```
import scipy.io
import numpy as np
import copy
freq_matrix = scipy.io.loadmat('wordVecV.mat')['V']


print(freq_matrix.shape)
M = np.zeros((freq_matrix.shape[0], freq_matrix.shape[1]))
for i in range(freq_matrix.shape[0]):
  for j in range(freq_matrix.shape[1]):
    if freq_matrix[i][j] > 0:
      M[i][j] = 1

     (1651, 10)


for doc_idx in range(freq_matrix.shape[1]):
  col = M[:, doc_idx]
  length = np.linalg.norm(col)
  for word_idx in range(freq_matrix.shape[0]):
    M[word_idx][doc_idx] = M[word_idx][doc_idx]/length


U, S, VH = np.linalg.svd(M)
print(S[:10])

 [1.53662942 1.01924241 0.95868454 0.95391295 0.9413064  0.9289078
      0.8977405  0.89188192 0.86866454 0.81608339]


K = 9
d_all = []
for i in range(freq_matrix.shape[1]):
  basis_vector = np.zeros(freq_matrix.shape[1])
  basis_vector[i] = 1
  d = np.zeros(freq_matrix.shape[0])
  for j in range(K):
    #print((VH[j,:].reshape(1, 10)@basis_vector).shape)
    d = d + S[j] * VH[j,:].reshape(1, freq_matrix.shape[1])@basis_vector * U[:, j]
  d_all.append(d.reshape(freq_matrix.shape[0], 1))


smallest_distance = np.inf
smallest_angle = np.inf
smallest_distance_set = (0, 0)
smallest_angle_set = (0, 0)
for i in range(0, freq_matrix.shape[1]):
  for j in range(i+1, freq_matrix.shape[1]):
    distance = np.linalg.norm(d_all[i] - d_all[j])
    if distance < smallest_distance:
      smallest_distance = distance
      smallest_distance_set = (i+1, j+1) # add one index
```

```
    angle = np.arccos(d_all[i].T @ d_all[j] / \
                      (np.linalg.norm(d_all[i]) * np.linalg.norm(d_all[j])))
    if angle <smallest_angle:
      smallest_angle = angle
      smallest_angle_set = (i+1, j+1) # add one index
print("smallest distance pair", smallest_distance_set)
print("smallest angle pair", smallest_angle_set)
```

```
smallest distance pair (9, 10)
smallest angle pair (9, 10)
```

```
K_list = [8, 7, 6, 5, 4, 3, 2, 1]
for K in K_list:
  print("K value:", K)
  d_all = []
  for i in range(freq_matrix.shape[1]):
    basis_vector = np.zeros(freq_matrix.shape[1])
    basis_vector[i] = 1
    d = np.zeros(freq_matrix.shape[0])
    for j in range(K):
      #print((VH[j,:].reshape(1, 10)@basis_vector).shape)
      d = d + S[j] * VH[j,:].reshape(1, freq_matrix.shape[1])@basis_vector * U[:, j]
    d_all.append(d.reshape(freq_matrix.shape[0], 1))

  smallest_distance = np.inf
  smallest_angle = np.inf
  smallest_distance_set = (0, 0)
  smallest_angle_set = (0, 0)
  for i in range(0, freq_matrix.shape[1]):
    for j in range(i+1, freq_matrix.shape[1]):
      distance = np.linalg.norm(d_all[i] - d_all[j])
      if distance < smallest_distance:
        smallest_distance = distance
        smallest_distance_set = (i+1, j+1) # add one index
      angle = np.arccos(d_all[i].T @ d_all[j] / \
                        (np.linalg.norm(d_all[i]) * np.linalg.norm(d_all[j])))
      if angle <smallest_angle:
        smallest_angle = angle
        smallest_angle_set = (i+1, j+1) # add one index
  print("smallest distance pair", smallest_distance_set)
  print("smallest angle pair", smallest_angle_set)
```

```
K value: 8
smallest distance pair (9, 10)
smallest angle pair (9, 10)
K value: 7
smallest distance pair (9, 10)
smallest angle pair (9, 10)
K value: 6
smallest distance pair (9, 10)
smallest angle pair (9, 10)
```

```
K value: 5
smallest distance pair (9, 10)
smallest angle pair (9, 10)
K value: 4
smallest distance pair (9, 10)
smallest angle pair (9, 10)
K value: 3
smallest distance pair (9, 10)
smallest angle pair (9, 10)
K value: 2
smallest distance pair (1, 6)
smallest angle pair (1, 6)
K value: 1
smallest distance pair (2, 8)
smallest angle pair (1, 5)
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:24: RuntimeWarning:
```