CS 621: Matrix Computations
Fall 2001
Problem Set 3

Handed out: Fri., Oct. 5.

Due: Wed., Oct. 17 in lecture.

1. Suppose that $A \in \mathbf{R}^{n\times n}$ is ill-conditioned, but AD is well-conditioned, where $D \in \mathbf{R}^{n\times n}$ is a diagonal matrix. Given a linear system $A\mathbf{x} = \mathbf{b}$, we might prefer to solve $AD\hat{\mathbf{x}} = \mathbf{b}$ using a stable algorithm like GEPP, and then set $\mathbf{x} = D\hat{\mathbf{x}}$. (This is called a *column-scaling* of A.) At first glance, it seems that we will get a more accurate answer in this case since we are applying a stable algorithm to a well-conditioned problem. But is it certain that solving $AD\hat{\mathbf{x}} = \mathbf{b}$ will give a better answer for $\mathbf{x}$ than solving $A\mathbf{x} = \mathbf{b}$? Review the relevant theorems and their assumptions to answer this question.

2. Prove the following theorem, which was stated but not proved in lecture. Let A be an $n \times n$ hermitian matrix. Then A is positive semidefinite if and only if it can be factored $A = LDL^H$ where D is real diagonal with nonnegative diagonal entries and L is unit lower triangular. [Hint: Follow the basic outline of the proof given in lecture. The proof needed for this question more difficult because you must also consider the case that $W(1,1) = 0$. Handle this case by showing that if $W(1,1)$ is zero, then the positive semidefiniteness condition implies that the entire first row and column of W must be all zeros, so the factorization can still proceed.]

3. Let A be a positive semidefinite matrix, and let LDL^H be its factorization as in the last question. Show that the rank of A is equal to the number of nonzero diagonal entries of D.

4. It has been proposed to compress images by using the SVD. In particular, assume the image is a grayscale image, represented as a rectangular matrix of floating-point gray values (0=black, 1=white, in between is gray). One could take the SVD of this matrix and then drop the smaller singular vectors and values as in the theorem proved in lecture to come up with an approximation to the original image.

   Take some black-and-white images (perhaps found on the internet) and try this compression method out. How many singular vectors and values are necessary to retain the details of the image? How many to retain the major features? Hand in printouts of your m-files, pictures (original and compressed) and a paragraph of conclusions.

   Note: the Matlab function `imread` is very useful for reading images. The Matlab function `svd` computes singular value decompositions. If you start with a color image, simply add the R,G,B values at each pixel and rescale the sums to make the image grayscale.