

# SVD

November 30, 2025

SVD Code Caleb Smith, Paige Swanson, Grace Tobin

```
[17]: import numpy as np
```

```
[18]: # returns the transpose of A times A
def ATxA(A):
    return np.matrix(A).transpose() * A
```

```
[19]: # returns the transpose of A times A
def AxAT(A):
    return A * np.matrix(A).transpose()
```

```
[20]: def getEigenvalues(A):
    # get eigenvalues with corresponding eigenvectors
    eigenvalues, eigenvectors = np.linalg.eigh(A)

    # Order eigenvalues and corresponding vectors in descending order
    order = np.argsort(eigenvalues)[::-1]
    eigenvalues = eigenvalues[order]
    eigenvectors = eigenvectors[:, order]
    return eigenvalues, eigenvectors
```

```
[21]: def SingularValues(eigenvalues):
    # return singular values the square root of the eigenvalues
    return np.sqrt(eigenvalues)
```

```
[22]: def OrthoMatrix(eigenvectors):
    # normalize
    for i in range(eigenvectors.shape[1]):
        norm = np.linalg.norm(eigenvectors[:, i])
        if norm > 0:
            eigenvectors[:, i] /= norm

    return np.matrix(eigenvectors)
```

```
[23]: def SVD(A):
    "Input is a numpy matrix. Returns the orthogonal matrix U, the singular
    ↪value matrix S, and the orthogonal matrix V^T."
    # compute  $A^T * A$  and  $A * A^T$ 
    ATA = ATxA(A)
    AAT = AxAT(A)

    # get eigenvalues with corresponding eigenvectors
    eigenvaluesATA, eigenvectorsATA = getEigenvalues(ATA)
    eigenvaluesAAT, eigenvectorsAAT = getEigenvalues(AAT)

    # Singular Value Matrix
    S_vec = SingularValues(eigenvaluesATA)
    S_mat = np.zeros(A.shape)
    for i in range(S_vec.size):
        S_mat[i,i] = S_vec[i]

    # V transpose matrix
    V_mat = OrthoMatrix(eigenvectorsATA).transpose()

    # U matrix
    U_mat = OrthoMatrix(eigenvectorsAAT)
    return U_mat, S_mat, V_mat
```

Example

```
[26]: A = np.matrix([[1,0,1],[0,1,0],[0,1,1],[0,1,0],[1,1,0]])
U_mat, S_mat, V_mat = SVD(A)
print(U_mat)
print(S_mat)
print(V_mat)
print(np.round(U_mat*S_mat*V_mat, 2))
```

```
[[-3.65148372e-01  8.16496581e-01  7.84245057e-17 -7.70698864e-02
  4.40522681e-01]
 [-3.65148372e-01 -4.08248290e-01  4.44089210e-16  6.19457630e-01
  5.62380871e-01]
 [-5.47722558e-01 -7.02491948e-17  7.07106781e-01  7.70698864e-02
 -4.40522681e-01]
 [-3.65148372e-01 -4.08248290e-01 -1.65188709e-16 -7.73597403e-01
  3.18664491e-01]
 [-5.47722558e-01 -1.47380436e-17 -7.07106781e-01  7.70698864e-02
 -4.40522681e-01]]

[[2.23606798  0.          0.          ]
 [0.          1.41421356  0.          ]
 [0.          0.          1.          ]
 [0.          0.          0.          ]
 [0.          0.          0.          ]]
```

```
[[-4.08248290e-01 -8.16496581e-01 -4.08248290e-01]
 [ 5.77350269e-01 -5.77350269e-01  5.77350269e-01]
 [-7.07106781e-01  2.59129669e-16  7.07106781e-01]]
[[ 1. -0.  1.]
 [-0.  1.  0.]
 [ 0.  1.  1.]
 [ 0.  1. -0.]
 [ 1.  1.  0.]]
```