```
M = csvread('HW10_dat.csv',1);
Color = csvread('HW10_color.csv');
X = M';

% Original data
figure;
scatter3(X(1,:),X(2,:),X(3,:),36,Color);
%
% % PCA
n = size(M,1);
mean = sum(M,1)/n;
PC = pca(M);
for i = 1:n
    proj(i,1) = (M(i,:)-mean)*PC(:,1);
    proj(i,2) = (M(i,:)-mean)*PC(:,2);
end
figure;
scatter(proj(:,1),proj(:,2),36,Color);
%
% % Gaussian KPCA
KK = KappaMatrix(X,'GaussianKernel',1.3);
[Sigma2,V2,Y2]=KernelPCA(KK,2);
figure;
scatter(Y2(1,:),Y2(2,:),36,Color);

k = 12;
scaling = sqrt(n);
Y = LocallyLinearEmbedding(X,k,scaling);
size(Y);
figure;
scatter(Y(:,1),Y(:,2),36,Color);


function D = DistanceMatrix(X)
    n = size(X,2);
    p = size(X,1);
    D = zeros(n,n);
    for i = 1:n
        D(i,i) = 0;
        xi = X(:,i);
        for j = i+1:n
            xj = X(:,j);
            D(i,j) = norm(xi-xj);
            D(j,i) = D(i,j);
        end
    end
end

function index = kNearestNeighbours(D,k)
    n = size(D,1);
    ind = zeros(k+1);
```

```
    index = zeros(n,k);
    for i = 1:n
        [~,ind] = mink(D(i,:),k+1);
        index(i,:) = ind(2:k+1);
    end
end

function W = weight(X,index)
    n = size(X,2);
    k = size(index,2);
    C = zeros(k,k);
    kones = ones(k,1);
    Wt = zeros(k,n);
    for i = 1:n
        xi = X(:,i);
        for p = 1:k
            xp = X(:,index(i,p));
            for q = p:k
                xq = X(:,index(i,q));
                C(p,q) = (xi-xp)'*(xi-xq);
                C(q,p) = C(p,q);
            end
        end
        s = svd(C);
        C = C + trace(C)*eye(k)/1000;
        temp = C\kones;
        Wt(:,i) = temp/(kones'*temp);
    end
    W = zeros(n,n);
    for i = 1:n
        for j = 1:k
            W(i,index(i,j)) = Wt(j,i);
        end
    end
end

function Y = ConstructY(W,k,scaling)
    n = size(W,1);
    I = eye(n);
    M = (I-W)'*(I-W);
    [U,S,~] = svd(M);
    for i = n:-1:1
        if(S(i,i) > S(n,n))
            target = i;
            break;
        end
    end
    Y = zeros(n,k);
    for i = 1:k
        Y(:,i) = U(:,target)*scaling;
        target = target-1;
```

```
    end
end

function Y = LocallyLinearEmbedding(X,k,scaling)
    D = DistanceMatrix(X);
    index = kNearestNeighbours(D,k);
    W = weight(X,index);
    Y = ConstructY(W,k,scaling);
end

% Taken from HW5.m

function G = GaussianKernel(x,y,sigma)
    G = exp(-((norm(x-y)/sigma)^2)/2);
end

function K = KappaMatrix(X,type,tuning_parameter)
  p = size(X,1);
  n = size(X,2);
  H = eye(n)-ones(n,1)*ones(1,n)/n;
  if type == "Polynomial"
      X = X*H;
  end
  K = zeros(n,n);
  for i=1:n
      xx = X(:,i);
      for j=i:n
         yy = X(:,j);
         if type == "GaussianKernel"
           K(i,j)=GaussianKernel(xx,yy,tuning_parameter);
         elseif type == "Polynomial"
           K(i,j)=PolynomialKernel(xx,yy,tuning_parameter);
         end
         K(j,i)=K(i,j);
      end
  end
  K = H*K*H;
end

function [Sigma,V,Y] = KernelPCA(kappa,d)
    [U,S,V] = svd(kappa);
    V = V(:,1:d);
    Sigma = sqrtm(S(1:d,1:d));
    Y = Sigma*V.';
end
```