Engineering Challenge: Matrix Factorizations

CS 205A: Mathematical Methods for Robotics, Vision, and Graphics (Fall 2013), Stanford University

Due Monday, October 28, at midnight (no late days permitted)

> Your midterm is in class on **October 28**, so you do not have a problem set this week. Problems on this challenge, however, can be used to make up late days (or add some to your reserve), although we will still enforce that no single assignment can be turned in more than five days late.

In the first four weeks of CS 205A, we have covered a large number of matrix factorizations for assorted linear algebra problems. Our main focus is to train you as *clients* of these tools and to help you recognize when they are relevant in different situations. Even so, it is valuable as review to implement each factorization yourself.

Several Matlab function prototypes are listed below (we will provide no additional "starter code"). For each **two** functions you implement correctly, you will receive one extra homework late day for CS 205A. Each function must be accompanied by a testing script in a separate file showing that your method works as advertised on a variety of test cases; you are welcome to check your output against that of the corresponding built-in Matlab methods. Feel free to change function prototypes so long as they carry out the specified task.

```
function [L,U] = lu_factorize_cs205a(A)
% Factors a square matrix A into a product L*U where
%  - L is lower triangular
%  - U is upper triangular
% It is OK to assume A admits such a factorization
```

```
function [L,U,P] = lup_factorize_cs205a(A)
% Factors square matrix A such that L*U = P*A where
%  - L is lower triangular
%  - U is upper triangular
%  - P is a permutation matrix
```

```
function x = solve_using_lu_cs205a(L,U,b)
% Solves A*x=b assuming A is factored as L*U using forward/backward substitution.
```

```
function L = cholesky_cs205a(A)
% Factors symmetric, positive definite matrix A into a product L*L^T
```

```
function [Q,R] = qr_cs205a(A)
% Factors potentially non-square matrix A into Q*R using the "reduced"
% factorization (Q may be non-square)
```

```
function values = eigenvalues_cs205a(A)
% Returns a vector of all the eigenvalues of A, assuming A is symmetric.
```

```
function [U,Sigma,V] = svd_cs205a(A)
% Computes the SVD of the matrix A.
```

```
function x = apply_pseudoinverse_cs205a(A,b)
% Returns A^+ * b.
```