Principal Components Analysis (PCA)

Examples
Example: Recognition of Handwritten Digits

- Very important commercially (e.g., postal service)
- The best classification algorithms use multiple techniques; achieve accuracy of > 99%

- MNIST database
  - Training set of 60,000 examples, and a test set of 10,000 examples
  - Images are size normalized to fit in a 20x20 pixel box, then centered in a 28x28 image

http://yann.lecun.com/exdb/mnist/
Classification using PCA

- The unknown input image is projected onto the top principal components
- Its representation is compared to all the images in the database
- The closest match is taken to be the identified digit
- For more robustness, the closest “k” matches can be found
  - The identify of the majority of these matches is taken to be the identified digit
  - This is known as the k-nearest-neighbor method

Comparison of images in “eigenspace” (from Ryan Crawford, EENG 510 final project report, 2011)
Examples of poorly written digits

• The problem can be difficult in some cases
• Can you tell what these are?

\[ \begin{array}{cccc}
\mathbf{0} & \mathbf{2} & \mathbf{4} & \mathbf{5} \\
\end{array} \]

Answers (from left to right): 0, 2, 4, 5, 7
Example – PCA to compress color images

• Recall that a color image is made up of pixels with red, green, blue (RGB) values.
• Essentially, we have a collection of 3D vectors in RGB space.
clear all
close all

RGB = im2double(imread('peppers.png'));

% Convert 3-dimensional array array to 2D, where each row is a pixel (RGB)
X = reshape(RGB, [], 3);
N = size(X,1); % N is the number of pixels

% Plot pixels in color space. To limit the number of points, only plot
% every 100th point.
figure
hold on
for i=1:100:size(X,1)
    mycolor = X(i,:);
    mycolor = max(mycolor, [0 0 0]);
    mycolor = min(mycolor, [1 1 1]);
    plot3(X(i, 1), X(i, 2), X(i, 3), ...'' , 'Color', mycolor);
end
xlabel('red'), ylabel('green'), zlabel('blue');
xlim([0 1]), ylim([0 1]), zlim([0 1]);
hold off
axis equal
% Rotate the display.
for az=-180:3:180
    view(az,30); % set azimuth, elevation in degrees
drawnow;
end

Code to convert RGB color image to a set of vectors, and plot them
Example (continued)

• Can we use fewer than 3 values per pixel?
• Do PCA on color vectors ... keep the top two PCs.

% Get mean and covariance
mx = mean(X);
Cx = cov(X);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Get eigenvalues and eigenvectors of Cx.  
% Produces V,D such that Cx*V = V*D.  
% So the eigenvectors are the columns of V.
[V,D] = eig(Cx);
e1 = V(:,3);
disp('Eigenvector e1:'), disp(e1);
e2 = V(:,2);
disp('Eigenvector e2:'), disp(e2);
e3 = V(:,1);
disp('Eigenvector e3:'), disp(e3);

d1 = D(3,3);
disp('Eigenvalue d1:'), disp(d1);
d2 = D(2,2);
disp('Eigenvalue d2:'), disp(d2);
d3 = D(1,1);
disp('Eigenvalue d3:'), disp(d3);
Example (continued)

• Construct the “A” matrix, whose rows are the eigenvectors of Cx

```matlab
% Construct matrix A such that the 1st row of A is the eigenvector % corresponding to the largest eigenvalue, the 2nd row is the eigenvector % corresponding to the second largest eigenvalue, etc.
A = [e1'; e2'; e3'];
```

• Project input vectors onto the PCs

```matlab
% Project input vectors x onto eigenvectors. For each (column) vector x, % we will use the equation y = A*(x - mx).
% To explain the Matlab commands below:
% X is our (N,3) array of vectors; each row is a vector.
% mx is the mean of the vectors, size (1,3).
% We first subtract off the mean using X - repmat(mx,N,1).
% We then transpose that result so that each vector is a column.
% We then apply our transform A to each column.
Y = A*(X - repmat(mx,N,1))'; % Y has size 3xN
```
Example (continued)

• Display the y vectors as images

```matlab
% Display y vectors as images
[height,width,depth] = size(RGB);
Y1 = reshape(Y(1,:), height, width);
Y2 = reshape(Y(2,:), height, width);
Y3 = reshape(Y(3,:), height, width);
figure;
subplot(1,3,1), imshow(Y1,[]);
subplot(1,3,2), imshow(Y2,[]);
subplot(1,3,3), imshow(Y3,[]);
```
Example (continued)

- Form the $A_k$ matrix by taking only the top $k$ rows of $A$
- Reconstruct the $x$ vectors using only the top $k$ PC’s:
  $$x' = A_k^T y + m_x$$
- Plot the $x$ vectors in RGB space.
Using the top k=2 principal components for reconstruction

% Reconstruct image using only Y1 and Y2. For each (column) vector y,
% we will use the equation x = A'*y + mx.
% To explain the Matlab commands below:
% Y is our (3,N) array of vectors; where each column is a vector.
% A(1:2,:) is the first two rows of A.
% Y(1:2,:) is the first two rows of Y.
% A(1:2,:)’ * Y(1:2,:) produces our transformed vectors (3xN); we then
% transpose that to make an array of size Nx3, and add the mean.
Xr = ( A(1:2,:)’ * Y(1:2,:) )' + repmat(mx,N,1); % Xr has size Nx3

% Plot reconstructed pixels in color space
figure
hold on
for i=1:100:size(Xr,1)
    mycolor = Xr(i,:);
    mycolor = max(mycolor, [0 0 0]);
    mycolor = min(mycolor, [1 1 1]);
    plot3(Xr(i, 1), Xr(i, 2), Xr(i, 3), ...'
         '.,'Color', mycolor);
end
xlabel('red'), ylabel('green'), zlabel('blue');
xlim([0 1]), ylim([0 1]), zlim([0 1]);
hold off
axis equal
Example (continued)

• Put the $x$ vectors back into the form of an RGB image, using “reshape”.

Original

Reconstructed
% Display original image and reconstructed image
figure, subplot(1,2,1), imshow(RGB), title('Original');

Ir(:,:,1) = reshape(Xr(:,1), height, width);
Ir(:,:,2) = reshape(Xr(:,2), height, width);
Ir(:,:,3) = reshape(Xr(:,3), height, width);
subplot(1,2,2), imshow(Ir), title('Reconstructed');