Classification of Handwritten Digits

ASSIGNMENT **(15 grades)**

Read the file hand_written_class.pdf. Construct an algorithm for classification of handwritten digits. Using a training set, and compute the SVD of each class matrix. Use the first few (5-20) singular vectors as basis and classify unknown test digits according to how well they can be represented in terms of the respective bases (use the relative residual vector in the lest squares problem as a measure).

SPECIFIC TASKS

1 Tune the algorithm for accuracy of classification. Give a table or graph of the percentage of correctly classified digits as a function of the number of basis vectors.

2 Check if all digits are equally easy or difficult to classify. Also look at some of the difficult ones, and see that in many cases they are very badly written.

3 Check the singular values of the different classes. Is it motivated to use different numbers of basis vectors for different classes? If so, perform a few experiments to find out if it really pays off to use fewer basis vectors in one or two of the classes.

DATA

1 dzip.txt and azip.txt The first is a vector that holds the digits (the number) and the second is an array of dimension 256 x 1707 that holds the training images. The images are vectors of dimension 256, that have been constructed from 16 x 16 images.

2 dtest.txt and testzip.txt hold the test data.

3 Ima2.m takes an image vector as input and displays it. You have to create a similar function in python.

The data are a subset of the US Postal Service Database.

OPTIONAL TASKS **(10 grades)**

TWO-STAGE ALGORITHM WITH SVD

In order to save operations in the test phase, implement a two-stage algorithm: In the first stage compare the unknown digit only to the first singular vector in each class. If for one class the residual is significantly smaller than for the others, classify as that class. Otherwise perform the algorithm above. Is it possible to get as good results for this variant? How frequently is the second stage unnecessary?

TANGENT DISTANCE

Implement classification using tangent distance. Compute the x-derivatives and y-derivatives of each digit using finite difference approximation. Then in order to compute the tangent distance stack the columns of each derivative so that a vector is obtained. The seven transformations described in the book should be used.

Rewrite the **ima2.m** function so that you can view 20 x 20 images. In the classification, compare each test digit with all the training digits, and classify as the closest.