ \) and \( A_2^+ \). The solutions to these two linear systems are obtained by multiplying \( A_1^+ \) by \( b_1 \) and \( A_2^+ \) by \( b_2 \).
**Step 1.** Construct:

\[ B_1 = A_1 A_1^t = \begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}, \quad C_1 = A_1^t A_1 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & -2 \\ 0 & -2 & 2 \end{bmatrix}; \]

\[ B_2 = A_2 A_2^t = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & -2 \\ 0 & -2 & 2 \end{bmatrix}, \quad C_2 = A_2^t A_2 = \begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}. \]

**Step 2.** We have \( \text{rank}(A_1) = \text{rank}(A_2) = 2 \). Thus the characteristic polynomials of the above matrices are as follows:

\[ K_{B_1}(\lambda) = K_{C_2}(\lambda) = (\lambda - 4)(\lambda - 2) \quad \text{and} \quad K_{C_1}(\lambda) = K_{B_2}(\lambda) = (\lambda - 4)(\lambda - 2)\lambda. \]

**Step 3.** These eigenvalues produce the matrices

\[ \Sigma_1 = \begin{bmatrix} \sqrt{2} & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{bmatrix} \quad \text{and} \quad \Sigma_2 = \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix}. \]

**Step 4.** Next we need to find all linearly independent eigenvectors, corresponding to \( \lambda_1 = 4, \lambda_2 = 2, \) and \( \lambda_3 = 0 \), in order to construct \( U_1, V_1 \) and \( U_2, V_2 \).

\[ B(A_1 A_1^t) = B(A_2 A_2^t) = \left\{ \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\} \quad \text{and} \quad B(A_1 A_1^t) = B(A_2 A_2^t) = \left\{ \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\}. \]

**Step 5.** By normalizing the above eigenvectors, we obtain:

\[ B_\perp(A_1 A_1^t) = B_\perp(A_2 A_2^t) = \left\{ \begin{pmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}, \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \right\} \quad \text{and} \quad B_\perp(A_1 A_1^t) = B_\perp(A_2 A_2^t) = \left\{ \begin{pmatrix} 0 \\ -\frac{1}{\sqrt{2}} \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} \right\}. \]

**Step 6.** Using these two basis, we construct:

\[ U_1 = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = V_2 \quad \text{and} \quad V_1 = \begin{bmatrix} 0 & 1 & 0 \\ -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} = U_2. \]
Step 7. The factorization of $A_1$ and $A_2$ are as follows.

$$A_1 = U_1 \Sigma_1 V_1^t = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix};$$

$$A_2 = U_2 \Sigma_2 V_2^t = \begin{bmatrix} 0 & 1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}.$$

Notice that by eliminating the third column of $\Sigma_1$ and the third row of $V_1^t$ (resp. the third row of $\Sigma_2$ and the third column of $U_2$), we obtain

$$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix} = A_1;$$

$$\begin{bmatrix} 0 & 1 \\ \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} = A_2.$$

Step 8. Now we can get the pseudo-inverses:

$$A_1^t = V_1 \Sigma_1^t U_1^t = \begin{bmatrix} 0 & 1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 \\ -1/4 & 1/4 \end{bmatrix};$$

$$A_2^t = V_2 \Sigma_2^t U_2^t = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 1/2 & -1/4 & 1/4 \end{bmatrix}.$$

Step 9. By multiplying $A_1^t$ by $b_1 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$, we obtain $\begin{bmatrix} 3 \\ 1/2 \end{bmatrix}$, as the solution of the linear system $S_1$.

This solution has minimum Euclidean norm, among all possible solutions.

Also, by multiplying $A_2^t$ by $b_2 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$, we obtain $\begin{bmatrix} 3 \\ -2 \end{bmatrix}$ which is the unique solution of the linear system $S_2$.

Least Squares Estimation. Given an $m \times n$ matrix $A$ and an $m$-vector $b$, if the constant vector $b$ is not in the range of the matrix $A$, then there is no vector $x$ such that $Ax = b$. So

$$x = V \left[ \text{diag} (\sigma_1^{-1}, \sigma_2^{-1}, \ldots, \sigma_r^{-1}) \right] U^t b$$

cannot be used to obtain an exact solution. However, the vector returned will do the closest possible job in the least squares sense. It will find the vector $x$ which minimizes $R = \|Ax - b\|$. $R$ is called the residual of the solution.
For example, there exists a unique solution in the case of
\[
A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix},
\]
but not if \( b = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^t \). In such cases, when \( b \) is not in the range of \( A \), then we seek to minimize the residual of the solution.

**Finding the Solution of the Least Squares Problem** \( Ax \approx b \) with Minimal Norm.

Consider the following linear systems:
\[
A_1 x = b_1 : \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix} \quad \text{and} \quad A_2 y = b_2 : \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}.
\]

**Step 1.** Compute the SVD: \( [USV]^t = svd(A) \).

\[
A_1 = U_1 \Sigma_1 V_1^t = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -\frac{1}{\sqrt{2}} & 0 \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix};
\]

\[
A_2 = U_2 \Sigma_2 V_2^t = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.
\]

**Step 2.** Choose \( r \) such that \( \sigma_r \) is the smallest non-zero singular-value.

**Step 3.** Find \( U_r, \Sigma_r \) and \( V_r \).

\[
\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -\frac{1}{\sqrt{2}} & 0 \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix};
\]

\[
\begin{bmatrix} 0 & 1 \\ -\frac{1}{\sqrt{2}} & 0 \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.
\]

**Step 4.** The solution with minimal norm is

\[
x^* = V_r \sigma_r^{-1} U_r^t b.
\]

\[
x^* = \begin{bmatrix} 0 & 1 \\ \frac{1}{\sqrt{2}} & 0 \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix}^{-1} \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 4 \\ 2 \end{bmatrix} = \begin{bmatrix} 3 \\ 1/2 \end{bmatrix};
\]

\[
y^* = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix}^{-1} \begin{bmatrix} 0 & -\frac{1}{\sqrt{2}} \frac{1}{\sqrt{2}} \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 4 \\ -2 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}.
\]
Notice that \( A_1 x^* = b_1 \) and \( A_2 x^* = b_2 \).

Now, consider the system:

\[
A_3 x = b_3 : \begin{bmatrix}
0 & 1 \\
1 & 0 \\
0 & 0
\end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}
\]

Using SVD, we obtain:

\[
U = \begin{bmatrix}
0 & -1 \\
-1 & 0 \\
0 & 0
\end{bmatrix}, \quad S = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad V = \begin{bmatrix}
-1 & 0 \\
0 & -1
\end{bmatrix}.
\]

The rank of \( A_3 \) is two, so we find

\[
U_2 = \begin{bmatrix}
0 & -1 \\
-1 & 0 \\
0 & 0
\end{bmatrix}, \quad S_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad V_2 = \begin{bmatrix}
-1 & 0 \\
0 & -1
\end{bmatrix}.
\]

The optimal solution is \( z^* = V_2 S_2^{-1} U_2^t b_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \). But \( A_3 z^* = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \neq \begin{bmatrix} 1 \\ 1 \end{bmatrix} \). The reason for not being able to find an exact solution is that \( b_3 \) is not in the range of \( A_3 \).

**Data Compression.** Any image from a digital camera, scanner or in a computer is a digital image. Suppose we have an \( 9 \) megapixel gray-scale image, which is \( a \times b = 9 \) megapixels (an \( a \times b \) matrix). For each pixel, we have some level of black and white, given by some integer between 0 and 255. Each of these integers (and hence each pixel) requires approximately 1 byte to store, resulting in an approximately 8.6 Mb image.

A color image usually has three components, a red, a green, and a blue (RGB). The composite of the three RGB values creates the final color for the single pixel. So storing color images involves three matrices of the same size, so it takes three times the space (25.8 Mb).

Now suppose we have a digital image taken with a 9 megapixel camera and each color pixel is determined by a 24-bit number. when we print the picture suppose we only use 8-bit colors. We are still using 9 million pixels but the information used to describe the image has been reduced by one-third. This is an example of image compression.

We will look at compressing this image through computing the singular value decomposition (SVD). The truncated SVD of the real \( m \times n \) matrix \( A \) with rank \( r \) is

\[
A = U_r \Sigma_r V_r^t = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \ldots & 0 \\ 0 & \sigma_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \sigma_r \end{bmatrix} \begin{bmatrix} v_1^t \\ v_2^t \\ \vdots \\ v_n^t \end{bmatrix} = u_1 \sigma_1 v_1^t + u_2 \sigma_2 v_2^t + \cdots + u_m \sigma_r v_n^t,
\]

where \( U_r \) is an \( m \times r \) matrix with orthonormal columns, \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \), and \( V_r \) is an \( n \times r \) matrix with orthonormal rows.

Since \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \), the terms \( u_i \sigma_i v_i^t \) with small \( i \) contribute most to the sum, and hence
contain the most information about the image. Keeping only some of these terms may result in a lower image quality, but lower storage size. Thus we can reduce the size of those three matrices. This process is sometimes called Principal Component Analysis (PCA).

Suppose we are using the SVD on an \( n \times n \) matrix \( A \). Then

- We need to store \( 2n^2 \) elements for \( U \) and \( V \) and at most \( n \) elements for \( \Sigma \).
- Keeping 1 term in the SVD, \( u_1 \sigma_1 v_1^t \) requires only \( 2n + 1 \) elements.
- If we keep only half of the \( \sigma_i \)'s, then storing the truncated SVD and the original matrix are approximately the same.
- Color images are in RGB, where each color is specified by 0 to 255. This gives us three matrices. The truncated SVD can be computed on all three separately or together.
- We are also interested in finding the percentage of relative change of a rank-reduced matrix to determine by how much we may reduce without significant loss of information. This is done by the formula:

\[
\frac{\| A - A_k \|_F}{\| A \|_F}.
\]

Click here to see an example of Data Compression

**Example.** Consider the \( 8 \times 6 \) matrix \( A \) of rank 3 and the vector \( b \):

\[
A = \begin{pmatrix}
64 & 2 & 3 & 61 & 60 & 6 \\
9 & 55 & 54 & 12 & 13 & 51 \\
17 & 47 & 46 & 20 & 21 & 43 \\
40 & 26 & 27 & 37 & 36 & 30 \\
32 & 34 & 35 & 29 & 28 & 38 \\
41 & 23 & 22 & 44 & 45 & 19 \\
49 & 15 & 14 & 52 & 53 & 11 \\
8 & 58 & 59 & 5 & 4 & 62 \\
\end{pmatrix}
\quad \text{and} \quad
b = \begin{pmatrix}
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
\end{pmatrix}.
\]

The SVD of \( A \) produces:

\[
U = \begin{pmatrix}
-0.3554 & 0.5585 & 0.3215 & 0.6644 & 0.1012 & -0.0192 & 0.0769 & 0.0193 \\
-0.3517 & -0.4047 & -0.3336 & 0.1951 & 0.7232 & -0.1989 & 0.0217 & -0.0014 \\
-0.3517 & -0.2507 & -0.3421 & 0.2965 & -0.4080 & 0.4722 & -0.3571 & 0.3027 \\
-0.3554 & 0.0963 & 0.3470 & -0.3927 & 0.0842 & -0.3018 & -0.6515 & 0.2591 \\
-0.3554 & -0.0578 & 0.3554 & -0.3453 & 0.2345 & 0.6869 & 0.1987 & -0.2418 \\
-0.3518 & 0.2115 & -0.3675 & -0.1177 & -0.2355 & -0.1752 & -0.1521 & -0.7571 \\
-0.3518 & 0.3656 & -0.3760 & -0.3739 & -0.0797 & -0.0980 & 0.4875 & 0.4558 \\
-0.3553 & -0.5200 & 0.3809 & 0.0737 & -0.4199 & -0.3659 & 0.3760 & -0.0367 \\
\end{pmatrix}
\]
\[ \Sigma = \begin{pmatrix} 225.1696 & 0 & 0 & 0 & 0 & 0 \\ 0 & 127.1865 & 0 & 0 & 0 & 0 \\ 0 & 0 & 11.7579 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \]

\[ V = \begin{pmatrix} -0.4085 & 0.4110 & 0.5583 & 0.3300 & 0.2377 & -0.4326 \\ -0.4080 & -0.4104 & -0.3983 & 0.2141 & -0.3786 & -0.5632 \\ -0.4081 & -0.4092 & -0.1582 & -0.1070 & 0.7810 & 0.1400 \\ -0.4083 & 0.4073 & -0.1623 & -0.7846 & -0.1220 & -0.1025 \\ -0.4082 & 0.4061 & -0.4025 & 0.4546 & -0.1157 & 0.5351 \\ -0.4083 & -0.4055 & 0.5624 & -0.1071 & -0.4024 & 0.4232 \end{pmatrix} \]

- The fact that the rank of \( A \) is three, we may truncate the above matrices and obtain:

\[ U_3 = U (:, 1 : 3) = \begin{pmatrix} -0.3554 & 0.5585 & 0.3215 \\ -0.3517 & -0.4047 & -0.3336 \\ -0.3517 & -0.2507 & -0.3421 \\ -0.3554 & 0.0963 & 0.3470 \\ -0.3554 & -0.0578 & 0.3554 \\ -0.3518 & 0.2115 & -0.3675 \\ -0.3518 & 0.3656 & -0.3760 \\ -0.3553 & -0.5200 & 0.3809 \end{pmatrix}, \quad \Sigma_3 = \begin{pmatrix} 225.1696 & 0 & 0 \\ 0 & 127.1865 & 0 \\ 0 & 0 & 11.7579 \end{pmatrix}, \]

\[ V_3 = V (:, 1 : 3) = \begin{pmatrix} -0.4085 & 0.4110 & 0.5583 \\ -0.4080 & -0.4104 & -0.3983 \\ -0.4081 & -0.4092 & -0.1582 \\ -0.4083 & 0.4073 & -0.1623 \\ -0.4082 & 0.4061 & -0.4025 \\ -0.4083 & -0.4055 & 0.5624 \end{pmatrix} \]

with

\[ U_3 \Sigma_3 V_3^t = \begin{pmatrix} 64 & 2 & 3 & 61 & 60 & 6 \\ 9 & 55 & 54 & 12 & 13 & 51 \\ 17 & 47 & 46 & 20 & 21 & 43 \\ 40 & 26 & 27 & 37 & 36 & 30 \\ 32 & 34 & 35 & 29 & 28 & 38 \\ 41 & 23 & 22 & 44 & 45 & 19 \\ 49 & 15 & 14 & 52 & 53 & 11 \\ 8 & 58 & 59 & 5 & 4 & 62 \end{pmatrix} = A \]

- By Choosing only the first two singular values of \( A \) and keeping the first two columns of \( U \) and
V; i.e., $U_2 = U(:, 1 : 2)$ and $V_2 = V(:, 1 : 2)$, we obtain:

$$U_2 = \begin{pmatrix} -0.3554 & 0.5585 \\ -0.3517 & -0.4047 \\ -0.3517 & -0.2507 \\ -0.3554 & 0.0963 \\ -0.3554 & -0.0578 \\ -0.3518 & 0.2115 \\ -0.3518 & 0.3656 \\ -0.3553 & -0.5200 \end{pmatrix}, \quad \Sigma_2 = \begin{pmatrix} 225.1696 & 0 \\ 0 & 127.1865 \end{pmatrix}, \quad V_2 = \begin{pmatrix} -0.4085 & 0.4110 \\ -0.4080 & -0.4104 \\ -0.4081 & -0.4092 \\ -0.4083 & 0.4073 \\ -0.4082 & 0.4061 \\ -0.4083 & -0.4055 \end{pmatrix},$$

with


According to Eckart and Young theorem, $A_2$ is the best rank-2 matrix approximation to $A$. The Frobenius norm: $\| A - A_2 \|_F = 11.7579 = \sigma_{33}$. The 2-norm $\| A - A_2 \|_2$ is also 11.7579.

$$\text{norm} (A - A_2, 'fro') = 11.7579 \quad \text{and} \quad \text{norm} (A - A_2, 2) = 11.7579.$$  

The relative error of reducing $A$ to the rank 2 matrix $A_2$ is

$$\frac{\| A - A_2 \|_F}{\| A \|_F} = 0.0454.$$  

This means that the loss of information from $A$ to $A_2$ is only 4.5%.

- By Choosing only the first singular value of $A$ and keeping the first columns of $U$ and $V$; i.e., $U_1 = U(:, 1)$ and $V_1 = V(:, 1)$, we obtain:

$$U_1 = \begin{pmatrix} -0.3554 \\ -0.3517 \\ -0.3517 \\ -0.3554 \\ -0.3554 \\ -0.3518 \\ -0.3518 \\ -0.3553 \end{pmatrix}, \quad \Sigma_1 = \begin{pmatrix} 225.1696 \end{pmatrix}, \quad V_1 = \begin{pmatrix} -0.4085 \\ -0.4080 \\ -0.4081 \\ -0.4083 \\ -0.4082 \\ -0.4083 \end{pmatrix},$$
with
\[ U_1 \Sigma_1 V_1^t = \begin{pmatrix}
32.6935 & 32.6576 & 32.6628 & 32.6780 & 32.6728 & 32.6783 \\
32.3475 & 32.3119 & 32.3171 & 32.3321 & 32.3270 & 32.3324 \\
32.3493 & 32.3138 & 32.3189 & 32.3339 & 32.3288 & 32.3343 \\
32.6879 & 32.6521 & 32.6572 & 32.6724 & 32.6672 & 32.6728 \\
32.6861 & 32.6502 & 32.6554 & 32.6606 & 32.6654 & 32.6709 \\
32.3549 & 32.3194 & 32.3245 & 32.3395 & 32.3344 & 32.3399 \\
32.3568 & 32.3212 & 32.3264 & 32.3414 & 32.3363 & 32.3417 \\
32.6805 & 32.6446 & 32.6498 & 32.6550 & 32.6598 & 32.6653 \\
\end{pmatrix} = A_1. \]

According to Eckart and Young theorem, \( A_1 \) is the best rank-1 matrix approximation to \( A \).
The Frobenius norm: \( \| A - A_1 \|_F = 127.1865 = \sigma_2 \). The 2-norm \( \| A - A_1 \|_2 \) is also 127.1865.

\[ \text{norm} ( A - A_1, 'fro') = 127.1865 \quad \text{and} \quad \text{norm} ( A - A_1, 2 ) = 127.1865. \]

The relative error of reducing \( A \) to the rank one matrix \( A_1 \) is
\[ \frac{\| A - A_1 \|_F}{\| A \|_F} = 0.4934. \]

This means that the loss of information from \( A \) to \( A_1 \) is about 50\%. Clearly this kind of loss is not acceptable.

To find the solution of the least squares problem \( Ax \approx b \) with minimal norm, we use \( U_3, \Sigma_3 \) and \( V_3 \). We have
\[ x^* = V_3 \Sigma_3^{-1} U_3^t b = 10^{-3} \begin{pmatrix}
4.4378 \\
5.6213 \\
5.3254 \\
5.3254 \\
5.6213 \\
4.4378 \\
\end{pmatrix}. \]

If we use \( U_2, \Sigma_2 \) and \( V_2 \), we obtain
\[ y^* = V_2 \Sigma_2^{-1} U_2^t b = 10^{-3} \begin{pmatrix}
5.1273 \\
5.1292 \\
5.1301 \\
5.1249 \\
5.1241 \\
5.1325 \\
\end{pmatrix} \quad \text{with} \quad \| Ay - b \|_2 = 0.01452. \]

**Remark.** Notice that the matrix \( A \) has \( 8 \times 6 = 48 \) entries. The total number of entries of \( U_3, \Sigma_3 \), and \( V_3 \) is \( 8 \times 3 + 3 \times 3 + 6 \times 3 = 51 \). If we store these three matrices instead of the original matrix \( A \), we only need to store three more entries, but then these matrices may be used to produce the original matrix \( A \) and to define \( A_2 \) and \( A_1 \).
Exercise. Consider the $5 \times 5$ matrix $A$ of rank four and the vector $b$:

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & -12 & 13 & 14 & 15 \\ 16 & 17 & -18 & 19 & -20 \\ 21 & 22 & 23 & 24 & 25 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} 1 \\ 2 \\ 5 \\ -3 \\ 2 \end{pmatrix}.$$

Use MATLAB to find:

1. The best rank-3 matrix approximation $A_3$ to $A$.
2. The best rank-2 matrix approximation $A_2$ to $A$.
3. The best rank-1 matrix approximation $A_1$ to $A$.
4. Find the loss of information for each case.
5. Find the solution of the least squares problem $Ax \approx b$ with minimal norm.
6. Use $A_3$ to find the solution of the least squares problem $Ax \approx b$. Also find the residual.

Web Searching. Search engines like Google use enormous matrices of cross-references in which pages link to which other pages, and what words are on each page. When you do a Google search, the higher ranks usually go to pages with your key words that have lots of links to them. But there are billions of pages out there, and storing a several hundred thousands by several billion matrix is trouble, not to mention searching through it.

Definition 4. Latent semantic indexing (LSI) is an indexing and retrieval method that uses the singular value decomposition to identify patterns in the relationships between the terms and concepts contained in an unstructured collection of text. LSI is based on the principle that words that are used in the same contexts tend to have similar meanings. A key feature of LSI is its ability to extract the conceptual content of a body of text by establishing associations between those terms that occur in similar contexts.

Definition 5. A term-document matrix is a matrix that describes the frequency of terms that occur in a collection of documents. The $(i,j)$-element of the matrix $A = (a_{ij})$ is a positive integer $p$, if the keyword in position $i$ appears $p$ times in document $j$; and is 0 otherwise.

The 3 billion document vectors associated with the Web (each of which is a 300,000 vector) to one another to form a 300,000-by-3,000,000,000 term-by-document matrix.

<table>
<thead>
<tr>
<th>$d_1$</th>
<th>$d_2$</th>
<th>$d_3$</th>
<th>$d_4$</th>
<th>$d_5$</th>
<th>$d_6$</th>
<th>...</th>
<th>...</th>
<th>...</th>
<th>$d_{3,000,000,000}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>term$_1$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>term$_2$</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>term$_3$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>:</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>:</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>term$_{300,000}$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

This huge term-by-document matrix may contain redundant information. This means that there are dependence among columns of the matrix. The LSI method manipulates the matrix to eradicate...
dependencies and thus consider only the independent, smaller part of this large term-by-document matrix.

In searching, you really care about the main directions that the Web is taking. So the first few singular values create a very good approximation for the enormous matrix.

The SVD produces \( A = U \Sigma V^T \). For each \( k \) less than the rank of \( A \), the truncated matrix will be

\[
A_k = U_k \Sigma_k V_k^T.
\]

The goal is using the truncated \( A_k = U_k \Sigma_k V_k^T \), to approximate \( A \) by choosing \( k \) large enough to give a good approximation to \( A \), yet small enough so that \( k << r \) requires much less storage.

Typically one does not only want to know about matches, but one would like to know the most relevant documents, i.e., one would like to rank the documents according to relevance.

The angle \( \theta_j \) between the \( j \)-th column of a term-by-document matrix \( A = [A_1, A_2, \ldots, A_j, \ldots, A_n] \) and the query vector \( q \) can be computed by

\[
\cos(\theta_j) = \frac{q^T A_j}{\|q\| \|A_j\|}. \quad (j = 1, 2, \ldots, n).
\]

Notice that when \( q \) is identical with the \( j \)-th column of \( A \), the angle between \( q \) and \( A_j \) vanishes and the cosine of this angle is one. Thus, a large value of \( \cos(\theta_j) \) suggests that document \( j \) may be relevant.

**Example.** If one has the following four (short) documents:

Document 1= \{Math, Math, Calculus, SVD, Algebra\}

Document 2= \{Math, Linear , SVD, Abstract, Algebra \}

Document 3= \{Linear, Abstract, Algebra, Calculus, Math, SVD\}

Document 4= \{Math, Algebra, Calculus, SVD, SVD\}

The term-document matrix is:

<table>
<thead>
<tr>
<th>Term</th>
<th>Document 1</th>
<th>Document 2</th>
<th>Document 3</th>
<th>Document 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Algebra</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Calculus</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Linear</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Math</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>SVD</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

which shows which documents contain which terms and how many times they appear.
From the above table we define the matrices \( A \), \( U \), \( S \), and \( V \):

\[
A = \begin{pmatrix}
0 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 \\
0 & 1 & 1 & 0 \\
2 & 1 & 1 & 1 \\
1 & 1 & 1 & 2
\end{pmatrix}, \quad U = \begin{pmatrix}
-0.2027 & -0.6485 & 0 & -0.1288 & -0.7142 & -0.1083 \\
-0.4359 & -0.0922 & 0 & 0.1375 & 0.3084 & -0.8292 \\
-0.3427 & 0.2849 & 0 & -0.8952 & 0 & 0 \\
-0.2027 & -0.6485 & 0 & -0.1288 & 0.6114 & 0.3847 \\
-0.5524 & 0.1860 & 0.7071 & 0.2707 & -0.1028 & 0.2764 \\
-0.5524 & 0.1860 & -0.7071 & 0.2707 & -0.1028 & 0.2764
\end{pmatrix},
\]

\[
S = \begin{pmatrix}
4.5710 & 0 & 0 & 0 \\
0 & 1.6426 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0.6387 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{pmatrix}, \quad V = \begin{pmatrix}
-0.5329 & 0.4569 & 0.7071 & 0.0850 \\
-0.4258 & -0.6194 & 0 & 0.6596 \\
-0.5007 & -0.4459 & 0 & -0.7419 \\
-0.5329 & 0.4569 & -0.7071 & 0.0850
\end{pmatrix}.
\]

According to the diagonal entries of the matrix \( S \), the rank of the matrix \( A \) is four. Now suppose we are only interested in the two largest singular values of the matrix \( A \); so we are going to truncate the SVD and find \( A_2 \).

\[
U_2 = \begin{pmatrix}
4.5710 & 0 & 0 & 0 \\
0 & 1.6426 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0.6387 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{pmatrix}, \quad S_2 = \begin{pmatrix}
4.5710 & 0 \\
0 & 1.6426
\end{pmatrix}, \quad V_2 = \begin{pmatrix}
-0.5329 & 0.4569 \\
-0.4258 & -0.6194 \\
-0.5007 & -0.4459 \\
-0.5329 & 0.4569
\end{pmatrix}.
\]

\[
A_2 = U_2 S_2 V_2^t = \begin{pmatrix}
0.0070 & 1.0543 & 0.9390 & 0.0070 \\
0.9925 & 0.9421 & 1.0652 & 0.9925 \\
1.0486 & 0.3772 & 0.5758 & 1.0486 \\
0.0070 & 1.0543 & 0.9390 & 0.0070 \\
1.4853 & 0.8860 & 1.1283 & 1.4853 \\
1.4853 & 0.8860 & 1.1283 & 1.4853
\end{pmatrix}
\]

Suppose we are searching for “Linear Algebra,” using the term-by-document matrix \( A \). Our query vector, then will be

\[
q = \begin{pmatrix}
0 \\
1 \\
0 \\
1 \\
0 \\
0
\end{pmatrix}^t.
\]

By taking cosine of \( q \) and each columns of \( A \) (resp \( A_2 \)), we obtain:

<table>
<thead>
<tr>
<th></th>
<th>Document 1</th>
<th>Document 2</th>
<th>Document 3</th>
<th>Document 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cosine of the angle of ( q ) and columns of ( A )</td>
<td>0.2673</td>
<td>0.6325</td>
<td>0.5774</td>
<td>0.2673</td>
</tr>
<tr>
<td>Cosine of the angle of ( q ) and columns of ( A_2 )</td>
<td>0.2773</td>
<td>0.6428</td>
<td>0.5897</td>
<td>0.2773</td>
</tr>
</tbody>
</table>

Notice that both \( A \) and \( A_k \) return documents two and three, the most relevant documents for our query “Linear Algebra.”
Exercise. Consider the term-by-document matrix and the query vector:

\[
C = \begin{bmatrix}
1 & 0 & 2 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 \\
3 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 4 \\
1 & 0 & 5 & 1 & 0 \\
0 & 0 & 5 & 1 & 0 \\
1 & 1 & 1 & 0 & 1
\end{bmatrix}, \quad q = \begin{bmatrix}
0 \\
1 \\
1 \\
0 \\
0 \\
1 \\
0
\end{bmatrix}.
\]

Order the documents in order of relevancy.