HW1 Eigendigits

Due: Friday, January 25, 2013 at 11:59pm

Overview: For this homework you will learn about eigenvector decomposition of covariance for analysis (also known as *principal components analysis* or *PCA*). After projecting images into the reduced eigenspace, you will use a simple classification method (k-nearest neighbors) to see what features the projection captures.

Data: Your training and test data is available here.

```
>> load digits.mat;
>> whos;
```

The training set contains 60000 examples, and the test set 10000 examples. The first 5000 test images are cleaner and easier than the last 5000.

Assignment:

1. **Load the data.** Download the data and load each image into column vectors in MATLAB. You may find the functions `dir` and `imread` useful.
2. **Find eigendigits.** Write a MATLAB function `hw1FindEigendigits` (in `hw1FindEigendigits.m`) that will take an (**x** by **k**) matrix **A** (where **x** is the total number of pixels in an image and **k** is the number of training images) and return a vector **m** of length **x** containing the mean column vector of **A** and an (**x** by **k**) matrix **V** that contains **k** eigenvectors of the covariance matrix of **A** (after the mean has been subtracted). These should be sorted in descending order by eigenvalue (i.e., `V(:,1)` is the eigenvector with the largest associated eigenvalue) and normalized (i.e., `norm(V(:,1)) = 1`). If you're still learning to use MATLAB, I recommend you learn about the following functions: `eig` and `sort`. You can `reshape` a vector and `imshow` it. Note that this assumes that k < x, and you are using the trick on page 14 of the lecture notes (using the page numbers at the bottom of each page) so that the covariance matrix will be k by k instead of x by x.
3. **Experiments.** Don't use all the training data -- it'll be too slow, but experiment with different amounts and find something that seems to work. Display some of the eigenvectors to see what they look like. With the mean and matrix of eigenvectors from a training set of digit data, you can project other datapoints into this eigenspace. Simply subtract the mean vector and multiply by the eigenvector matrix. This will give you a vector of length **k**. Display some reconstructions of the test digits using the projection. Even if your algorithm works perfectly, the reconstructed digits will not look exactly like the originals. What happens if you trim the vector down and only use the top **n** eigenvectors? How accurately can you classify using the method on page 21 of the lecture notes using the Euclidean distance metric? Plot figures showing the accuracy as the number of training points increases, and as the number of eigenvectors increases.
4. **Write and submit.** Write, review, and submit a brief report containing your plots and detailing what you did and how well it worked. At the beginning of your report include your email address and EID in addition to your name. Also submit your code. Submit using `turnin` to `lewfish`. This is **hw1**.