CS170A — Mathematical Models & Methods for Computer Science
HW#2 — Matrix computations
Due: 5:00pm Wed January 29, 2003

D. Stott Parker, Blai Bonet, Fotios Konstantinidis
stott@cs.ucla.edu, bonet@cs.ucla.edu, fotios@cs.ucla.edu

1. **Singular Value Decomposition**
   One of the most important and useful results in computational linear algebra is that any $n \times n$ matrix A has a *Singular Value Decomposition (SVD)*

$$A = U \Sigma V$$

   where U and V are unitary matrices (intuitively: rotation-like transformations), and Σ is a diagonal matrix of nonnegative real *singular values*:

$$\Sigma = \begin{pmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_n \end{pmatrix}.$$

   The singular values are a lot like eigenvalues, but they are always real values, and they are never negative.

   In Matlab, the command `[U,Sigma,V] = svd(A);` finds the SVD of `A`.

   In Maple, the command `Sigma := evalf(Svd(A,U,V));` does this, provided that `A` is a matrix of numeric values. (Executing this has the side-effect of binding the variables `U` and `V` to the unitary matrices in the SVD.)

   For each of the following matrices, find the SVD, and determine (yes/no) whether the matrix is: unitary, hermitian, invertible, ill-conditioned, positive definite.

   - in Maple:
     - `A := linalg[fibonacci](5);`
     - `B := linalg[hilbert](8);`
   - in Matlab:
     - `C = hadamard(8);`
     - `D = dingdong(8);`
     - `E = pascal(8);`

     These matrices are defined by m-files in the directory `~cs170ata/www/testmatrices/` or equivalently `http://www.seas.ucla.edu/cs170a/testmatrices/`

   A matrix is called *ill-conditioned* if its condition number $\kappa(A) = \parallel A \parallel \parallel A^{-1} \parallel$ is very large. (Any standard matrix norm will do). Look up the function `cond(A)` in either Maple or Matlab.

   For extra credit: using either Matlab or Maple, find the SVD and determine these properties for three interesting matrices of your choice at the Matrix Market site discussed in class (`http://math.nist.gov/MatrixMarket/`). Matlab I/O routines for reading these matrices are at `http://math.nist.gov/~KRemington/mmio/mmiomatlab.html`.

2. **What Linear Transformations Do**

Write a program in Matlab that takes a *symmetric real* 3×3 *matrix* A and makes a 3-D plot of what A does when applied to the 3-D sphere, including the 3 singular values of A in the title of the plot.

That is, view each point (x, y, z) on the sphere (such that $\sqrt{x^2+y^2+z^2} = 1$) as a 3-dimensional vector $v = [x, y, z]$, and then — for a large set of regularly-spaced points v on the sphere — plot the resulting point $A\,v$.

A similar program has been discussed in class for the 2-D sphere (i.e., the circle) in the Maple worksheet `~cs170ata/www/testmatrices/Eigenvalues.mws` or equivalently `http://www.seas.ucla.edu/cs170a/lecture3/Eigenvalues.mws`

Make a Matlab `movie()` of the output of your program for the following 15 matrices $A(t)$, where $t = 1, 2, \ldots, 15$ is a time parameter:

$$\begin{aligned} A(t) &= Q(t)^\top \, S(t) \, Q(t) \\ S(t) &= \text{diag}(\ 1,\ t,\ 1/t\ ) \\ Q(t) &= R(\ \frac{\pi}{6},\ \frac{\pi}{10}\,t,\ \frac{\pi}{30}\,t\ ). \end{aligned}$$

$R(\ \theta_x,\ \theta_y,\ \theta_z\ )$ is the product of the following three 3-D rotation matrices, which represent rotations around the x, y and z axes by angles θ_x, θ_y and θ_z:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) \\ 0 & \sin(\theta_x) & \cos(\theta_x) \end{pmatrix} \quad \begin{pmatrix} \cos(\theta_y) & 0 & \sin(\theta_y) \\ 0 & 1 & 0 \\ -\sin(\theta_y) & 0 & \cos(\theta_y) \end{pmatrix} \quad \begin{pmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Each angle θ_i is positive in the counterclockwise direction.

Notice: $Q(t)$ is a real orthogonal matrix, and is therefore unitary.

Notice also: $A(t) = Q(t)^\top \, S(t) \, Q(t)$ is a singular value decomposition of $A(t)$.

Please turn in hardcopy of a Matlab `subplot()` with enough frames of the movie (5, say) to show what your program produces.

For extra credit: instead of doing this for a sphere, do it instead on the *globe*. Specifically: the Matlab command `wrldtrv` runs a Matlab demo that plots world travel routes on the globe. Code that generates a 3-D plot of the globe is in the demo files `wrldtrv.m` and `topo.mat` in the `\toolbox\matlab\demos\` subdirectory of the Matlab distribution. Copies of these files are also in `http://www.seas.ucla.edu/cs170a/lecture3/`