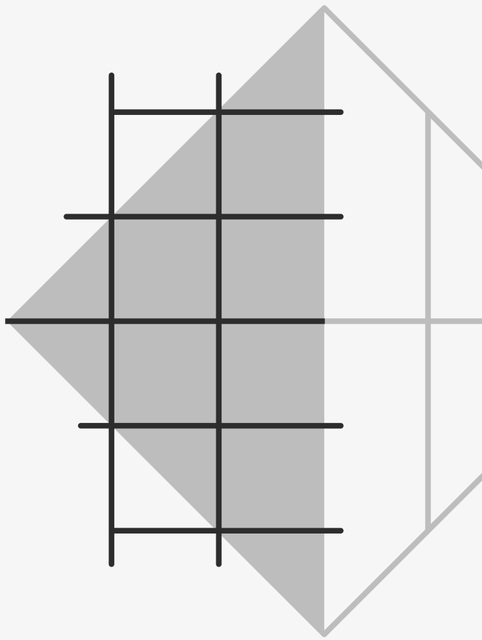
Matrix Decomposition

A study of various methods and algorithms

Team 33
Khooshi Asmi (2022114006)
Saravana Mitra Somayaji Tangirala (2022102016)
Vyakhya Gupta (2022101104)

Overview

This project focuses on the study and analysis of matrix decomposition techniques, namely Singular Value Decomposition (SVD), Eigenvalue Decomposition, LU Decomposition, and QR Decomposition. These decomposition methods have significant applications in various fields of science and engineering, including linear algebra, signal processing, data analysis, and machine learning.The project begins by introducing the concepts and principles behind each decomposition method, and exploring their mathematical formulations and properties.



Our Team

Saravana Mitra Somayaji Tangirala
2022102016

Implementation of SVD algorithm and eigenvalue decomposition algorithm, performance evaluation, and comparative analysis and application.

Vyakhya Gupta
2022101104

Implementation of LU decomposition algorithm and QR decomposition algorithm and their performance evaluation, and comparative analysis and application.

Khooshi Asmi
2022114006

Exploration of applications of matrix decomposition in various fields, signal processing, and machine learning. Compilation of individual work in LaTeX.

SVD Decomposition

Introduction

Singular Value Decomposition (SVD) is a powerful matrix factorization technique in linear algebra that has found extensive applications across various fields. It provides a unique and efficient representation of matrices, allowing for dimensionality reduction, data compression, and extracting important features.

Applications

Dimensionality Reduction

Image Compression

Principal component analysis

Theory and mathematical formulation

Singular value decomposition takes a rectangular matrix of gene expression data (defined as A, where A is an n × p matrix) in which the n rows represent the genes, and the p columns represent the experimental conditions. The SVD theorem states:

$$A_{n\times p} = U_{n\times n} \times \Sigma_{n\times p} \times V^{T}_{p\times p}$$

where U and V are orthogonal

Applications

NETFLIX AND SVD:

Given the very large data matrix it was only expected that competitors attempted to do dimensionality reduction and as it turns out this was the basis for the winning algorithm. Dimensionality reduction can be done via matrix factorization that has the following advantage: when explicit feedback is not available, we can infer user preferences using implicit feedback, which indirectly reflects opinion by observing user behavior including purchase history, browsing history, search patterns, or even mouse movements.

Eigenvalue Decomposition

Introduction

In linear algebra, eigendecomposition is the factorization of a matrix into a canonical form, whereby the matrix is represented in terms of its eigenvalues and eigenvectors. Only diagonalizable matrices can be factorized in this way. When the matrix being factorized is a normal or real symmetric matrix,
the decomposition is called ”spectral decomposition,” derived from the spectral theorem

Applications

Data Preparation

Covariance Matrix

Dimensionality Reduction

Selection of Principal Components

Steps:

To perform eigenvalue decomposition, follow these steps:

1. Ensure that the matrix A is a square matrix.
2. Calculate the eigenvalues (λ) of the matrix A. These are the values that satisfy the equation $Av = \lambda v$, where v is the eigenvector corresponding to the eigenvalue λ.
3. Solve the equation $(A - \lambda I)v = 0$, where I is the identity matrix and v is the eigenvector. This equation helps find the eigenvectors corresponding to each eigenvalue.
4. Arrange the eigenvectors v1, v2, ..., vn in matrix V, where each eigenvector is a column vector.
5. Arrange the eigenvalues λ1, λ2, ..., λn in a diagonal matrix Λ.
6. Calculate the inverse of matrix V, V^{-1} .
7. Finally, express the original matrix A as $A = V\Lambda V^{-1}$.

Classifications

1. Symmetric Eigenvalue Decomposition: This type of decomposition is applicable to symmetric matrices. For a real symmetric matrix, the decomposition is given by A = QΛQ^T.
2. Generalized Eigenvalue Decomposition: This decomposition is used when dealing with two related matrices. Given two matrices A and Bthe decomposition is represented as Av = λBv. The eigenvectors and eigenvalues are then used to form the matrices V and Λ, respectively.
3. Jordan Canonical Form: The Jordan Canonical Form decomposes a matrix into a block diagonal matrix, where each block represents Jordan blocks associated with eigenvalues and generalized eigenvectors.

LU Decomposition

Introduction

LU decomposition, also known as LU factorization, is a matrix factorization technique that decomposes a square matrix A into the product of two matrices: a lower triangular matrix L and an upper triangular matrix U . The LU decomposition can be represented as A = LU .

Applications

Solving Systems of Linear Equations

Computing Matrix Inverses

Sparse Matrix Computations

Theory and mathematical formulation

Gaussian Elimination

To obtain the LU decomposition, we start by performing Gaussian elimination with partial pivoting on the matrix A. The goal is to transform A into an upper triangular matrix U, while simultaneously applying the same row operations to another matrix L, which keeps track of the multipliers used in the elimination steps. In the operation Ri − kRj , we will refer to the scalar k as the multiplier. The multipliers are precisely the entries of L that are below its diagonal.

Lower and Upper matrices

The lower triangular matrix L is constructed based on the multipliers used during the elimination process. It has ones on its main diagonal and the multipliers below the diagonal.

The resulting matrix U is the upper triangular matrix obtained after performing Gaussian elimination.

PTLU decomposition

Let A be a square matrix. A factorization of A as $A = P^T LU$, where P is a permutation matrix, L is unit lower triangular, and U is upper triangular, is called a $P^T LU$ factorization of A.
Example: To find $P^T LU$ factorization of

$$A = \begin{bmatrix} 0 & 0 & 6 \\ 1 & 2 & 3 \\ 2 & 1 & 4 \end{bmatrix}$$

First we reduce A to row echelon form. Clearly, we need at least one row interchange.

$$A = \begin{bmatrix} 0 & 0 & 6 \\ 1 & 2 & 3 \\ 2 & 1 & 4 \end{bmatrix} -> \begin{bmatrix} 1 & 2 & 3 \\ 0 & 0 & 6 \\ 2 & 1 & 4 \end{bmatrix} -> \begin{bmatrix} 1 & 2 & 3 \\ 0 & 0 & 6 \\ 0 & -3 & -2 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -2 \\ 0 & 0 & 6 \end{bmatrix}$$

We have used two row interchanges (R1 4 R2 and then R2 4 R3), so the required permutation matrix is

$$P = P_1 P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

So the LU factorisation of PA is

$$PA = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 4 \\ 0 & 0 & 6 \end{bmatrix} -> \begin{bmatrix} 1 & 2 & 3 \\ 0 & 3 & -2 \\ 0 & 0 & 6 \end{bmatrix} = U$$

So,

$$A = P^T LU = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 0 & 3 & -2 \\ 0 & 0 & 6 \end{bmatrix}$$

In general, P will be the product P = Pk ... P2P1 of all the row interchange
matrices P1, P2, ..., Pk (where P1 is performed first, and so on.)
Such a matrix
P is called a permutation matrix.

Solving AX=B

This involves two steps:

The original equation is to solve $Ax - b = 0$.
At the end of the Gaussian elimination, the resulting equations were:
$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1$ $a'_{22}x_2 + a'_{23}x_3 + \ldots + a'_{2n}x_n = b'_2$ $a''_{33}x_3 + a''_{34}x_4 + \ldots + a''_{3n}x_n = b''_3$ $\ldots$ $a_{nn}^{(n-1)}x_n = b_n^{(n-1)}$

which can be written as: $Ux - d = 0$ (1)

Premultiplying (1) by another matrix L, which results in: $L(Ux - d) = 0$
That is: $LUx - Ld = 0$ (2)

Comparing (1) and (2), it is clear that:
$LU = A$ $Ld = b$ (3)

To reduce computational load, L is taken as a lower triangular matrix with 1's along the diagonal.

1. Forward substitution: Solve $Ld = b$ to find d. The values of d_i are given by $d_1 = b_1$ $d_i = b_i - \sum_{j=1}^{i-1} l_{ij}d_j \quad i = 2, 3, ..., n$

2. Back substitution: Solve $Ux = d$ to find x. The values of x_i are given by $x_n = \frac{d_n}{u_{nn}}$ $x_i = \frac{d_i - \sum_{j=i+1}^{n} u_{ij}x_j}{u_{ii}} \quad i = n-1, n-2, ..., 1$

Cholesky Decomposition

The Cholesky factorization, also known as Cholesky decomposition, is a process of breaking down of a Hermitian, positive-definite matrix into the product of a lower triangular matrix and its conjugate transpose. If A is a real matrix (and thus symmetric positive-definite), the factorization can be stated as follows: A = LL^T

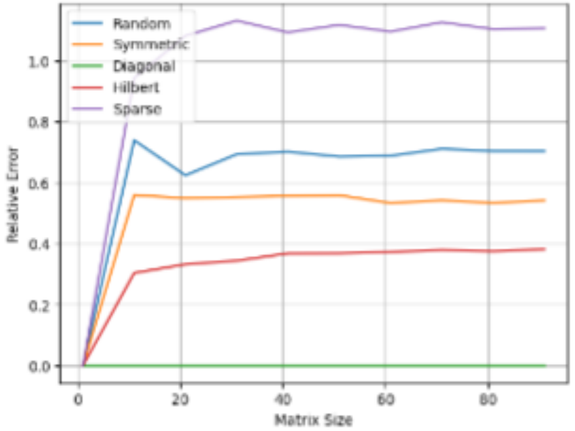
Initialize an $n \times n$ lower triangular matrix L with zeros
for $i = 1$ to n **do**
    for $j = 1$ to $i - 1$ **do**
        Compute the sum term: $\text{sum} = \sum_{k=1}^{j-1} L[i,k] \cdot L[j,k]$
    end for
    Compute the diagonal term: $L[i,i] = \sqrt{A[i,i] - \text{sum}}$
    for $j = i + 1$ to n **do**
        Compute the off-diagonal terms: $L[j,i] = \frac{A[j,i] - \sum_{k=1}^{j-1} L[j,k] \cdot L[i,k]}{L[i,i]}$
    end for
end for

Computation Time

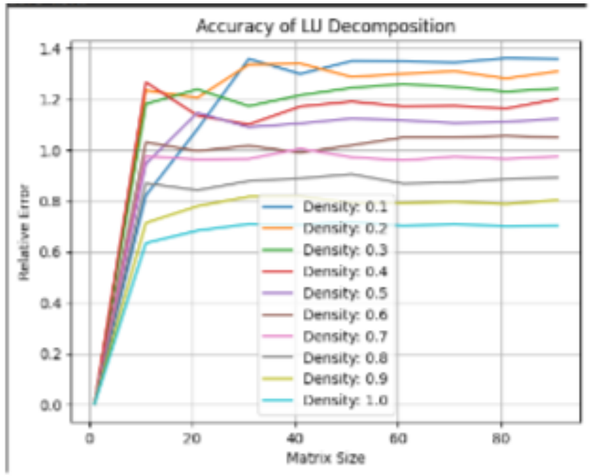
LU decomposition is less expensive time-wise compared to decompositions such as QR and SVD which involve sophisticated calculations and orthogonalisations.
The increase in size of matrix leads to a quick increase in time required.

Accuracy & Scalability

LU decomposition is less expensive time-wise compared to decompositions such as QR and SVD which involve sophisticated calculations and orthogonalisations. The increase in size of matrix leads to a quick increase in time required.



(a) Errors according to types of matrices



(b) Errors according to density of sparse matrices

Analysis

If A is n × n, then the total number of operations (multiplications and divisions) required to solve a linear system Ax = b using an LU factorization of A is
T = 1/3 n3, the same as is required for Gaussian elimination. This is since the forward elimination phase produces the LU factorization in 1/3 n3 steps, whereas both forward and backward substitution require 1/2 n2 steps. Therefore, for large values of n, the n3 term is dominant. From this point of view, then, Gaussian elimination and the LU factorization are equivalent.

However, the LU factorization has other advantages:

- For an invertible matrix A, an LU factorization of A can be used to find A^{-1}, if necessary. Moreover, this can be done in such a way that it simultaneously yields a factorization of A^{-1}.
- Numerical stability: LU decomposition is numerically stable for well-conditioned matrices, meaning it preserves the accuracy of the original matrix under small perturbations.
- Reusability: Once the LU decomposition is computed, it can be reused for solving multiple linear systems with the same coefficient matrix, leading to computational savings.

Limitations:
• Singularity: LU decomposition cannot be directly applied to singular matrices or matrices that become singular during the decomposition process. It requires additional techniques, such as pivoting, to handle these cases.
• It requires (like most) pivoting to ensure numerical stability.

Parallel LU Decomposition Method and It's Application in Circle Transportation

This paper presents a method that utilizes the Parallel LU Decomposition Algorithm for solving large-scale dense linear equations. The approach is based on the divide and conquer strategy, and it focuses on analyzing the speedup and efficiency of the algorithm. Furthermore, the parallel LU decomposition algorithm is applied to solve a circling transportation problem, showcasing its practical application.

Parallelizing the LU decomposition process allows for the distribution of computational tasks across multiple processors or computing units, enabling faster execution and improved scalability. The parallelization is typically achieved by partitioning the matrix and assigning subtasks to different processing elements. If there are P processors, matrix L were divided into P processors in row, then each processor have n/p row, matrix U were divided into P processors in column.

How to reasonably dispatch vehicles, minimize the transport process,to improve the run of empty transportation departments of economic significance. Under the background of the transportation problem with practical application,use of linear programming theory, developed a cycle of transport routes, combined cycle parallel LU decomposition algorithm for solving linear equations, the general theory of the transport plan.

Parallel LU Decomposition Method and It's Application in Circle Transportation: Yang, Aimin & Liu, Chunfeng & Chang, Jincai & Guo, Xiaoqiang. (2010). Research on Parallel LU Decomposition Method and It's Application in Circle Transportation. JSW. 5. 1250-1255. 10.4304/jsw.5.11.1250-1255.

In order to efficiently complete the transportation task, the vehicle dispatching department has chosen to use single measures for 5-ton load trucks. With a total requirement of 40 trucks, the available eight trucks must each make five round-trips. It is important to consider the specific transport routes to maximize efficiency and overall operational benefit. Dispatching officers need to consider the location where each truck should return for reloading after unloading goods, in order to maximize the overall transportation benefit obtained.

Table 1 goods transportation task

Name of goods	Delivery Point	Receiving Point	The quantity of goods (T)
H_1	A_1	B_1	55
H_2	A_2	B_2	30
H_3	A_3	B_3	80
H_4	A_4	B_4	35

Delivery /Receiving Point	A_1	A_2	A_3	A_4	The number of start
B_1	3	8	9	6	11
B_2	10	7	12	4	6
B_3	5	3	6	8	16
B_4	7	9	4	1	7
The number of receive	11	6	16	7	

Table 3 Backhaul traffic flow program

	A_1	A_2	A_3	A_4	
B_1			11		11
B_2	1		5		6
B_3	9			7	16
B_4	1	6			7
	11	6	16	7	

According to the Freight traffic flow plan and return programs, organizations loop transport, can get four cycles cyclic route, namely:

Suppose four lines' cycles were x_1, x_2, x_3, x_4, is to be determined according to cycle transport program Transportation scheme, it has the linear equations :

1. $A_1 \Rightarrow B_1 \rightarrow A_3 \Rightarrow B_3 \rightarrow A_1$ (A_1 front line has already appeared,this circuit has been closed, Under similar)
2. $A_1 \Rightarrow B_1 \rightarrow A_3 \Rightarrow B_3 \rightarrow A_4 \Rightarrow B_4 \rightarrow A_1$
3. $A_1 \Rightarrow B_1 \rightarrow A_3 \Rightarrow B_3 \rightarrow A_4 \Rightarrow B_4 \rightarrow A_2 \Rightarrow B_2 \rightarrow A_1$
4. $A_2 \Rightarrow B_2 \rightarrow A_3 \Rightarrow B_3 \rightarrow A_4 \Rightarrow B_4 \rightarrow A_1$

$$\begin{cases} x_1 + x_2 + x_3 = 11 \\ x_3 + x_4 = 6 \\ x_1 + x_2 + x_3 + x_4 = 16 \\ x_2 + x_3 + x_4 = 7 \end{cases}$$

By the parallel LU decomposition algorithm can get

$$(x_1 \quad x_2 \quad x_3 \quad x_4)^T = (9 \quad 1 \quad 1 \quad 5)$$

In order to fulfill the transportation requirements (11, 6, 16, 7), a specific plan can be implemented. This plan involves four line cycles, with the following frequencies: nine times, one time, one time, and five times respectively.

This study focuses on addressing the circular transportation problem by utilizing the parallel LU decomposition algorithm to solve linear equations. The main objective is to improve the efficiency of solving linear equations by enhancing the speed of computation.

QR Decomposition

Introduction

Let A be an m × n matrix with linearly independent columns. Then A can be factored as A = QR, where Q is an m × n matrix with orthonormal columns and R is an invertible upper triangular matrix.

Applications

Solving Systems of Linear Equations

Least Squares Fitting

Eigenvalue Computation

Theory and mathematical formulation

Consider the *Gram-Schmidt Process*, with the vectors to be considered in the process as columns of the matrix A. That is,

$$A = \begin{bmatrix} a_1 & | & a_2 & | & \cdots & | & a_n \end{bmatrix}$$

$$u_1 = a_1, \quad e_1 = \frac{u_1}{\|u_1\|}$$

$$u_2 = a_2 - (a_2 \cdot e_1)e_1, \quad e_2 = \frac{u_2}{\|u_2\|}$$

$$u_{k+1} = a_{k+1} - (a_{k+1} \cdot e_1)e_1 - \cdots - (a_{k+1} \cdot e_k)e_k, \quad e_{k+1} = \frac{u_{k+1}}{\|u_{k+1}\|}$$

where $\|v\|$ is the norm.

The resulting QR factorization is

$$A = \begin{bmatrix} a_1 & | & a_2 & | & \cdots & | & a_n \end{bmatrix} = \begin{bmatrix} e_1 & | & e_2 & | & \cdots & | & e_n \end{bmatrix} \begin{bmatrix} a_1 \cdot e_1 & a_2 \cdot e_1 & \cdots & a_n \cdot e_1 \\ 0 & a_2 \cdot e_2 & \cdots & a_n \cdot e_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n \cdot e_n \end{bmatrix} = QR$$

where each column $\mathbf{q}_i$ $(1 \leq i \leq n)$ is a unit vector and $\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n$ are orthogonal to each other. In other words, $Q^T Q = I$, where Q^T represents the transpose of Q, and I is the identity matrix.

Applications

2.3.2 Solving Ax=B

$$A = QR, Q^T = Q^-1, Ax = B$$

$$QRx = B$$

$$x = Q^T B$$

$$x = R^-1 Q^T B$$

2.3.3 Least Square Fitting

$$Y = X\beta$$

The solutions to the previous least squares problem are given by the nxn matrix equation, also known as the normal equation:

$$(X^T X)\beta = X^T Y$$

where X^T is the transpose of X.
($(X^T X)^{-1} X^T$ is called the Moore-Penrose pseudo-inverse of X)

Python Code:

```
Q, R = np.linalg.qr(X_train)
Rinv_Qt = np.dot(np.linalg.inv(R),Q.T)
beta = np.dot(Rinv_Qt,y_train)

y_pred = np.dot(X_test, beta)              # predicted y=XB

mean_sq_error = mean_squared_error(y_pred,y_test)

print("Mean sq. error is",mean_sq_error)
```

2.3.4 Eigenvalues

Here's the step-by-step procedure to compute eigenvalues using QR decomposition:

Input: Square matrix A of size $n \times n$

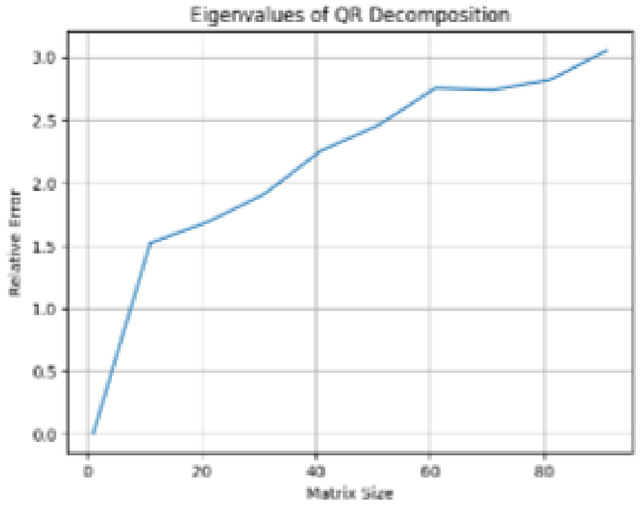
Step 1: Perform QR decomposition on matrix A to obtain matrices Q and R.
Initialize $Q = A$ and $R =$ None
for $i = 1$ to n **do**
    Set $v = Q[:, i]$
    for $j = 1$ to $i - 1$ **do**
        Compute $R[j, i] = Q[:, j]^T \cdot Q[:, i]$
        Update $v = v - R[j, i] \cdot Q[:, j]$
    end for
    Compute $R[i, i] = \|v\|$, the norm of v
    Set $Q[:, i] = v / R[i, i]$
end for
Step 2: Compute the product $B = R \cdot Q$
Step 3: Repeat until convergence or a specified number of iterations:
for each iteration **do**
    Perform QR decomposition on matrix B to obtain matrices Q and R
    Update $B = R \cdot Q$
end for
Step 4: The eigenvalues of the original matrix A are given by the diagonal elements of the converged upper triangular matrix B

Computation Time

QR decomposition tends to be computationally more expensive than LU de-composition, especially for large matrices. QR decomposition involves orthog-onalization and triangularization of the matrix, which can be computationally intensive operations. On the other hand, LU decomposition involves the factor-ization of the matrix into lower and upper triangular matrices.

Accuracy in calculation of Eigenvalues

It is visible that the absolute error in eigenvalues computation using QR decom-position is close to 3 even for very large matrices of the order 100. Hence this method is accurate.



Analysis

The choice of the QR decomposition algorithm depends on the specific requirements of the problem and the properties of the matrix.

- Computational Complexity: The computational complexity of QR decomposition depends on the size of the matrix being decomposed. For an n x n matrix, the complexity is typically $O(n^3)$. This is because the algorithm involves performing several matrix multiplications and matrix factorizations.
- QR decomposition may require additional memory to store intermediate matrices, such as the Q and R matrices.
- QR decomposition can be sensitive to numerical stability issues, especially when dealing with ill-conditioned or nearly singular matrices. Care should be taken to use appropriate techniques, such as pivoting, to enhance stability and accuracy.

Principal component analysis using QR decomposition: Sharma, A., Paliwal, K.K., Imoto, S. et al. Principal component analysis using QR decomposition. Int. J. Mach. Learn. & Cyber. 4, 679–683 (2013). https://doi.org/10.1007/s13042-012-0131-7

PCA using QR decomposition

QR-based PCA refers to Principal Component Analysis (PCA) algorithms that utilize the QR decomposition (also known as QR factorization) as part of the computation.

PCA is a widely used technique in statistics and machine learning for dimensionality reduction and data analysis. Similar to the singular value decomposition (SVD) based PCA method this method is numerically stable.

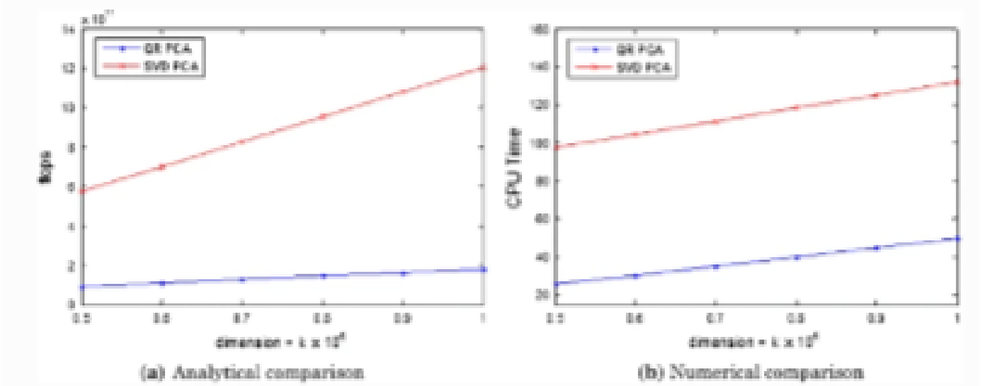
2.6.1 Process

1. Start with a rectangular matrix $HR^{d\times n}$, where d is much larger than n.
2. Use the economic QR decomposition to factorize H into two matrices: $Q_1R^{d\times t}$ and $R_1R^{t\times n}$, where t is the rank of Σ_x.
3. Substitute the QR decomposition into the expression $\Sigma_x = HH^T$, resulting in $\Sigma_x = Q_1R_1R_1^TQ_1^T$.
4. Perform singular value decomposition (SVD) on the matrix R_1 to obtain $R_1 = U_1D_1V^T$, where $U_1R^{n\times t}$, $VR^{t\times t}$, and $D_1R^{t\times t}$.
5. Substitute the SVD result into the expression $\Sigma_x = Q_1VD_1U_1^TU_1D_1V^TQ_1^T$, simplifying it to $\Sigma_x = Q_1V\Lambda V^TQ_1^T$, where $\Lambda = D_1^2$.
6. The matrix Q_1V is an orthogonal matrix that diagonalizes Σ_x, with Λ being the eigenvalue matrix.
7. Select the h largest diagonal entries of D_1 and extract the corresponding column vectors from V to form the matrix $V_hR^{t\times h}$.
8. The PCA transform is given by $\Phi = Q_1V_hR^{d\times h}$, where Φ represents the eigenvector matrix.

By following these steps, the QR-based PCA method allows for the projection of data into a lower-dimensional space while retaining the most significant variations. This approach is numerically stable and particularly useful for working with rectangular matrices.

Analysis

The paper analyses the difference between computational complexities of QR and LU. Figure 1a shows the analytical comparison of flops between QR based PCA
and SVD based PCA methods. Figure 1b shows the numerical comparison of their times when implemented on Matlab software. It can be seen from this figure that the proposed method is computationally more efficient than the SVD based PCA method both analytically and Matlab-wise.



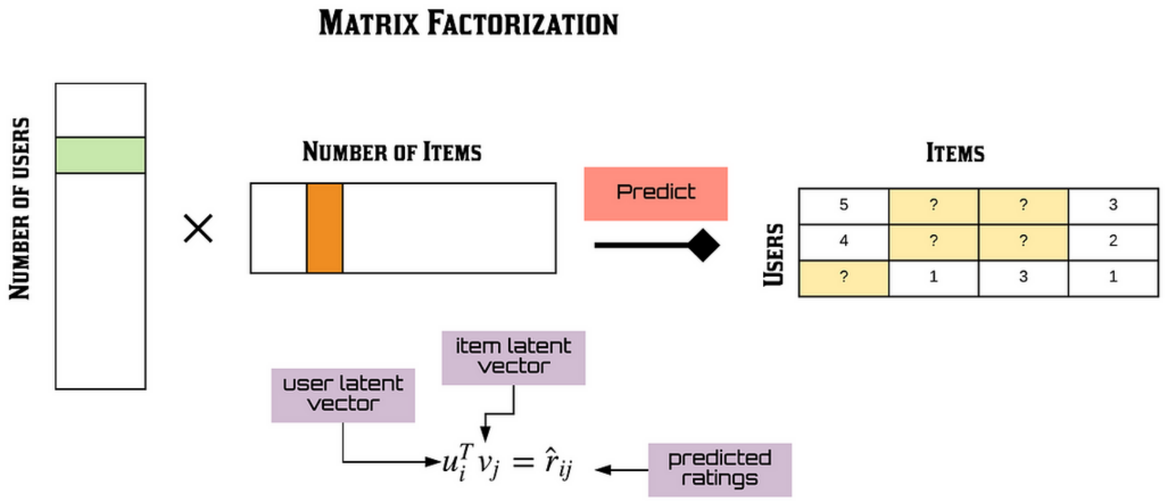
A comparison of cpu time between QR based PCA and SVD based PCA methods

The proposed method offers a computational advantage by utilizing the QR decomposition of the covariance matrix. By decomposing the rectangular matrix H into an orthogonal matrix and an upper triangular matrix, the SVD procedure can be applied to the upper triangular matrix to compute the leading eigenvectors.

Applications of matrix decomposition

Recommender system using SVD

Recommender systems provide personalized recommendations to users and are widely used in online platforms. SVD, a matrix factorization technique, can be applied to build effective recommender systems. Here are the key steps:

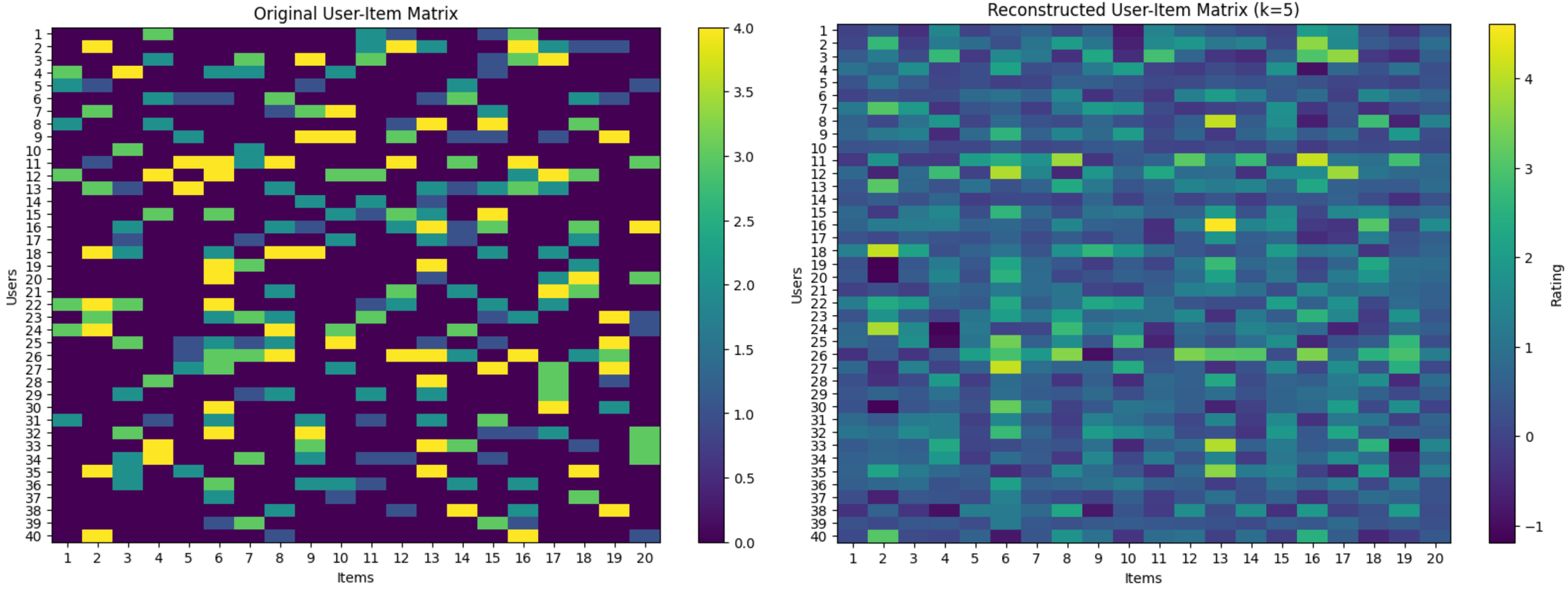


Steps Required

1. Data Representation: Represent user-item interaction data as a matrix, where each entry represents user preferences or ratings.
2. Matrix Factorization with SVD: Decompose the user-item matrix into three separate matrices: users, singular values, and items.
3. Dimensionality Reduction: Retain the top k singular values and their corresponding singular vectors to reduce the dimensionality of the matrix.
4. Latent Factor Calculation: The reduced matrices represent the latent factors in the user-item matrix, capturing underlying characteristics and preferences.
5. Recommendation Generation: Calculate similarity between user and item latent representations to generate recommendations.
6. Rating Prediction: Predict ratings for items that users have not interacted with using the reduced matrices.
7. Evaluation and Iteration: Evaluate system performance using metrics and refine the recommender system as needed.

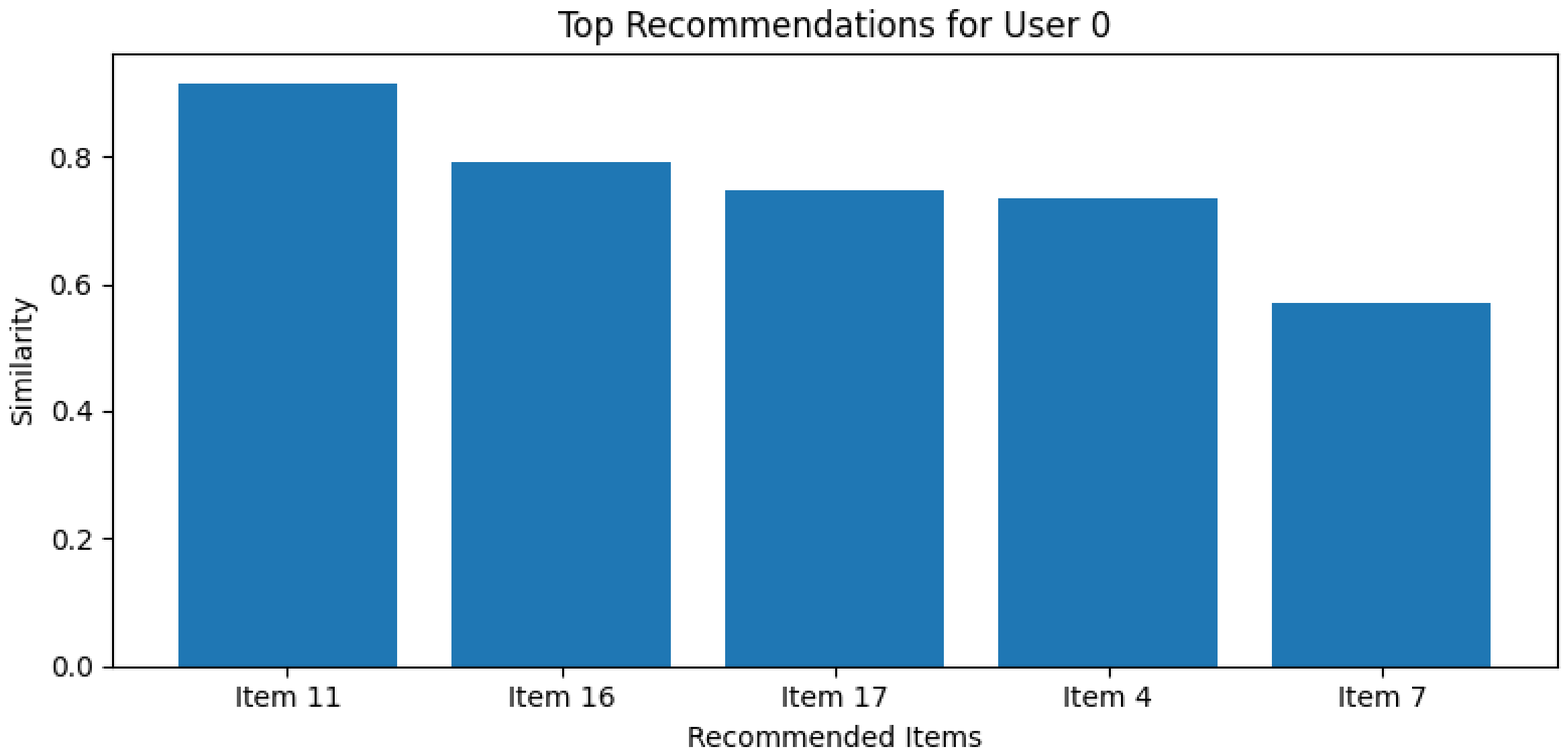
Recommender system using SVD

Predictions



Recommender system using SVD

Results

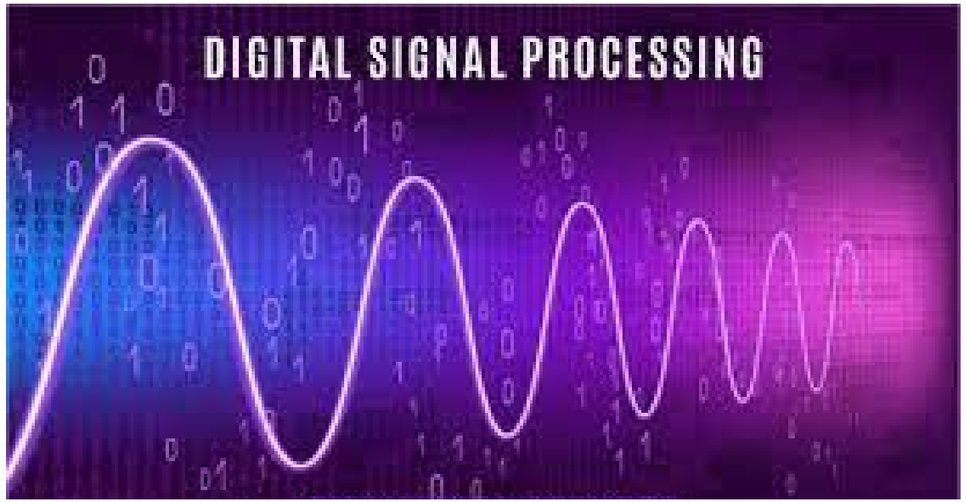


Recommendations for a user based on randmly generated data

Applications of matrix decomposition

Digital Signal Processing using PCA

Digital Signal Processing (DSP) involves manipulating and analyzing digital signals using mathematical techniques. PCA, a matrix decomposition method, is widely used in DSP.

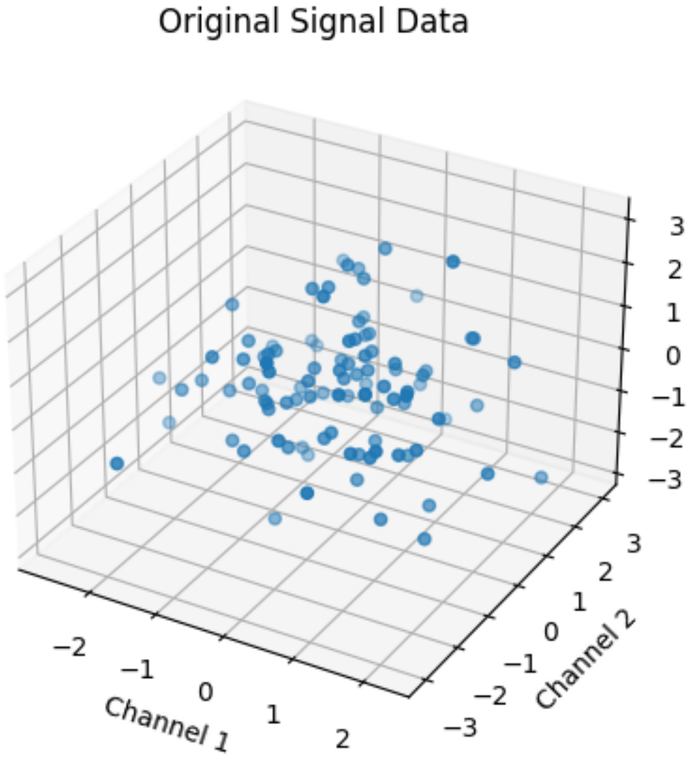


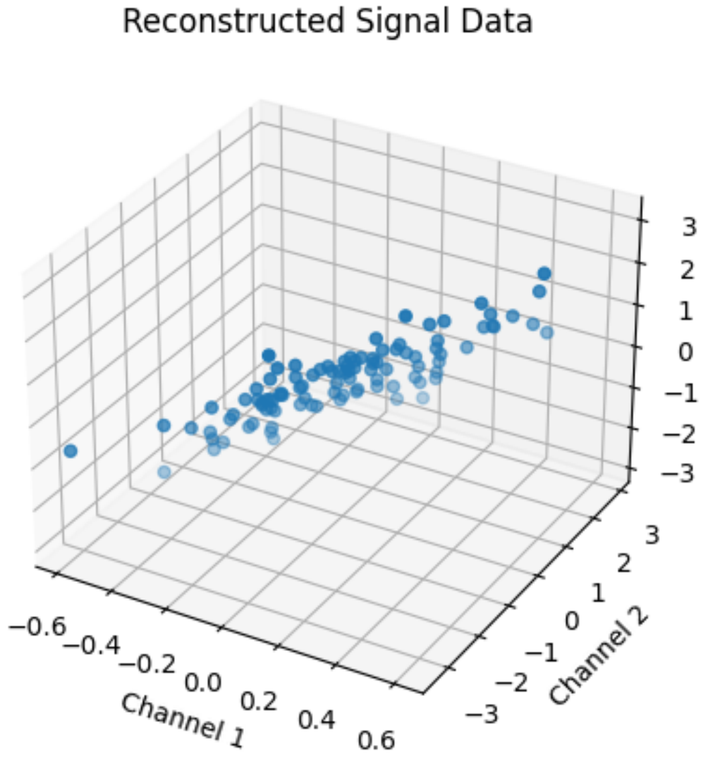
Steps Required

1. Signal Representation: Represent digital signals as matrices, with rows or columns corresponding to different aspects of the signal.
2. Matrix Formation: Construct the signal matrix by arranging the samples appropriately.
3. Data Centering: Center the signal matrix by subtracting the mean value of each column or row.
4. Principal Component Analysis: Apply PCA to the centered signal matrix to transform correlated variables into uncorrelated principal components.
5. Eigendecomposition: Decompose the covariance matrix of the centered data to obtain eigenvectors and eigenvalues.
6. Eigenvalue and Eigenvector Selection: Select the top K eigenvalues and their corresponding eigenvectors to retain the most significant principal components.
7. Dimensionality Reduction: Transform the original signal matrix into a lower-dimensional space using the selected eigenvectors.
8. Reconstruction: Reconstruct the original signal from the reduced-dimension representation.

Digital Signal Processing using PCA

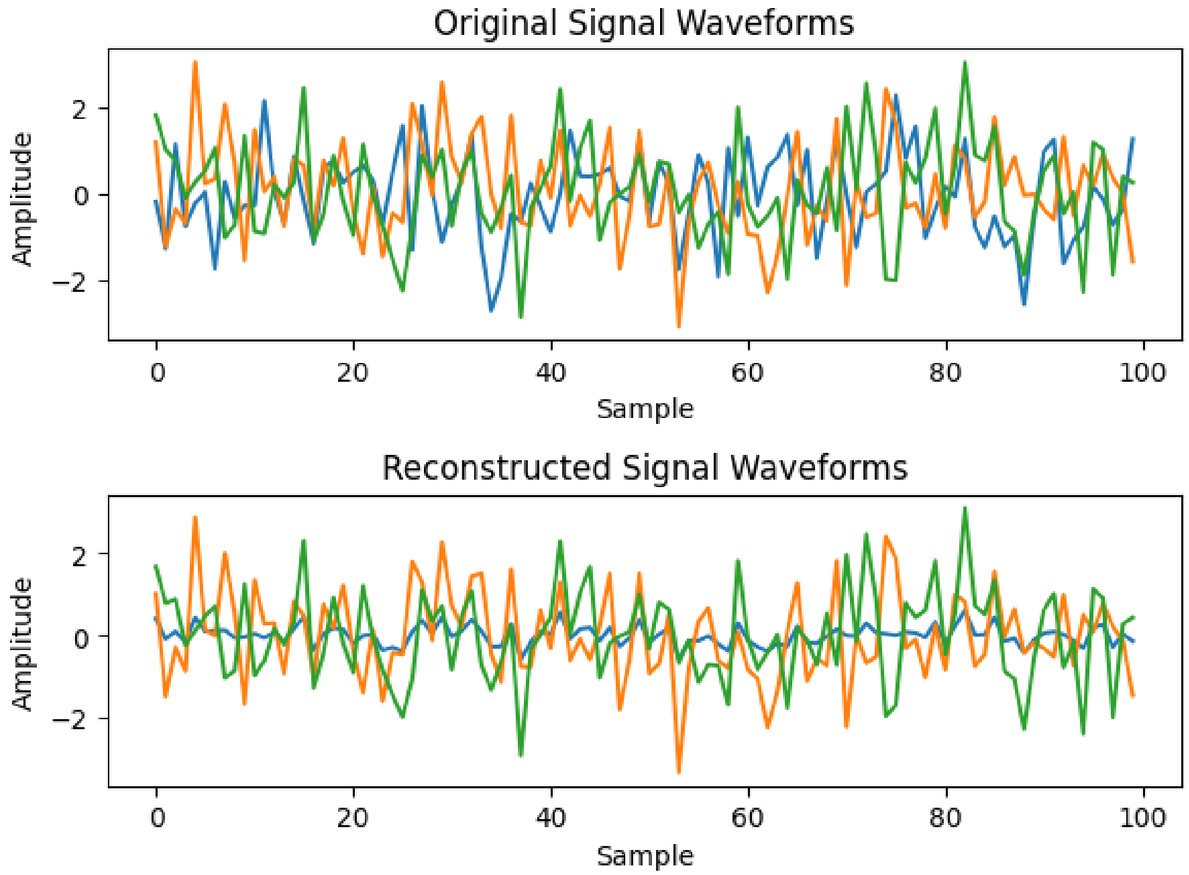
Results





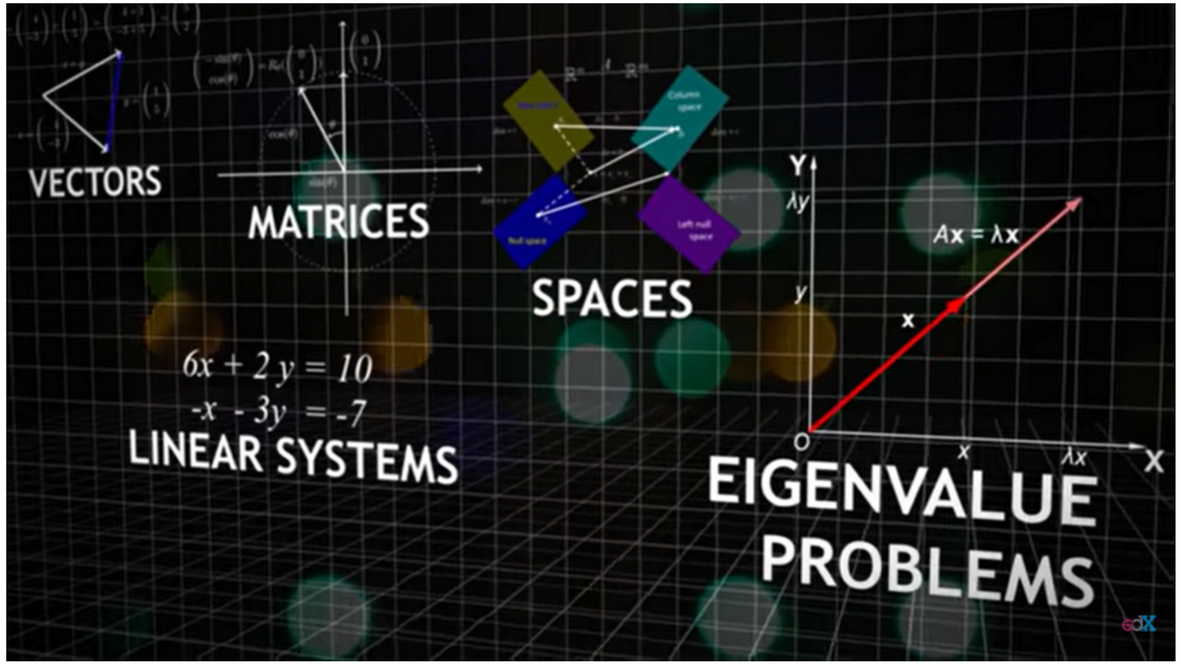
Digital Signal Processing using PCA

Results



Other Applications

Apart from recommender systems and digital signal processing, matrix decomposition has various other applications in fields such as mathematics, computer science, physics, and engineering.
Some other applications include solving systems of linear equations, eigenvalue problems, image processing, graph theory and network analysis, control systems, and data compression.
Matrix decomposition techniques provide powerful tools for analyzing complex data structures, extracting meaningful insights, reducing dimensionality, and enhancing computational efficiency in a wide range of applications.



Thank you