Jason J. Corso, jcorso@buffalo.edu, SUNY Buffalo
Solutions by David Johnson

Solutions

Problem 1: Recall (2pts) (Answer in one sentence only.)

What quantity is PCA maximizing during dimension reduction?

PCA maximizes the total variance in the transformed dataset.

Problem 2: Work (8 pts) (Show all derivations/work and explain.) The problem of principal component analysis ultimately reduces to the eigenproblem:

$$Se = \lambda e$$

First, describe what the variables S, e and λ represent.

S: The scatter matrix of the data, an unscaled covariance matrix computed via: $\sum_{x_k=1}^{n}(x_k - m)(x_k - m)^T$.

e: The optimal linear projection of our data we wish to find, which minimizes the squared error of our projection and maximizes variance.

λ: A Lagrange multiplier used in our optimization setup, and also a measure of the variance represented by the corresponding e.

Now, given this problem setup for PCA, let's say you are given a very high-dimensional dataset to work with (on the order of 100,000 or even 1,000,000 features)—would it be a good idea to start your analysis by reducing the data's dimensionality with this algorithm? Why or why not?

Probably not—applying the basic PCA algorithm to data of this dimensionality would be problematic, because the algorithm requires first computing, and then computing the eigendecomposition of, the d by d scatter matrix. Even just calculating the matrix has a time complexity of $\mathbf{O}(nd^2)$, and storing it requires $\mathbf{O}(d^2)$ space (about 3.6 terabytes of memory, for 1,000,000 features and 32-bit precision). Actually solving the eigenproblem may be even more expensive.

There are, however, various ways of approximately computing a low-dimensional PCA basis that can be efficiently applied even to very high-dimensional data, and would indeed be good tools to use in this situation.