SUYASH PRATAP SINGH (181B226)

TASKS:-

For the given documents:

1. Create a term incidence matrix(try to write a generic code for generating TI matrix)
2. Calculate sVD .
3. Check the Result is same as original or not
4. Compute a 4 rank approximation of the matrix.

c1: Human machine interface for Lab ABC computer applications c2: A survey of user opinion of computer system response time c3: The EPS user interface management system c4: System and human system engineering testing of EPS c5: Relation of user-perceived response time to error measurement m1: The generation of random, binary, unordered trees m2: The intersection graph of paths in trees m3: Graph minors IV: Widths of trees and well-quasi-ordering m4: Graph minors: A survey

In [1]:

```
import sklearn
import sklearn.feature_extraction
import numpy as np
from scipy.linalg import svd
from numpy import zeros
from numpy import diag
from numpy import dot
```

Create a term incidence matrix

In [2]:

```
suy = sklearn.feature_extraction.text.CountVectorizer(min_df=1)
teju=[]
vin=int(input("How many documents you want to enter\n"))
for i in range(vin):
  s=input()
  teju.append(s)
```

```
How many documents you want to enter
9
Human machine interface for Lab ABC computer applications
 A survey of user opinion of computer system response time
The EPS user interface management system
 System and human system engineering testing of EPS
Relation of user-perceived response time to error measurement
The generation of random, binary, unordered trees
The intersection graph of paths in trees
Graph minors IV: Widths of trees and well-quasi-ordering
Graph minors: A survey
```

In [3]:
```
Z = suy.fit_transform(teju).toarray()
print('{0}'.format(Z))
print('suy.vocabulary_: {0}'.format(suy.vocabulary_))
```

```
[[1 0 1 0 1 0 0 0 1 0 0 1 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 1 0 0 0 0 0 0 1 1 1 0 0 1 0
  0 0 1 0 0]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0
  0 0 1 0 0]
 [0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 1 0 0 0
  0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 1 1 0 0 0 0 1 1
  0 0 1 0 0]
 [0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0
  1 1 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0
  1 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0
  1 0 0 1 1]
 [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
  0 0 0 0 0]]
suy.vocabulary_: {'human': 11, 'machine': 17, 'interface': 13, 'for': 8, 'la
b': 16, 'abc': 0, 'computer': 4, 'applications': 2, 'survey': 30, 'of': 21,
'user': 38, 'opinion': 22, 'system': 31, 'response': 29, 'time': 34, 'the': 3
3, 'eps': 6, 'management': 18, 'and': 1, 'engineering': 5, 'testing': 32, 're
lation': 28, 'perceived': 25, 'to': 35, 'error': 7, 'measurement': 19, 'gener
ation': 9, 'random': 27, 'binary': 3, 'unordered': 37, 'trees': 36, 'intersec
tion': 14, 'graph': 10, 'paths': 24, 'in': 12, 'minors': 20, 'iv': 15, 'width
s': 40, 'well': 39, 'quasi': 26, 'ordering': 23}
```

In [5]:
```
for row in Z:
    for j in range(len(row)):
        if(row[j]>1):
            row[i]>1
print(Z)
```

```
[[1 0 1 0 1 0 0 0 1 0 0 1 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 1 0 0 0 0 0 0 1 1 1 0 0 1 0
  0 0 1 0 0]
 [0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0
  0 0 1 0 0]
 [0 1 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 2 1 0 0 0
  0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 1 1 0 0 0 0 1 1
  0 0 1 0 0]
 [0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0
  1 1 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0
  1 0 0 0 0]
 [0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0
  1 0 0 1 1]
 [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
  0 0 0 0 0]]
```

Apply SVD

```
In [6]: U,S,V = np.linalg.svd(Z)
        S = np.diag(S)
        V = V[:9,:]
```

```
In [7]: print(U.shape , S.shape , V.shape)

        (9, 9) (9, 9) (9, 41)
```

```
In [8]: S
```

```
Out[8]: array([[4.60598869, 0.        , 0.        , 0.        , 0.        ,
                0.        , 0.        , 0.        , 0.        ],
               [0.        , 3.34993222, 0.        , 0.        , 0.        ,
                0.        , 0.        , 0.        , 0.        ],
               [0.        , 0.        , 3.09443767, 0.        , 0.        ,
                0.        , 0.        , 0.        , 0.        ],
               [0.        , 0.        , 0.        , 2.72832464, 0.        ,
                0.        , 0.        , 0.        , 0.        ],
               [0.        , 0.        , 0.        , 0.        , 2.6542251 ,
                0.        , 0.        , 0.        , 0.        ],
               [0.        , 0.        , 0.        , 0.        , 0.        ,
                2.14167375, 0.        , 0.        , 0.        ],
               [0.        , 0.        , 0.        , 0.        , 0.        ,
                0.        , 2.05835898, 0.        , 0.        ],
               [0.        , 0.        , 0.        , 0.        , 0.        ,
                0.        , 0.        , 1.89716432, 0.        ],
               [0.        , 0.        , 0.        , 0.        , 0.        ,
                0.        , 0.        , 0.        , 1.44075363]])
```

```
In [9]: U
```

```
Out[9]: array([[-0.09333534,  0.23129924,  0.37970627, -0.87006897,  0.13300341,
                -0.02943345, -0.10065601, -0.08588579, -0.02297975],
               [-0.58434714,  0.28952872, -0.25501977, -0.02505255,  0.10029583,
                 0.61123465,  0.18851088,  0.2083633 ,  0.21547659],
               [-0.22674715,  0.21247752,  0.27202544,  0.06316425, -0.31010962,
                -0.49774712,  0.4498949 ,  0.52097844,  0.0877915 ],
               [-0.42234451,  0.24325052,  0.58123846,  0.44585954,  0.08278368,
                -0.0130243 , -0.2596213 , -0.35137835, -0.16713077],
               [-0.36655947,  0.25552383, -0.60513248, -0.0740426 ,  0.10793109,
                -0.55497424, -0.19506018, -0.20836907, -0.16601938],
               [-0.26443017, -0.28619692, -0.02697676, -0.11741757, -0.62499482,
                 0.11729726, -0.55266225,  0.28259397, -0.20971716],
               [-0.28993495, -0.39986778, -0.00863379, -0.13475064, -0.3889193 ,
                -0.01285304,  0.45698463, -0.60167877,  0.12464145],
               [-0.35237971, -0.64952124,  0.10916483, -0.01636442,  0.528746  ,
                -0.2006151 , -0.10804789,  0.23302116,  0.23658822],
               [-0.08668845, -0.17141495, -0.0068928 , -0.04332643,  0.19008294,
                 0.12424727,  0.34717437,  0.12136665, -0.87997075]])
```

In [10]:

```
V
```

```
Out[10]: array([[-2.02639097e-02, -1.68199331e-01, -2.02639097e-02,
                 -5.74100773e-02, -1.47130731e-01, -9.16946479e-02,
                 -1.40923416e-01, -7.95832322e-02, -2.02639097e-02,
                 -5.74100773e-02, -1.58272881e-01, -1.11958558e-01,
                 -6.29473858e-02, -6.94926776e-02, -6.29473858e-02,
                 -7.65046834e-02, -2.02639097e-02, -2.02639097e-02,
                 -4.92287679e-02, -7.95832322e-02, -9.53254955e-02,
                 -6.21873670e-01, -1.26866822e-01, -7.65046834e-02,
                 -6.29473858e-02, -7.95832322e-02, -7.65046834e-02,
                 -5.74100773e-02, -7.95832322e-02, -2.06450054e-01,
                 -1.45687634e-01, -3.59484885e-01, -9.16946479e-02,
                 -1.69586231e-01, -2.06450054e-01, -7.95832322e-02,
                 -1.96862146e-01, -5.74100773e-02, -2.55678822e-01,
                 -7.65046834e-02, -7.65046834e-02],
                [ 6.90459449e-02, -1.21277296e-01,  6.90459449e-02,
                 -8.54336446e-02,  1.55474176e-01,  7.26135645e-02,
                  1.36040973e-01,  7.62773131e-02,  6.90459449e-02,
                 -8.54336446e-02, -3.64426468e-01,  1.41659509e-01,
                 -1.19365932e-01,  1.32473353e-01, -1.19365932e-01,
                 -1.93890861e-01,  6.90459449e-02,  6.90459449e-02,
                  6.34274082e-02,  7.62773131e-02, -2.45060536e-01,
                 -7.69430975e-02,  8.64282311e-02, -1.93890861e-01,
                 -1.19365932e-01,  7.62773131e-02, -1.93890861e-01,
                 -8.54336446e-02,  7.62773131e-02,  1.62705544e-01,
                  3.52585554e-02,  2.95082768e-01,  7.26135645e-02,
                 -1.41372168e-01,  1.62705544e-01,  7.62773131e-02,
                 -3.98690437e-01, -8.54336446e-02,  2.26132952e-01,
                 -1.93890861e-01, -1.93890861e-01],
                [ 1.22706066e-01,  2.23111065e-01,  1.22706066e-01,
                 -8.71782363e-03,  4.02937523e-02,  1.87833306e-01,
                  2.75741181e-01, -1.95554911e-01,  1.22706066e-01,
                 -8.71782363e-03,  3.02601776e-02,  3.10539372e-01,
                 -2.79010141e-03,  2.10613941e-01, -2.79010141e-03,
                  3.52777594e-02,  1.22706066e-01,  1.22706066e-01,
                  8.79078753e-02, -1.95554911e-01,  3.30502790e-02,
                 -1.48776398e-01, -8.24123137e-02,  3.52777594e-02,
                 -2.79010141e-03, -1.95554911e-01,  3.52777594e-02,
                 -8.71782363e-03, -1.95554911e-01, -2.77967224e-01,
                 -8.46397940e-02,  3.81162173e-01,  1.87833306e-01,
                  7.63999502e-02, -2.77967224e-01, -1.95554911e-01,
                  2.37698343e-02, -8.71782363e-03, -1.90059349e-01,
                  3.52777594e-02,  3.52777594e-02],
                [-3.18902290e-01,  1.57420825e-01, -3.18902290e-01,
                 -4.30365094e-02, -3.28084684e-01,  1.63418800e-01,
                  1.86570097e-01, -2.71384856e-02, -3.18902290e-01,
                 -4.30365094e-02, -7.12677238e-02, -1.55483490e-01,
                 -4.93895177e-02, -2.95750993e-01, -4.93895177e-02,
                 -5.99797494e-03, -3.18902290e-01, -3.18902290e-01,
                  2.31512972e-02, -2.71384856e-02, -2.18782061e-02,
                  1.94915238e-02, -9.18239424e-03, -5.99797494e-03,
                 -4.93895177e-02, -2.71384856e-02, -5.99797494e-03,
                 -4.30365094e-02, -2.71384856e-02, -3.63208798e-02,
                 -2.50626254e-02,  3.40806503e-01,  1.63418800e-01,
                 -6.92747298e-02, -3.63208798e-02, -2.71384856e-02,
                 -9.84240019e-02, -4.30365094e-02, -1.31695826e-02,
                 -5.99797494e-03, -5.99797494e-03],
                [ 5.01100726e-02,  2.30398574e-01,  5.01100726e-02,
        -2.35471672e-01,  8.78973088e-02,  3.11893960e-02,
        -8.56468208e-02,  4.06638807e-02,  5.01100726e-02,
        -2.35471672e-01,  1.24296029e-01,  8.12994686e-02,
        -1.46528377e-01, -6.67261443e-02, -1.46528377e-01,
         1.99209178e-01,  5.01100726e-02,  5.01100726e-02,
        -1.16836217e-01,  4.06638807e-02,  2.70824407e-01,
        -3.53631214e-02,  3.77872362e-02,  1.99209178e-01,
        -1.46528377e-01,  4.06638807e-02,  1.99209178e-01,
        -2.35471672e-01,  4.06638807e-02,  7.84511169e-02,
         1.09402464e-01, -1.66701886e-02,  3.11893960e-02,
        -4.98836266e-01,  7.84511169e-02,  4.06638807e-02,
        -1.82790871e-01, -2.35471672e-01, -3.83850999e-02,
         1.99209178e-01,  1.99209178e-01],
       [-1.37431986e-02, -9.97534751e-02, -1.37431986e-02,
         5.47689683e-02,  2.71657249e-01, -6.08136388e-03,
        -2.38491701e-01, -2.59131083e-01, -1.37431986e-02,
         5.47689683e-02, -4.16594130e-02, -1.98245624e-02,
        -6.00140053e-03, -2.46153536e-01, -6.00140053e-03,
        -9.36721113e-02, -1.37431986e-02, -1.37431986e-02,
        -2.32410337e-01, -2.59131083e-01, -3.56580125e-02,
         2.60683905e-01,  2.85400448e-01, -9.36721113e-02,
        -6.00140053e-03, -2.59131083e-01, -9.36721113e-02,
         5.47689683e-02, -2.59131083e-01,  2.62693647e-02,
         3.43414546e-01,  4.08273828e-02, -6.08136388e-03,
        -1.83642769e-01,  2.62693647e-02, -2.59131083e-01,
        -4.49045435e-02,  5.47689683e-02, -2.06140972e-01,
        -9.36721113e-02, -9.36721113e-02],
       [-4.89010938e-02, -1.78622482e-01, -4.89010938e-02,
        -2.68496535e-01,  4.26819996e-02, -1.26130233e-01,
         9.24394643e-02, -9.47648968e-02, -4.89010938e-02,
        -2.68496535e-01,  3.38187415e-01, -1.75031327e-01,
         2.22014059e-01,  1.69668603e-01,  2.22014059e-01,
        -5.24922494e-02, -4.89010938e-02, -4.89010938e-02,
         2.18569697e-01, -9.47648968e-02,  1.16173357e-01,
        -1.36703668e-01,  9.15830934e-02, -5.24922494e-02,
         2.22014059e-01, -9.47648968e-02, -5.24922494e-02,
        -2.68496535e-01, -9.47648968e-02, -3.18180341e-03,
         2.60248700e-01,  5.78923249e-02, -1.26130233e-01,
         1.72087221e-01, -3.18180341e-03, -9.47648968e-02,
        -9.89747254e-02, -2.68496535e-01,  2.15387894e-01,
        -5.24922494e-02, -5.24922494e-02],
       [-4.52706104e-02, -6.23863705e-02, -4.52706104e-02,
         1.48955978e-01,  6.45581975e-02, -1.85212398e-01,
         8.93966250e-02, -1.09831853e-01, -4.52706104e-02,
         1.48955978e-01, -1.30347676e-01, -2.30483008e-01,
        -3.17146366e-01,  2.29338412e-01, -3.17146366e-01,
         1.22826027e-01, -4.52706104e-02, -4.52706104e-02,
         2.74609023e-01, -1.09831853e-01,  1.86798690e-01,
        -1.20750995e-01,  1.09828808e-01,  1.22826027e-01,
        -3.17146366e-01, -1.09831853e-01,  1.22826027e-01,
         1.48955978e-01, -1.09831853e-01, -3.04456781e-06,
         1.73801471e-01,  1.40130352e-02, -1.85212398e-01,
         1.06418635e-01, -3.04456781e-06, -1.09831853e-01,
        -4.53643607e-02,  1.48955978e-01,  2.74605978e-01,
         1.22826027e-01,  1.22826027e-01],
       [-1.59498157e-02,  4.82091096e-02, -1.59498157e-02,
        -1.45560739e-01,  1.33608436e-01, -1.16002323e-01,
       -5.50678915e-02, -1.15230931e-01, -1.59498157e-02,
       -1.45560739e-01, -3.60048433e-01, -1.31952139e-01,
        8.65112876e-02,  4.49846158e-02,  8.65112876e-02,
        1.64211433e-01, -1.59498157e-02, -1.59498157e-02,
        6.09344315e-02, -1.15230931e-01, -4.46559721e-01,
        1.73045232e-01,  1.49558252e-01,  1.64211433e-01,
        8.65112876e-02, -1.15230931e-01,  1.64211433e-01,
       -1.45560739e-01, -1.15230931e-01,  3.43273214e-02,
       -4.61212901e-01, -2.15119624e-02, -1.16002323e-01,
        1.88498020e-03,  3.43273214e-02, -1.15230931e-01,
        1.05161981e-01, -1.45560739e-01,  9.52617529e-02,
        1.64211433e-01,  1.64211433e-01]])
```

In [11]:

```
np.set_printoptions(formatter={"float": lambda x: ("%5.2f" %x)})
U
```

Out[11]:

```
array([[-0.09,  0.23,  0.38, -0.87,  0.13, -0.03, -0.10, -0.09, -0.02],
       [-0.58,  0.29, -0.26, -0.03,  0.10,  0.61,  0.19,  0.21,  0.22],
       [-0.23,  0.21,  0.27,  0.06, -0.31, -0.50,  0.45,  0.52,  0.09],
       [-0.42,  0.24,  0.58,  0.45,  0.08, -0.01, -0.26, -0.35, -0.17],
       [-0.37,  0.26, -0.61, -0.07,  0.11, -0.55, -0.20, -0.21, -0.17],
       [-0.26, -0.29, -0.03, -0.12, -0.62,  0.12, -0.55,  0.28, -0.21],
       [-0.29, -0.40, -0.01, -0.13, -0.39, -0.01,  0.46, -0.60,  0.12],
       [-0.35, -0.65,  0.11, -0.02,  0.53, -0.20, -0.11,  0.23,  0.24],
       [-0.09, -0.17, -0.01, -0.04,  0.19,  0.12,  0.35,  0.12, -0.88]])
```

In [12]:
```
print(U.dot(S).dot(V))
```

```
[[ 1.00  0.00  1.00  0.00  1.00 -0.00 -0.00  0.00  1.00  0.00  0.00  1.00
   0.00  1.00  0.00  0.00  1.00  1.00 -0.00  0.00  0.00  0.00  0.00  0.00
   0.00  0.00  0.00  0.00  0.00  0.00  0.00 -0.00 -0.00 -0.00  0.00  0.00
   0.00  0.00  0.00  0.00  0.00]
 [-0.00  0.00 -0.00  0.00  1.00 -0.00 -0.00 -0.00 -0.00  0.00 -0.00 -0.00
  -0.00 -0.00 -0.00  0.00 -0.00 -0.00 -0.00 -0.00 -0.00  2.00  1.00  0.00
  -0.00 -0.00  0.00  0.00 -0.00  1.00  1.00  1.00 -0.00 -0.00  1.00 -0.00
   0.00  0.00  1.00  0.00  0.00]
 [-0.00 -0.00 -0.00  0.00 -0.00 -0.00  1.00 -0.00 -0.00  0.00  0.00 -0.00
   0.00  1.00  0.00 -0.00 -0.00 -0.00  1.00 -0.00 -0.00 -0.00 -0.00 -0.00
   0.00 -0.00 -0.00  0.00 -0.00 -0.00 -0.00  1.00 -0.00  1.00 -0.00 -0.00
   0.00  0.00  1.00 -0.00 -0.00]
 [-0.00  1.00 -0.00 -0.00 -0.00  1.00  1.00 -0.00 -0.00 -0.00  0.00  1.00
   0.00 -0.00  0.00  0.00 -0.00 -0.00 -0.00 -0.00  0.00  1.00 -0.00  0.00
   0.00 -0.00  0.00 -0.00 -0.00 -0.00  0.00  2.00  1.00  0.00 -0.00 -0.00
   0.00 -0.00 -0.00  0.00  0.00]
 [ 0.00 -0.00  0.00 -0.00 -0.00 -0.00  0.00  1.00  0.00 -0.00  0.00 -0.00
  -0.00  0.00 -0.00 -0.00  0.00  0.00  0.00  1.00  0.00  1.00 -0.00 -0.00
  -0.00  1.00 -0.00 -0.00  1.00  1.00  0.00 -0.00 -0.00  0.00  1.00  1.00
  -0.00 -0.00  1.00 -0.00 -0.00]
 [-0.00 -0.00 -0.00  1.00  0.00  0.00  0.00 -0.00 -0.00  1.00 -0.00  0.00
  -0.00 -0.00 -0.00 -0.00 -0.00 -0.00  0.00 -0.00 -0.00  1.00  0.00 -0.00
  -0.00 -0.00 -0.00  1.00 -0.00  0.00  0.00  0.00  0.00  1.00  0.00 -0.00
   1.00  1.00  0.00 -0.00 -0.00]
 [-0.00 -0.00 -0.00 -0.00 -0.00  0.00  0.00  0.00 -0.00 -0.00  1.00  0.00
   1.00  0.00  1.00 -0.00 -0.00 -0.00  0.00  0.00 -0.00  1.00 -0.00 -0.00
   1.00  0.00 -0.00 -0.00  0.00  0.00 -0.00  0.00  0.00  1.00  0.00  0.00
   1.00 -0.00  0.00 -0.00 -0.00]
 [-0.00  1.00 -0.00 -0.00  0.00  0.00  0.00  0.00 -0.00 -0.00  1.00  0.00
  -0.00  0.00 -0.00  1.00 -0.00 -0.00  0.00  0.00  1.00  1.00  0.00  1.00
  -0.00  0.00  1.00 -0.00  0.00  0.00  0.00  0.00  0.00 -0.00  0.00  0.00
   1.00 -0.00  0.00  1.00  1.00]
 [-0.00  0.00  0.00  0.00 -0.00  0.00  0.00  0.00  0.00  0.00  1.00  0.00
  -0.00 -0.00 -0.00 -0.00 -0.00 -0.00 -0.00  0.00  1.00 -0.00 -0.00 -0.00
  -0.00  0.00 -0.00  0.00  0.00 -0.00  1.00  0.00  0.00 -0.00 -0.00  0.00
  -0.00  0.00 -0.00 -0.00 -0.00]]
```

verify whether U is column wise orthonormal

In [13]:
```
U.transpose().dot(U)
```

Out[13]:
```
array([[ 1.00,  0.00,  0.00,  0.00, -0.00,  0.00,  0.00,  0.00,  0.00],
       [ 0.00,  1.00, -0.00, -0.00,  0.00, -0.00, -0.00, -0.00, -0.00],
       [ 0.00, -0.00,  1.00,  0.00, -0.00, -0.00,  0.00,  0.00, -0.00],
       [ 0.00, -0.00,  0.00,  1.00, -0.00, -0.00,  0.00, -0.00, -0.00],
       [-0.00,  0.00, -0.00, -0.00,  1.00, -0.00,  0.00, -0.00,  0.00],
       [ 0.00, -0.00, -0.00, -0.00, -0.00,  1.00, -0.00,  0.00, -0.00],
       [ 0.00, -0.00,  0.00,  0.00,  0.00, -0.00,  1.00,  0.00,  0.00],
       [ 0.00, -0.00,  0.00, -0.00, -0.00,  0.00,  0.00,  1.00,  0.00],
       [ 0.00, -0.00, -0.00, -0.00,  0.00, -0.00,  0.00,  0.00,  1.00]])
```

verify whether V is row wise orthonormal

```
In [14]: V.dot(V.transpose())
```

```
Out[14]: array([[ 1.00, -0.00, -0.00, -0.00, -0.00, -0.00,  0.00, -0.00,  0.00],
                [-0.00,  1.00, -0.00,  0.00, -0.00,  0.00,  0.00,  0.00,  0.00],
                [-0.00, -0.00,  1.00,  0.00,  0.00,  0.00,  0.00, -0.00, -0.00],
                [-0.00,  0.00,  0.00,  1.00,  0.00, -0.00,  0.00, -0.00, -0.00],
                [-0.00, -0.00,  0.00,  0.00,  1.00,  0.00,  0.00,  0.00, -0.00],
                [-0.00,  0.00,  0.00, -0.00,  0.00,  1.00, -0.00,  0.00,  0.00],
                [ 0.00,  0.00,  0.00,  0.00,  0.00, -0.00,  1.00, -0.00,  0.00],
                [-0.00,  0.00, -0.00, -0.00,  0.00,  0.00, -0.00,  1.00, -0.00],
                [ 0.00,  0.00, -0.00, -0.00, -0.00,  0.00,  0.00, -0.00,  1.00]])
```

calculate a rank 4 approximation of given matrix

In [15]:

```
k = 4
UK = U[:,:k]
SK = S[:k,:k]
VK = V[:k,:]
#Ak = UK.dot(SK).dot(VK)
print(UK)
print(SK)
print(VK)
```

```
[[-0.09  0.23  0.38 -0.87]
 [-0.58  0.29 -0.26 -0.03]
 [-0.23  0.21  0.27  0.06]
 [-0.42  0.24  0.58  0.45]
 [-0.37  0.26 -0.61 -0.07]
 [-0.26 -0.29 -0.03 -0.12]
 [-0.29 -0.40 -0.01 -0.13]
 [-0.35 -0.65  0.11 -0.02]
 [-0.09 -0.17 -0.01 -0.04]]
[[ 4.61  0.00  0.00  0.00]
 [ 0.00  3.35  0.00  0.00]
 [ 0.00  0.00  3.09  0.00]
 [ 0.00  0.00  0.00  2.73]]
[[-0.02 -0.17 -0.02 -0.06 -0.15 -0.09 -0.14 -0.08 -0.02 -0.06 -0.16 -0.11
  -0.06 -0.07 -0.06 -0.08 -0.02 -0.02 -0.05 -0.08 -0.10 -0.62 -0.13 -0.08
  -0.06 -0.08 -0.08 -0.06 -0.08 -0.21 -0.15 -0.36 -0.09 -0.17 -0.21 -0.08
  -0.20 -0.06 -0.26 -0.08 -0.08]
 [ 0.07 -0.12  0.07 -0.09  0.16  0.07  0.14  0.08  0.07 -0.09 -0.36  0.14
  -0.12  0.13 -0.12 -0.19  0.07  0.07  0.06  0.08 -0.25 -0.08  0.09 -0.19
  -0.12  0.08 -0.19 -0.09  0.08  0.16  0.04  0.30  0.07 -0.14  0.16  0.08
  -0.40 -0.09  0.23 -0.19 -0.19]
 [ 0.12  0.22  0.12 -0.01  0.04  0.19  0.28 -0.20  0.12 -0.01  0.03  0.31
  -0.00  0.21 -0.00  0.04  0.12  0.12  0.09 -0.20  0.03 -0.15 -0.08  0.04
  -0.00 -0.20  0.04 -0.01 -0.20 -0.28 -0.08  0.38  0.19  0.08 -0.28 -0.20
   0.02 -0.01 -0.19  0.04  0.04]
 [-0.32  0.16 -0.32 -0.04 -0.33  0.16  0.19 -0.03 -0.32 -0.04 -0.07 -0.16
  -0.05 -0.30 -0.05 -0.01 -0.32 -0.32  0.02 -0.03 -0.02  0.02 -0.01 -0.01
  -0.05 -0.03 -0.01 -0.04 -0.03 -0.04 -0.03  0.34  0.16 -0.07 -0.04 -0.03
  -0.10 -0.04 -0.01 -0.01 -0.01]]
```

```
In [16]: Ak = UK.dot(SK).dot(VK)
         print(Ak)
```

```
[[ 0.96 -0.13  0.96  0.05  1.01 -0.07  0.05 -0.07  0.96  0.05 -0.01  0.89
   0.05  1.08  0.05 -0.06  0.96  0.96  0.12 -0.07 -0.06 -0.01  0.05 -0.06
   0.05 -0.07 -0.06  0.05 -0.07 -0.03  0.05  0.02 -0.07  0.22 -0.03 -0.07
   0.04  0.05  0.09 -0.06 -0.06]
 [ 0.05  0.15  0.05  0.08  0.54  0.16  0.28  0.44  0.05  0.08  0.05  0.20
   0.06  0.17  0.06 -0.01  0.05  0.05  0.12  0.44 -0.01  1.72  0.49 -0.01
   0.06  0.44 -0.01  0.08  0.44  0.94  0.49  0.93  0.16  0.26  0.94  0.44
   0.13  0.08  1.06 -0.01 -0.01]
 [ 0.12  0.30  0.12 -0.02  0.24  0.33  0.51 -0.03  0.12 -0.02 -0.08  0.45
  -0.03  0.29 -0.03 -0.03  0.12  0.12  0.17 -0.03 -0.05  0.47  0.12 -0.03
  -0.03 -0.03 -0.03 -0.02 -0.03  0.09  0.10  0.97  0.33  0.13  0.09 -0.03
  -0.08 -0.02  0.27 -0.03 -0.03]
 [-0.07  0.82 -0.07 -0.03  0.09  0.77  1.11 -0.17 -0.07 -0.03 -0.02  0.70
  -0.04  0.26 -0.04  0.05 -0.07 -0.07  0.33 -0.17  0.02  0.90  0.16  0.05
  -0.04 -0.17  0.05 -0.03 -0.17 -0.01  0.13  2.04  0.77  0.27 -0.01 -0.17
  -0.02 -0.03  0.32  0.05  0.05]
 [-0.07 -0.27 -0.07  0.05  0.37 -0.17 -0.20  0.57 -0.07  0.05 -0.09 -0.24
   0.02 -0.10  0.02 -0.10 -0.07 -0.07 -0.03  0.57 -0.11  1.26  0.44 -0.10
   0.02  0.57 -0.10  0.05  0.57  1.02  0.44  0.08 -0.17  0.04  1.02  0.57
  -0.03  0.05  0.98 -0.10 -0.10]
 [ 0.05  0.25  0.05  0.17  0.13 -0.03 -0.04  0.05  0.05  0.17  0.56  0.02
   0.21  0.03  0.21  0.28  0.05  0.05 -0.02  0.05  0.36  0.84  0.08  0.28
   0.21  0.05  0.28  0.17  0.05  0.13  0.16  0.01 -0.03  0.36  0.13  0.05
   0.65  0.17  0.11  0.28  0.28]
 [ 0.05  0.32  0.05  0.21  0.11 -0.04 -0.07  0.02  0.05  0.21  0.72  0.01
   0.26  0.02  0.26  0.36  0.05  0.05 -0.03  0.02  0.46  0.93  0.06  0.36
   0.26  0.02  0.36  0.21  0.02  0.08  0.16 -0.05 -0.04  0.44  0.08  0.02
   0.83  0.21  0.05  0.36  0.36]
 [-0.06  0.61 -0.06  0.28 -0.07  0.05  0.02 -0.10 -0.06  0.28  1.06 -0.01
   0.36 -0.09  0.36  0.56 -0.06 -0.06 -0.03 -0.10  0.70  1.13 -0.01  0.56
   0.36 -0.10  0.56  0.28 -0.10 -0.11  0.13  0.05  0.05  0.61 -0.11 -0.10
   1.20  0.28 -0.14  0.56  0.56]
 [ 0.00  0.11  0.00  0.08  0.01 -0.03 -0.05 -0.00  0.00  0.08  0.28 -0.02
   0.10 -0.02  0.10  0.14  0.00  0.00 -0.02 -0.00  0.18  0.29  0.00  0.14
   0.10 -0.00  0.14  0.08 -0.00 -0.00  0.04 -0.07 -0.03  0.16 -0.00 -0.00
   0.32  0.08 -0.02  0.14  0.14]]
```

THANK YOU