```python
import numpy as np

def load_sample_data_pca():

    np.random.seed(3)
    eps=0.5
    n=30
    x=np.random.uniform(-1,1,n)
    y=x+eps*np.random.uniform(-1,1,n)
    x=x-np.mean(x)
    y=y-np.mean(y)
    data=np.vstack((x,y)).transpose()

    return data


def load_multidimensional_data_pca(n_data, n_vec, dim, eps ):

    points=[]
    vectors=np.random.uniform(-1,1,(dim,n_vec))
    for idata in range(n_data):
        alphas=np.random.normal(size=n_vec)
        points.append(np.sum(np.dot(vectors,np.diag(alphas)),axis=1))

    points=np.array(points)
    pert=eps*np.random.normal(size=points.shape)

    return points+pert

def load_ex1_data_pca(eps=0.1):

    np.random.seed(1231)
    n=30
    x=np.random.uniform(-1,1,n)

    y=2*x*x

    epsx=eps*np.random.uniform(-1,1,n)
    epsy=eps*np.random.uniform(-1,1,n)

    x=x+epsx
    y=y+epsy

    x=x-np.mean(x)
    y=y-np.mean(y)

    data=np.vstack((x,y)).transpose()
    return data

def load_ex2_data_pca(dim=10 , eps=0.0 , seed=8, fat=True, eps1=0.05, n_add=30):

    group = np.array([[0.067, 0.21], [0.092, 0.21],
  [0.294, 0.445], [0.227, 0.521], [0.185, 0.597],
  [0.185, 0.689], [0.235, 0.748], [0.319, 0.773],
  [0.387, 0.739], [0.437, 0.672], [0.496, 0.739],
  [0.571, 0.773], [0.639, 0.765], [0.765, 0.924],
  [0.807, 0.933], [0.849, 0.941], [0.118, 0.143], [0.118, 0.176],
  [0.345, 0.378], [0.395, 0.319], [0.437, 0.261],
  [0.496, 0.328], [0.546, 0.395], [0.605, 0.462],
  [0.655, 0.529], [0.697, 0.597], [0.706, 0.664],
  [0.681, 0.723], [0.849, 0.798], [0.857, 0.849],
  [0.866, 0.899]])

    points=[]
    np.random.seed(seed)
    n_data=group.shape[0]
```

```python
    vectors=np.random.uniform(-1,1,(dim,2))
    vectors[:,0]=vectors[:,0]/np.linalg.norm(vectors[:,0])
    vectors[:,1]=vectors[:,1]-np.dot(vectors[:,1],vectors[:,0])*vectors[:,0]
    vectors[:,1]=vectors[:,1]/np.linalg.norm(vectors[:,1])

    for idata in range(n_data):
        points.append(np.sum(np.dot(vectors,np.diag(group[idata,:])),axis=1))

    points=np.array(points)
    pert=eps*np.random.normal(size=points.shape)

    data=points+pert

    data=data-np.mean(data,axis=0)
    if (fat):
        data_added={}
        for iadd in range(n_add):
            data_added[iadd]=np.zeros((n_data,dim))
            for idata in range(n_data):
                noise=np.random.uniform(-eps1,eps1,dim)
                data_added[iadd][idata,:]=data[idata,:]+noise[:]
        for iadd in range(n_add):
            data=np.concatenate([data,data_added[iadd]],axis=0)

    return data

def load_ex1_data_clust(dim=5, n_clusters=6, eps=12.0, dist=20, seed=13124,
n_points=20, return_centers=False):

    np.random.seed(seed)
    centers=np.random.uniform(-dist,dist,(dim,n_clusters))
    cov=np.identity(dim)*eps

    data={}
    for iclust in range(n_clusters):
        data[iclust] = np.random.multivariate_normal(centers[:,iclust], cov,
n_points)

    if (return_centers):
        return centers,np.concatenate([data[iclust] for iclust in data.keys
()],axis=0)
    else:
        return np.concatenate([data[iclust] for iclust in data.keys()],axis=0)

def km_load_th1():
    return load_ex1_data_clust(dim=2, n_clusters=3, eps=12.0, dist=20,
seed=13124, n_points=40, return_centers=False)


def gm_load_th1():
    np.random.seed(1321)
    points1=np.random.multivariate_normal([0,0], [[0.01,0.0],[0.0,1.0]], 1000)
    points2=np.random.multivariate_normal([0,4], [[0.01,0.0],[0.0,1.0]], 1000)
    points3=np.random.multivariate_normal([1,2], [[0.01,0.0],[0.0,1.0]], 1000)

    points=np.concatenate([points1,points2, points3], axis=0)
    return points


def gm_load_th2():
    np.random.seed(14321)
    points1=np.random.multivariate_normal([0,0.0], [[0.01,0.0],[0.0,1.0]], 1000)
    points2=np.random.multivariate_normal([0,0.0], [[1.5,0.0],[0.0,1.0]], 1000)
    points=np.concatenate([points1,points2], axis=0)
    return points
```