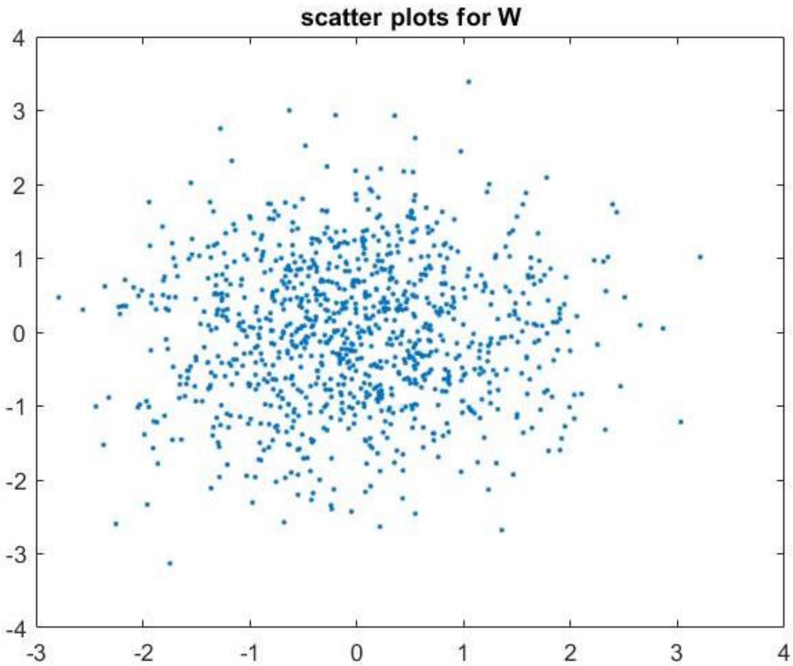
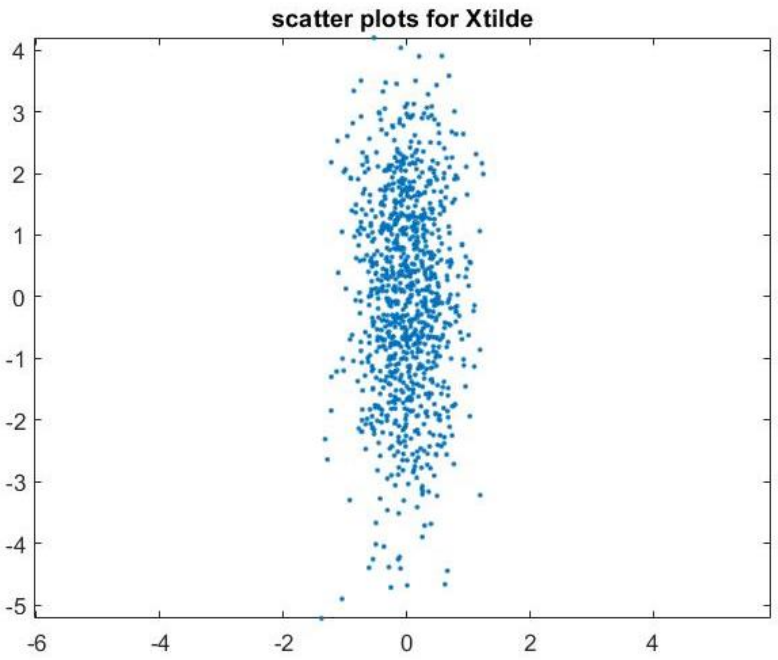
HW 5 Report

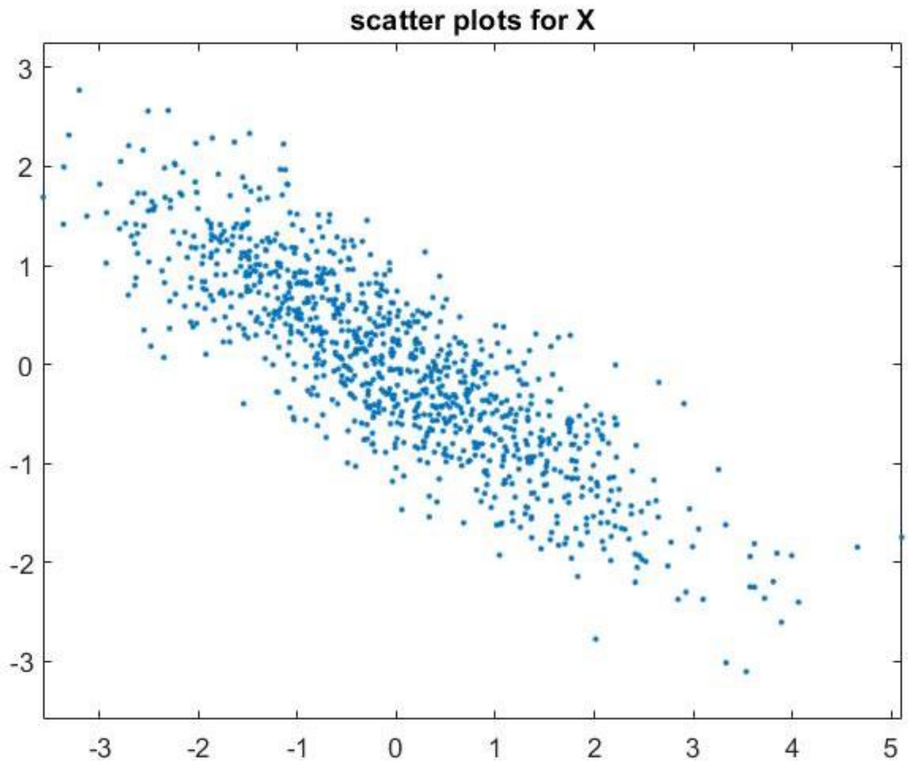
Ruijie Song

March.5.2021

2.1 Exercise: Generating Gaussian random vectors







2.2 Exercise: Covariance Estimation and Whitening

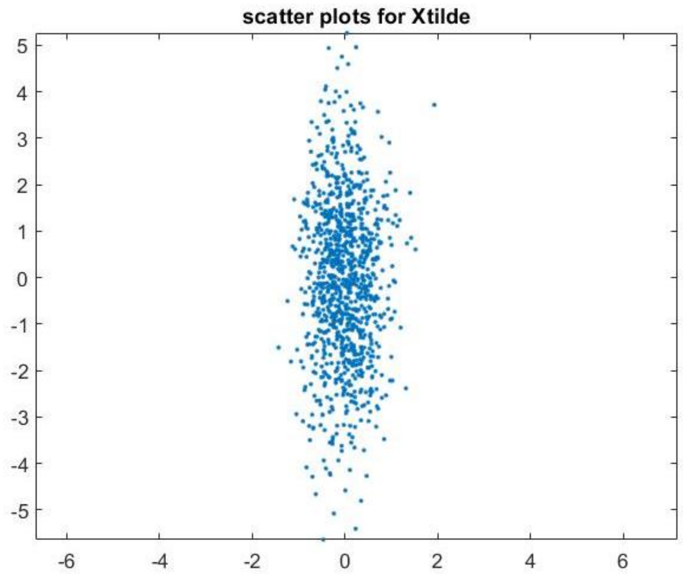
1. the theoretical value of the covariance matrix Rx:

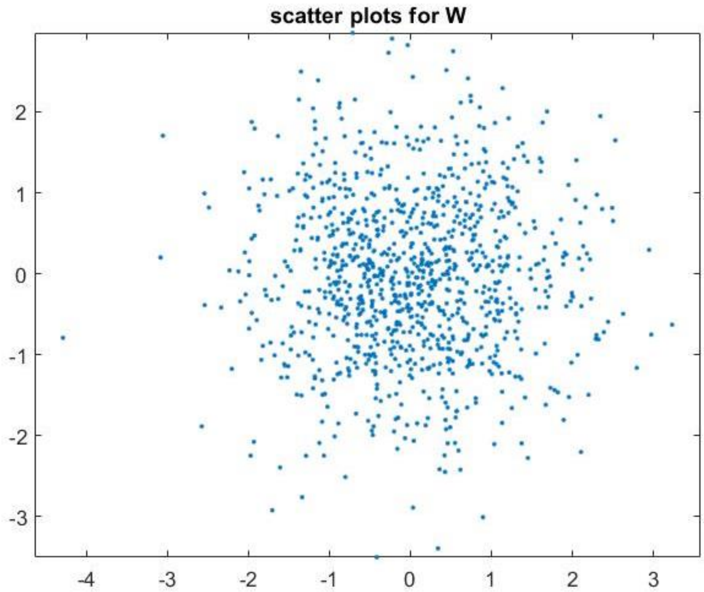
2	-1.2
-1.2	1

2. numerical listing of your covariance estimate Rx:

1.80186189486342	-1.05439805350776
-1.05439805350776	0.903742306202859

3.



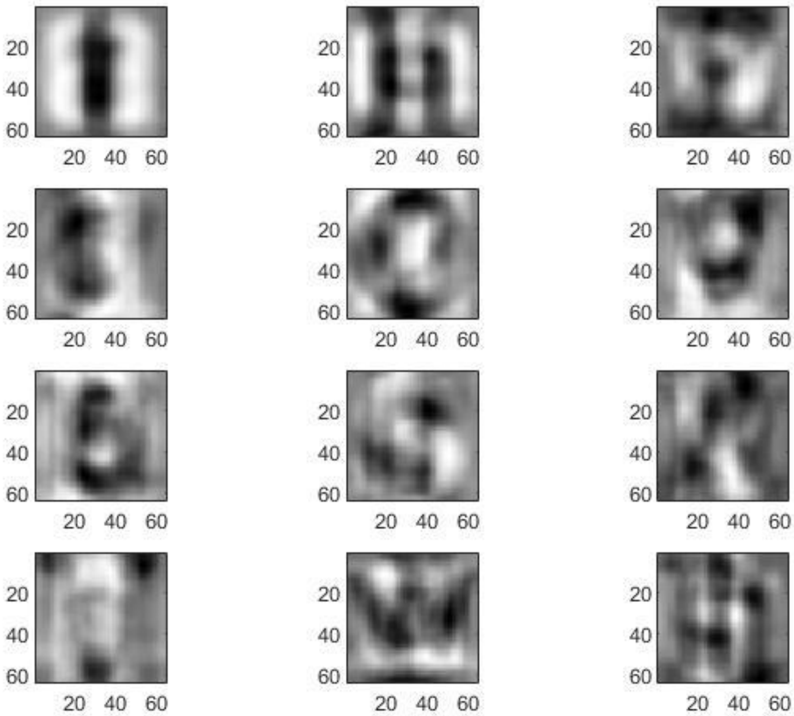


4. numerical listing of the covariance estimate Rw:

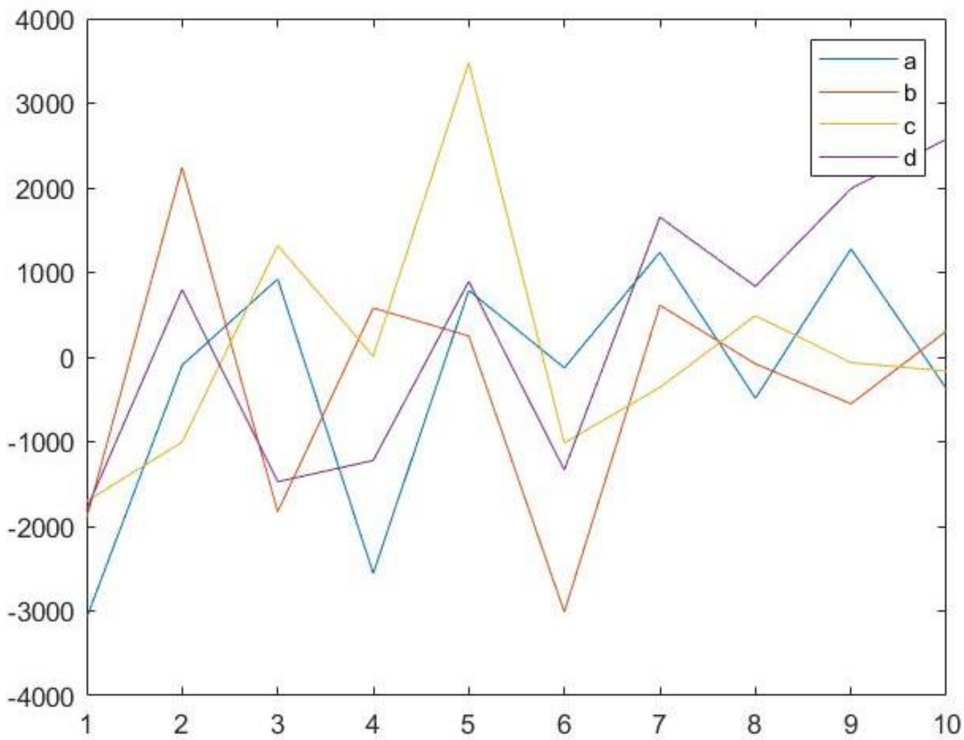
1	0
0	1

4. Eigenimages, PCA, and Data Reduction

1.



2.



3.

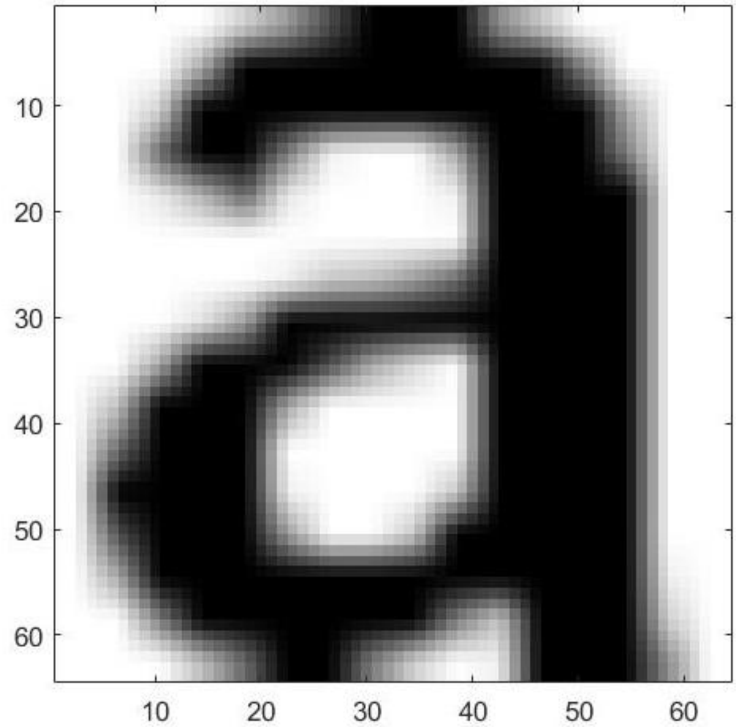
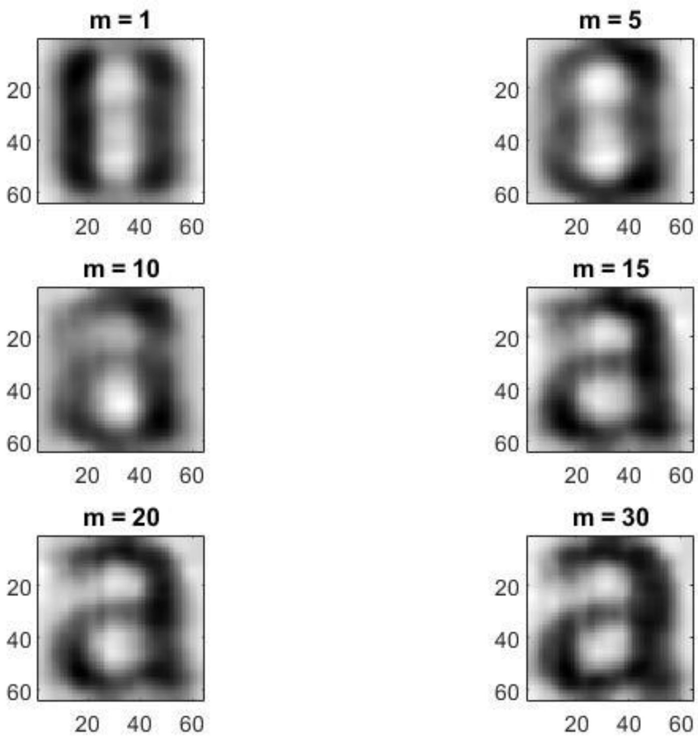


Figure 1. original image



5.1 Exercise: Classification and PCA

Input	Output
d	a
j	y
l	i
n	v
p	e
q	a
u	a
y	x

1. Let Bk = Λk, i.e. assume each class has a different diagonal covariance, where the elements of Λk are the diagonal elements of Rk.

i	l
y	v

2. Let Bk = Rwc, i.e. assume each class has the same covariance, where Rwc is defined as the average within-class covariance

g	q
y	v

3. Let Bk = Λ, i.e. each class has the same diagonal covariance, where the elements of Λ are the diagonal elements of the matrix, Rwc, defined above.

f	t
y	v

4. Let Bk = I, i.e. each class has an identity covariance around a different mean, μk.

q	g
f	t
y	v

1. The 2,3 & 4 work the best
2. The accuracy of the estimates is more important. The accuracy of the data model may not impact the estimates. Even the covariance is I, the accuracy of estimates is high.

```
clear all
%{
%% 2.1
% Plot of W
n = 1000;
p = 2;
mu = 0;
sigma = 1;
W = normrnd(mu,sigma,p,n);
%{
figure()
plot(W(1,:),W(2,:),'.')
title('scatter plots for W')
%}
% Plot of W
Rx = [2,-1.2;-1.2,1];
[E,gamma] = eig(Rx);
Xtilde = gamma^(0.5) * W;
%{
figure()
plot(Xtilde(1,:),Xtilde(2,:),'.')
axis('equal')
title('scatter plots for Xtilde')
%}
X = E*Xtilde;
%{
figure()
plot(X(1,:),X(2,:),'.')
axis('equal')
title('scatter plots for X')
%}
%% 2.2
miu_head = mean(X,2);
Z = X - miu_head;
R_head = 1/(n-1) * (Z*Z.');

[E,gamma] = eig(R_head);
Xtilde = E.' * X;
figure()
plot(Xtilde(1,:),Xtilde(2,:),'.')
axis('equal')
title('scatter plots for Xtilde')

W = gamma^(-0.5) * E.' * X;
figure()
plot(W(1,:),W(2,:),'.')
axis('equal')
title('scatter plots for W')
```

```
miu_W = mean(W,2);
Z_W = W - miu_W;
R_W = 1/(n-1) * (Z_W*Z_W.');
%}
%% 4.1
run('./training_data/read_data.m')

miu_head = mean(X,2);
Z = X - miu_head;
[U S V] = svd(Z,0);
%{
for i=1:12
  img=reshape(U(:,(i)),64,64);
  figure(1); subplot(4,3,i); imagesc(img);
  axis('image'); colormap(gray(256));
end
%}
Y = U(:,1:10).' * Z(:,1:4);
%{
figure()
for i = 1:4
    plot(1:10,Y(:,i))
    hold on
end
legend('a','b','c','d')
%}
%{
figure()
img=reshape(X(:,1),64,64);
imagesc(img);
axis('image'); colormap(gray(256));
%}
%{
m = [1,5,10,15,20,30];
figure()
for i=1:length(m)
    subplot(3,2,i)
    Xre = U(:,1:m(i)) * U(:,1:m(i)).' * Z(:,1);
    Xre = Xre + miu_head;
    img=reshape(Xre,64,64);
    imagesc(img);
    axis('image');
    title(['m = ',num2str(m(i))])
    colormap(gray(256));
end
%}
```

```
%% 5.1
empty_cell=cell(26,2);
params=cell2struct(empty_cell,{'M','R'},2);
% trans. to a lower dimension
A = U(:,1:10);
Y = A.' * Z;
% store the mean and covariance
Rwc = zeros(10,10);
for k = 1:26
    params(k).M = mean(Y(:,k:26:end),2);
    %params(k).R = (Y(:,k:26:end)-params(k).M)*(Y(:,k:26:end)-params(k).M).' / (12-1);
    params(k).R = eye(10);
    % params(k).R =
    % diag(diag((Y(:,k:26:end)-params(k).M)*(Y(:,k:26:end)-params(k).M).' /
    % (12-1))); % Sigma k
    %{
    % Bk = Rwc
    for i = 1:10
        for j = 1:10
            Rwc(i,j) = Rwc(i,j) + params(k).R(i,j);
        end
    end
    %}
end
%{
for k = 1:26
    params(k).R = Rwc / 26;
    % params(k).R = diag(diag(params(k).R)); % find the diag. of Rwc
end
%}
% transfer the test data into matrix
[rowX,colX] = size(X);
test_data = zeros(rowX,length(datachar));
i = 1;
for ch = datachar
    im_name = sprintf('./test_data/veranda/%s.tif',ch);
    test_data(:,i) = reshape(imread(im_name),rowX,1);
    i = i + 1;
end
% reduce the dimension of test data
y = A.' * (test_data-miu_head);
% test the classifier
k_star = zeros(26,length(datachar));
kmin_ind = zeros(1,26);
for i = 1:length(datachar) % traverse through the Input
    yi = y(:,i);
    for j = 1:26 % traverse through the data in the STRUCTURE
        k_star(j,i) = (yi-params(j).M).' * (params(j).R)^-1 * (yi-params(j).M) + log(det(params(j)↙
```

```
R));
    end
    [kmin,ind] = min(k_star(:,i));
    kmin_ind(i) = ind;
end
% show the output of the classifier
for i = 1:26
    output(i) = datachar(kmin_ind(i));
end
```