# CS-663 Assignment 4 Q1

Soham Naha (193079003)
Akshay Bajpai (193079002)
Mohit Agarwala (19307R004)

November 6, 2020

## 1

**Given a matrix $A$ of size $m \times n$, write a MATLAB routine called MySVD which takes this matrix as input and outputs the left and right singular vectors (i.e. column vectors of $U$ and $V$ under usual notation) and the singular values (i.e. diagonal entries of $S$) of $A$. You are not allowed to use the svd or svds functions of MATLAB directly. You should use only the eigenvalue decomposition routines eig or eigs for this task. Cross-check your answer by verifying that $A = USV^T$ based on your computation. [15 points]**

In order to use eigen value decomposition (EVD) to calculate the singular value decomposition (SVD) components$(U, \Sigma, V^T)$, for a given matrix $A$, we need to first calculate either of $AA^T$ or $A^TA$, calculate their eigen vectors and eigen values.
Here, we calculate first the eigen vectors$(V)$ and eigen values$(\Lambda)$ of $A^TA$, sort in descending order and calculate the other unitary vector matrix $U$ from the relation $AV = \Sigma U$.

**mySVD.py**

```
import numpy as np

def SVD(matrix):
    """Calculates SVD using eigen value decomposition method and
        returns the U,S,V_T
    :param matrix: any random matrix
    :output U,S,V_T : the singular value components
    """
    # calculating A'A
    temp1 = np.matmul(random_matrix.T,random_matrix)
    # calculating eigen values and vectors of A'A
    lambda_,V = np.linalg.eig(temp1)

    # sorting the singular values and in descending order
    sorted_lambda = np.sort(lambda_)[::-1]

    idx = np.argsort(lambda_)[::-1]
    V_sort = np.zeros_like(V)
    _,c = V_sort.shape

    for i in range(c):
        V_sort[:,i] = V[:,idx[i]]

    U_sort = np.matmul(random_matrix,V_sort)
    _,c = U_sort.shape
    for i in range(c):
        U_sort[:,i] = U_sort[:,i]/np.sqrt(np.sum(np.square(U_sort[:,i])))

    sigma = np.sqrt(sorted_lambda)
    return U_sort,sigma,V_sort.T


if __name__=="__main__":
    m,n = 4,4
```

```
    matrix = np.random.randint(1,10,(m,n))
    print("Input matrix of size {}x{}:\n {}".format(m,n,matrix))
    print("****************************************")
    U,S,V_T = SVD(matrix)
    print("U matrix\n",U)
    print("Singular Values\n",S)
    print("V'(V transpose) Matrix\n",V_T)
    print("*****************************************")
    print("calculated matrix: \n",np.matmul(np.matmul(U,np.diag(S)),V_T))
```

### Sample Program Output

```
Input matrix of size 4x4:
 [[6 8 6 2]
 [5 9 2 2]
 [9 1 7 9]
 [6 4 9 3]]
****************************************
U matrix
 [[ 0.49902061 -0.40367479 -0.17508303 -0.74657286]
 [ 0.39465136 -0.60441194  0.50622627  0.4718808 ]
 [ 0.57958458  0.68105716  0.4395442  -0.08392709]
 [ 0.50922535  0.08884829 -0.72102829  0.46142575]]
Singular Values
 [22.48631246  9.09254006  4.43526137  0.14115208]
V'(V transpose) Matrix
 [[ 0.58875788  0.45185323  0.55249373  0.37939885]
 [ 0.13401049 -0.8394415   0.21293963  0.48170102]
 [ 0.25034854  0.16026185 -0.77796787  0.55354109]
 [-0.7567899   0.255896    0.21017466  0.56357156]]
*****************************************
calculated matrix:
 [[6. 8. 6. 2.]
 [5. 9. 2. 2.]
 [9. 1. 7. 9.]
 [6. 4. 9. 3.]]
```