GHARDA INSTITUTE OF TECHNOLOGY

Department of Computer Engineering

Machine Learning Lab BE Computer (Semester-VII)

Experiment No.7: Singular Value Decomposition (SVD)

Aim : To study, understand and implement a **Singular Value Decomposition (SVD)** on a matrix, decompose it into its constituent matrices, and then reconstruct the original matrix.

Theory :

Singular Value Decomposition (SVD) is a fundamental matrix decomposition technique in linear algebra. For any matrix A of size m×n, SVD expresses it as the product of three matrices:

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

where:

- U is an orthogonal matrix containing the left singular vectors.
- Σ is a diagonal matrix with the singular values of A (which represent the magnitude of each component).
- V^T is the transpose of an orthogonal matrix containing the right singular vectors.

This decomposition helps in many applications such as dimensionality reduction, image compression, and solving linear systems. SVD can be used for square as well as non-square matrices.

Description

1. **Input Matrix**: The user inputs the dimensions and elements of the matrix.
2. **SVD Decomposition**: The matrix is decomposed into three components: U (left singular vectors), Σ (singular values), and V^T (right singular vectors).
3. **Reconstruction**: The original matrix is reconstructed by multiplying U, Σ, and V^T using the matrix dot product.
4. **Output**: The program prints the input matrix, U matrix, singular values, V^T matrix, and the reconstructed matrix.

Applications :

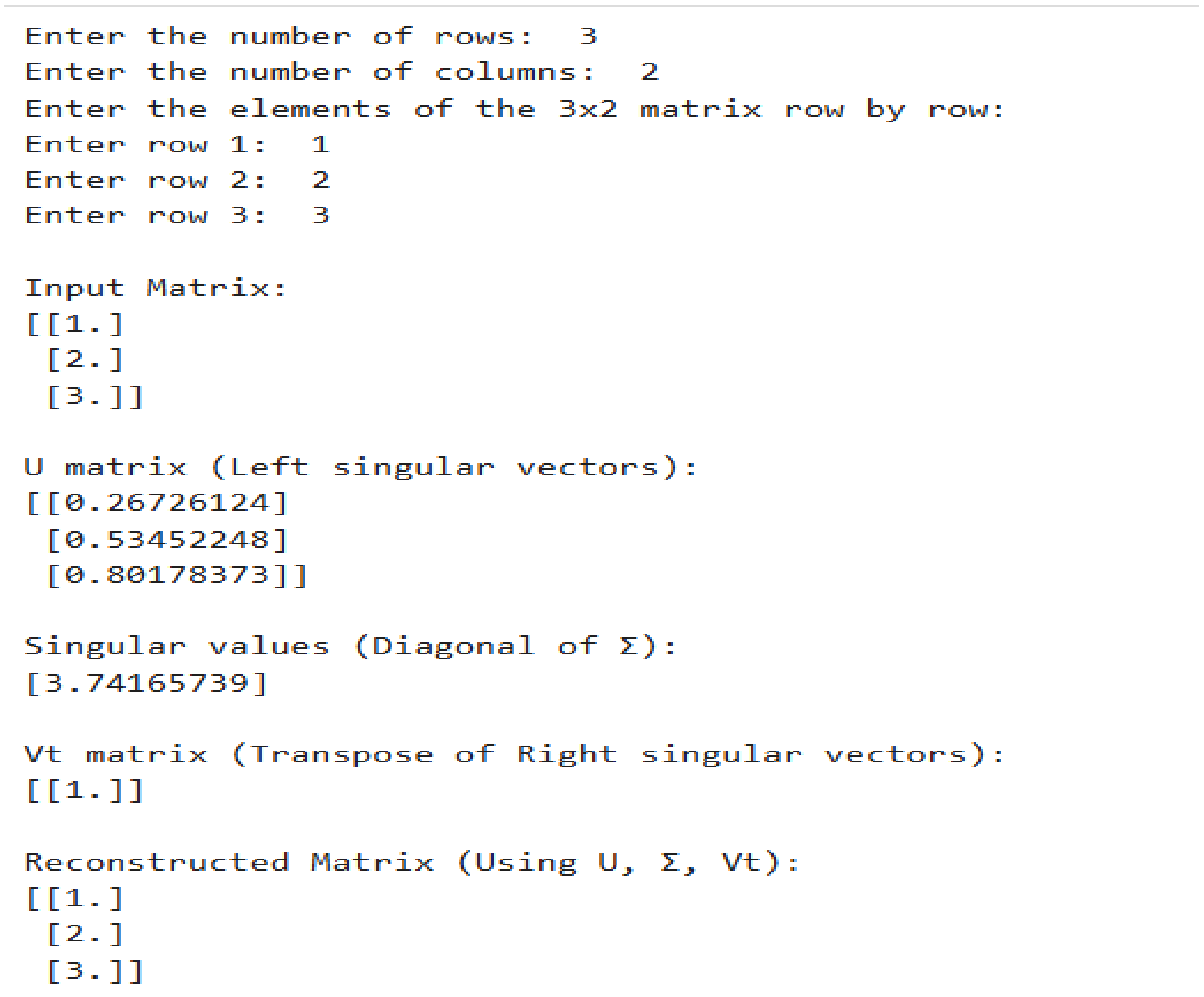
- **Dimensionality Reduction**: SVD can reduce the number of features in data while retaining the most important ones (e.g., in PCA).

- **Image Compression**: By approximating an image matrix with a smaller rank, SVD compresses images while maintaining significant detail.
- **Latent Semantic Analysis (LSA)**: SVD is used in Natural Language Processing (NLP) to analyze relationships between terms and documents.
- **Noise Reduction**: By retaining only the largest singular values, we can remove noise and reconstruct a cleaner version of the data.

Program Code :

```
import numpy as np
 # Function to compute SVD and reconstruct the matrix
         def svd_decomposition(matrix):
 # Compute SVD
          U, S, Vt = np.linalg.svd(matrix, full_matrices=False)  # Use reduced SVD
 # Create the diagonal matrix of singular values
          Sigma = np.diag(S)  # No need to manually create a zero matrix
  # Reconstruct the original matrix using U, Sigma, and Vt
          A_reconstructed = np.dot(U, np.dot(Sigma, Vt))
        return U, S, Vt, A_reconstructed
# Main program
         if __name__ == "__main__":
# Input matrix dimensions
          rows = int(input("Enter the number of rows: "))
          cols = int(input("Enter the number of columns: "))
 # Input matrix elements
   print(f"Enter the elements of the {rows}x{cols} matrix row by row:")
   matrix = []
   for i in range(rows):
      row = list(map(float, input(f"Enter row {i+1}: ").split()))
      matrix.append(row)
  # Convert the list to a NumPy array
         matrix = np.array(matrix)
  # Perform SVD
         U, S, Vt, A_reconstructed = svd_decomposition(matrix)
  # Output results
         print("\nInput Matrix:")
         print(matrix)
   print("\nU matrix (Left singular vectors):")
   print(U)
   print("\nSingular values (Diagonal of Σ):")
   print(S)
   print("\nVt matrix (Transpose of Right singular vectors):")
   print(Vt)
   print("\nReconstructed Matrix (Using U, Σ, Vt):")
   print(A_reconstructed)
```

Output :



```
Enter the number of rows:  3
Enter the number of columns:  2
Enter the elements of the 3x2 matrix row by row:
Enter row 1:  1
Enter row 2:  2
Enter row 3:  3

Input Matrix:
[[1.]
 [2.]
 [3.]]

U matrix (Left singular vectors):
[[0.26726124]
 [0.53452248]
 [0.80178373]]

Singular values (Diagonal of Σ):
[3.74165739]

Vt matrix (Transpose of Right singular vectors):
[[1.]]

Reconstructed Matrix (Using U, Σ, Vt):
[[1.]
 [2.]
 [3.]]
```

Conclusion :
The implementation of SVD demonstrates its power in decomposing a matrix into its fundamental components. The reconstructed matrix validates that SVD captures the essential structure of the original matrix. This technique has far-reaching implications in data science, particularly for reducing data complexity, removing noise, and compressing information.

References :

1. https://press.jhu.edu/books/title/11694/matrix-computations
2. https://math.mit.edu/~gs/linearalgebra/
3. https://epubs.siam.org/doi/book/10.1137/1.9781611971484
4. https://onlinelibrary.wiley.com/doi/full/10.1002/wics.101