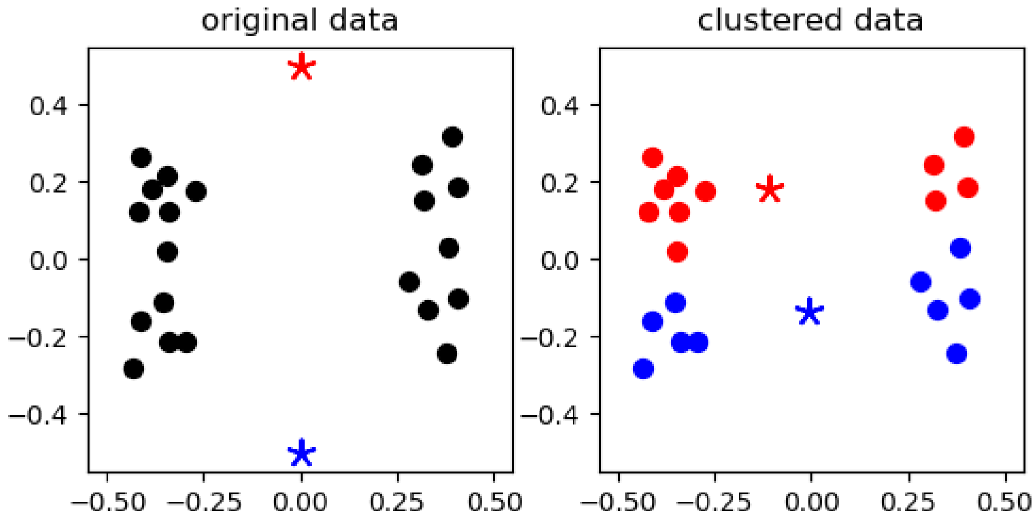
Exercise 9.1

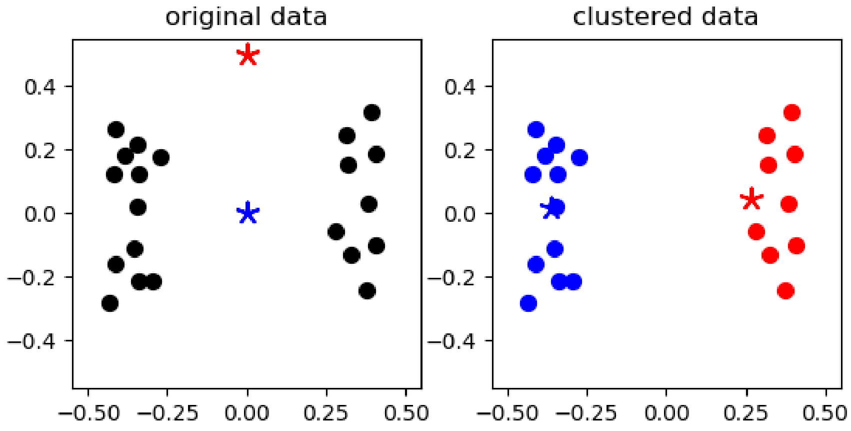
a) Code

```python
def your_K_means(X, K, C0):
    C = C0
    for i in np.arange(100):
        k_star_arr = []
        W = np.zeros([K,np.shape(X)[1]])
        for p in np.arange(np.shape(X)[1]):
            for k in np.arange(K):
                k_s = (np.linalg.norm(C[:,k].reshape(2,1)-X[:,p].reshape(2,1),ord=2))**2
                k_star_arr.append(k_s)
            k_star = np.argmin(k_star_arr)
            k_star_arr = []
            if k_star == 0:
                W[:,p] = np.array([[1,0]])
            else:
                W[:,p] = np.array([[0,1]])
        Xp_sum = X.dot(W.T)
        Sk_inv = np.linalg.inv(W.dot(W.T))
        C = Sk_inv.dot(Xp_sum)

    return C, W
```

b) C = [[0,0],[-0.5,0.5]]



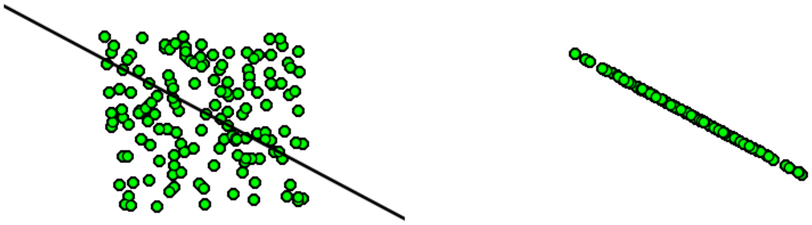
c) C = [[0,0],[0,0.5]]



Exercise 9.2

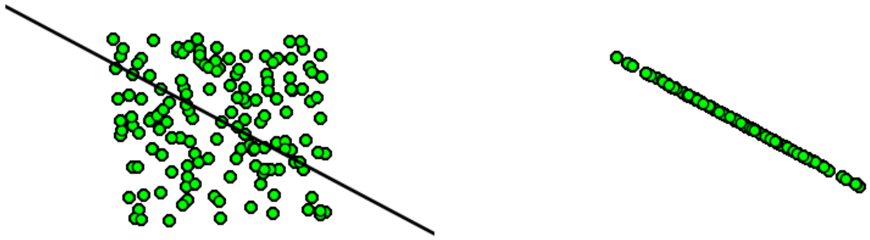
a) Code and Figure

```python
def your_PCA(X, K):
    covmat = X.dot(X.T)
    U,S,V = np.linalg.svd(covmat)
    U_k = U[:,0]
    S_kk = S[0].reshape(1,1)
    V_k = V[:,0]
    C = U_k.dot(S_kk)
    W = V_k.T

    return C, W
```



b) Code and Figure

```python
def your_PCA(X, K):
    W0 = np.random.randint(1,100,(1,150))
    W = W0
    for i in np.arange(100):
        C = X.dot(W.T).dot(np.linalg.pinv(W.dot(W.T)))
        W = np.linalg.pinv((C.T).dot(C)).dot(C.T).dot(X)

    return C, W
```

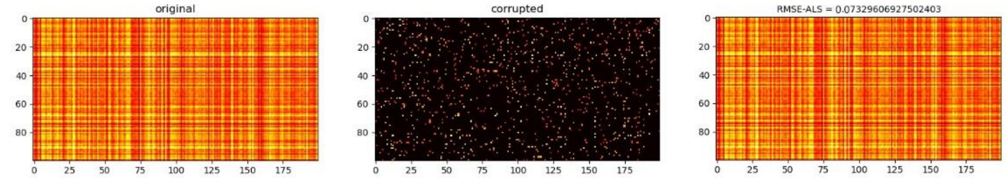


Exercise 9.4

Code:

```python
def matrix_complete(X, K):
    C0 = np.random.randint(100,200,(100,5))
    C = C0
    W_arr = np.zeros((5,200))
    C_arr = np.zeros((100,5))
    for i in np.arange(60):
        for p in np.arange(np.shape(X)[1]):
            wp = ((C.T).dot(X[:,p].reshape(100,1)))/(((C).dot(C.T)).sum())
            W_arr[:,p] = np.squeeze(wp)
            if X[:,p].sum() == 0:
                wp = np.array([0,0,0,0,0])
                W_arr[:,p] = wp
        W = W_arr
        for n in np.arange(np.shape(X)[0]):
            cn = (X[n,:].dot(W.T)).dot(np.linalg.pinv(W.dot(W.T)))
            C_arr[n,:] = np.squeeze(cn)
            if X[n,:].sum() == 0:
                cn = np.array([0,0,0,0,0])
                C_arr[n,:] = cn
        C = C_arr
        W_arr = np.zeros((5,200))
        C_arr = np.zeros((100,5))

    return C, W
```

Figure:


original

corrupted

RMSE-ALS = 0.07329606927502403