Sofia Lima

15-686 Neural Computation

Dr. Tai-Sing Lee

22 February, 2023

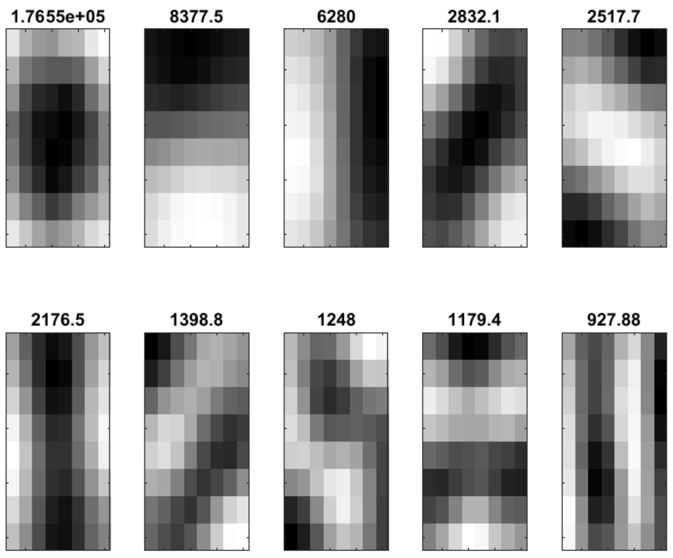
Problem Set 2: Hebbian Learning

Part 1: PCA of natural images (6 pts)

1. Compute sample covariance matrix
    a. Load 10 images; extract & save 500 8x8 sample patches w/ extract_patches.m
    b. Calculate sample covariance matrix
    See C_est.mat

2. Compute the PCs of $\hat{C}$
    a. Perform SVD
    b. Display 10 eigenvectors as patch (enlarge, specify range of intensity variation, contrast normalization)

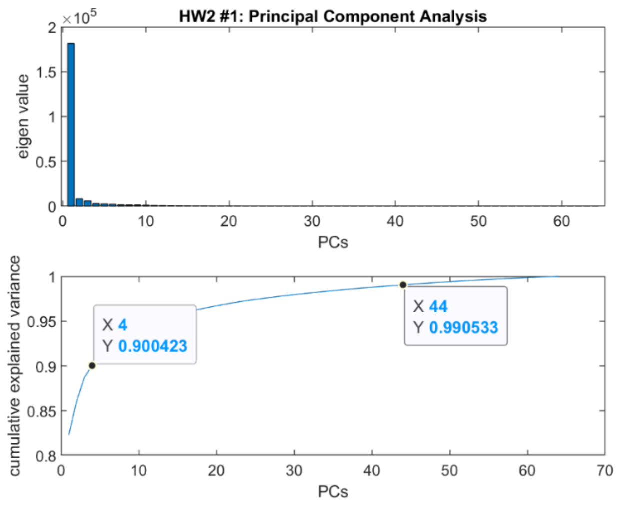
    In the figure below, I am showing the 10 eigenvectors extracted from singular value decomposition (SVD) on the covariance matrix. Because the covariance matrix is square, SVD is equivalent to eigen value decomposition. The eigenvectors are each plotted as an 8x8 contrast normalized patch.



3. Answer

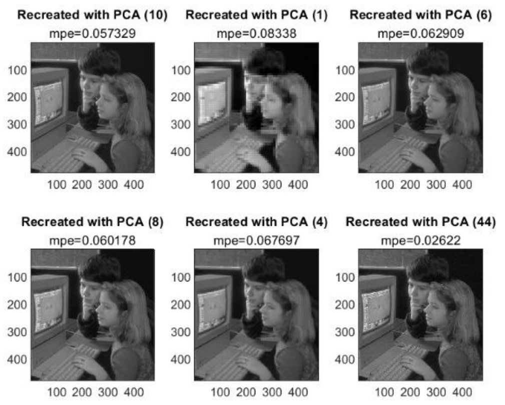
a. Plot 64 eigenvalues. Plot 10 PCs as image patches in two rows with contrast normalization. How many PCs for 90% variance? 99%?

In the figures below, I am showing the 64 eigen values in order of decreasing magnitude. I also plot the cumulative sum, which can then be used to determine that 4 principal components (PCs) are necessary to explain 90% of the variance, and 44 PCs are necessary to explain 99% of the variance. I compute this explained variance by dividing element-wise the cumulative sum by the trace of the singular value matrix S.



b. Divide patches from test image. Project onto 10 PCs. Synthesize image based on 10, 1, 6, 8, 90%, and 99% PCs.

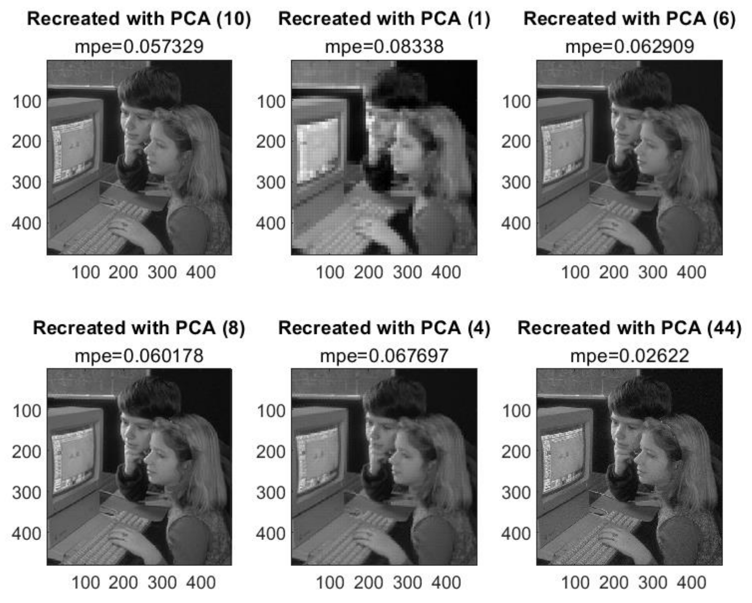
In the figure below, I am showing the original image compared to images synthesized with varying numbers of principal components (PCs). 4 and 44 PCs are required to capture 90% and 99% of the variance in the original training data respectively.



c. Convert original and synthesized to uint8 (hint: look up the function uint8). Compute average error per pixel

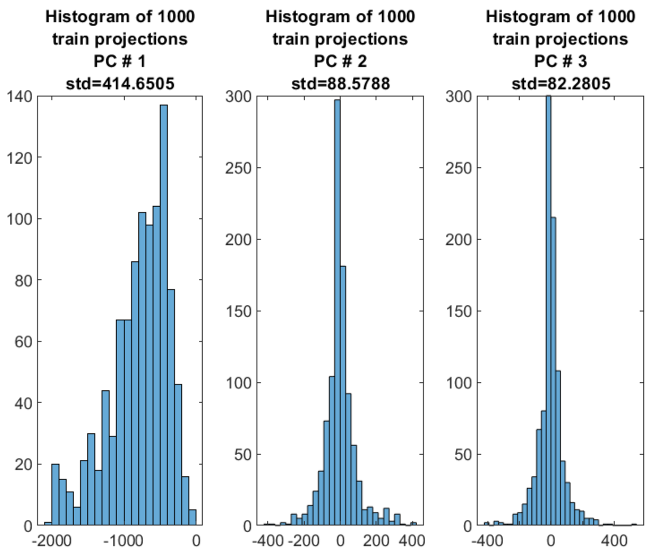
$$MPE = mean_i \frac{\sum_{n,m}\left|\widehat{Y}_i(n,m) - Y_i(n,m)\right|}{\sum_{n,m}\left|Y_{n,m}(t)\right|}$$

In the figure below, I am showing the image reconstructed with varying number of principal components (PCs) and the corresponding MPE. The lowest error is the one reconstructed with 44 PC with mean absolute pixel error 0.02622. The errors increase in relative order when reconstructed with 8, 6, 4, 10, and 1.



d. Plot a histogram of the projection of 1000 samples drawn from the 10 training images onto each one of the top 3 PCs. Are these distributions Gaussian? Compute standard deviations. Relationship between SD and eigenvalues from a?

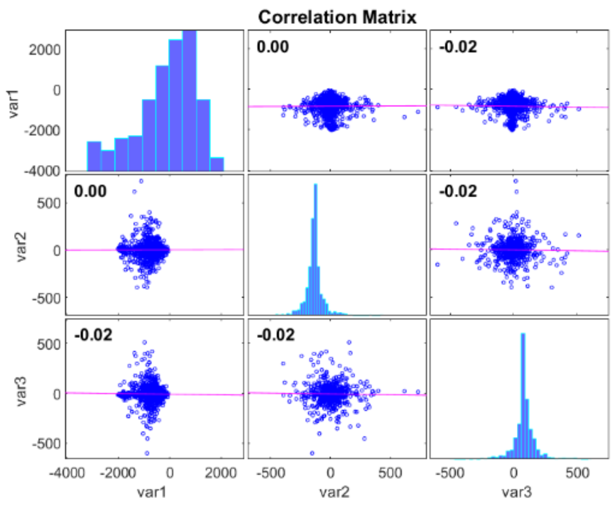
In the figure below, I am showing the histogram of 1000 samples drawn from the 10 training images projected onto each one of the top 3 principal components (PCs) and report the corresponding standard deviations. The distributions for the data projected on PC 2 and 3 do follow Gaussian trends, but the data projected onto the first PC does not have a mean around 0 and is more skewed. The standard deviation of the data projected onto the first PC is 4-fold greater then that of the other PCs. This follows the relationship from part a where the first eigen value (which explains the magnitude of the corresponding direction of variance) is much larger than the second and third.



e. Compute Pearson correlation between each three possible pairs of the sets of coefficients from the 3 PCs from d. Observations consistent with expectation?

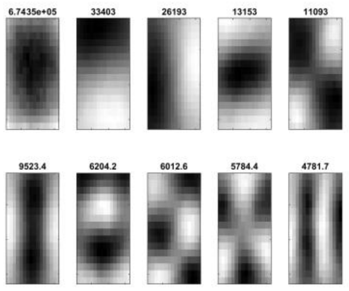
I used the corrplot() function in matlab to generate the figure below. It shows the histogram of the coefficients along the diagonal plots of the corresponding principal component (PC) basis projection. The off-diagonal plots show the correlation between the coefficients obtained by projecting the data onto each of the corresponding PCs. This matrix of Pearson correlation values is symmetric.

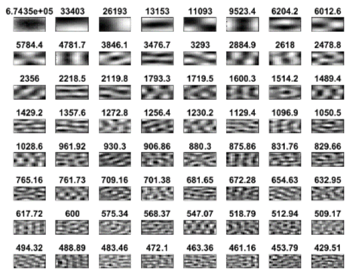
There is little correlation among the sets of coefficients. These observations are consistent with the expectation that there should be little correlation between these coefficients as they should be corresponding to orthogonal basis vectors.



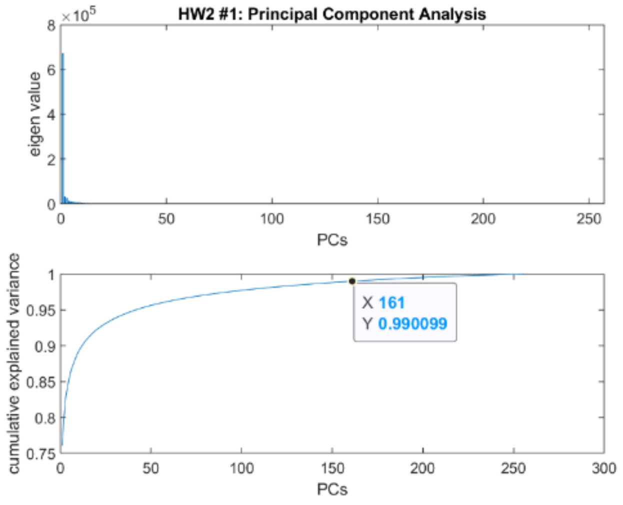
f. Compute the first 10 PCs of the natural image, but with patch size 16x16. Plot the 64 eigenvectors. Plot 10 PCs as 10 image patches in two rows with contrast normalization. Compare and contrast with results from a. How many PCs for 99% variance? Which patch size is more efficient (use fewer numbers) 8x8 PCs vs 16x16 PCs?

In the figure on the left, I am showing the top 10 principal components (PCs) with the highest corresponding eigenvalues in decreasing order, each plotted as a 16x16 contrast normalized patch. In the figure on the right, I am showing 64 of the 256 eigenvectors/PCs, each plotted as a 16x16 contrast normalized patch.





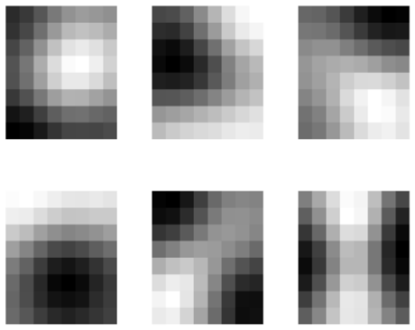
In the figure below, I show that 161 PCs are required to capture 99% of the variance. In this scenario, using an 8x8 patch is more efficient.



Part 2: Hebbian learning of natural images (5 pts) Use Sanger's rule to perform PCA on same data

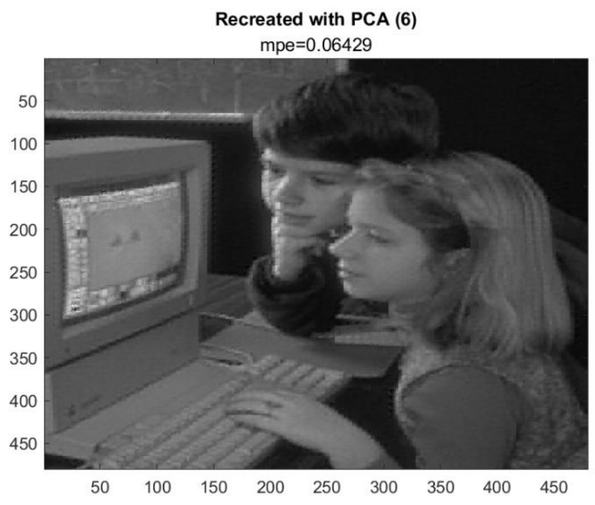
1. Mean center data
2. Use $\eta = 2 * 10^{-10}$
   a. Plot PCs as 8x8 patches. Compare and contrast with part 1.

      In the figure below, I am showing the 8x8 receptive fields learned by Hebbian learning (i.e. learning a PCA model using Sanger's rule). These top 6 PCs are different from those compared to part 1, but do capture some patterns in a visual field. These are the learned PCs which have been updated according to Sanger's rule for 600 iterations.



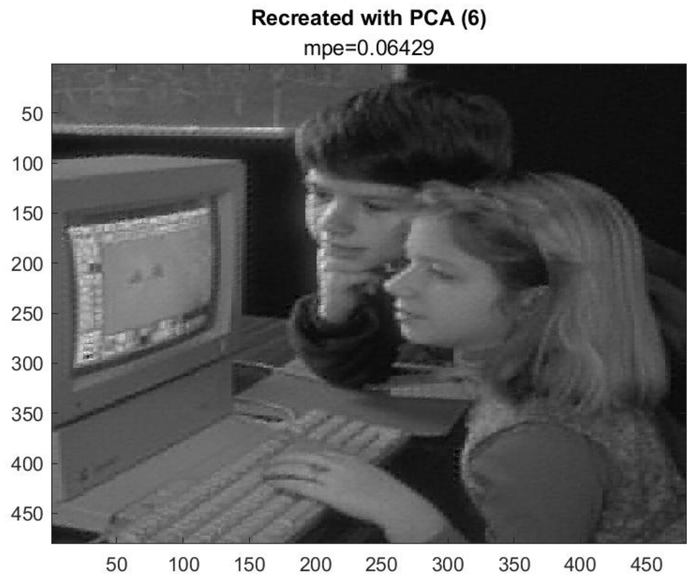
   b. Reconstruct im11.tif using these 6 neurons' receptive fields. Add back the mean. Compare with part 1—is the reconstruction better or worse, qualitatively?

      In the figure below, I am showing the reconstructed images from all top 6 neurons' receptive fields. The image is similar qualitatively to the reconstructions from part 1 where not only gross features are represented, but also finer details.



c. Repeat part 1c for the synthesized image. Compute mean error between reconstructed and original. Compare with part 1c—is the reconstruction better or worse quantitatively?

In the figure below, I am showing the reconstructed images from all top 6 neurons' receptive fields. The image is similar quantitatively to the reconstructions from part 1 where the mean pixel error is around 0.06429.



d. Provide a general discussion, reflection and observations on what you learned in Part 1 and Part 2 so far. Can you explain intuitively what the neurons are learning ? Any why that is a good thing for neurons to learn? Limit to half a page.

The neurons are learning weights corresponding to basis vectors in the direction of greatest variance. The neurons learn these weights by subtracting explained variance from the data at each iteration. This is a good thing for neuron to learn because this allows for sparse encoding, where you only need a few neurons (PCs) to represent an image, rather than having many many neurons to learn unique images.

Part 3: Creative Exploration (1 pt)

Explore the topics or methods covered in this problem set, i.e. the use of PCA or Hebbian Learning, or on topics related to synaptic plasticity in general such as STDP, or other firing-rate based learning rules such as BCM rule. This exploration should begin with a question, and then you perform an experiment to answer this question. The project could involve carrying out a computational experiment or carrying out in-depth research to answer this question. Limit to 2 pages.

In the figures below, I am plotting the results of similar analysis to part 1 and part 2 but with a modification to the firing-rate based learning rule—I include a term $w^T w$, similar to L2 regularization, in the update equation. Adding this term did not affect the quality of the reconstruction, which has a similar mean pixel error (MPE) as without.

On the left are the 6 learned receptor fields (PCs; basis vectors as 8x8 patches) using this modified update function. On the right is the reconstructed image which is very similar to the original image with a low MPE.