CS4220 Assignment 3 Due: 3/1/14 (Sat) at 6pm

You must work either on your own or with one partner. You may discuss background issues and general solution strategies with others, but the solutions you submit must be the work of just you (and your partner). If you work with a partner, you and your partner must first register as a group in CMS and then submit your work as a group. Each submitted m-file is worth 5 points. One point may be deducted for poor style.

Topics: Band systems. Cholesky factorization. LDL with Symmetric pivoting. Sparse factorizations.

1 Downward Displacement of a Cantilevered Beam

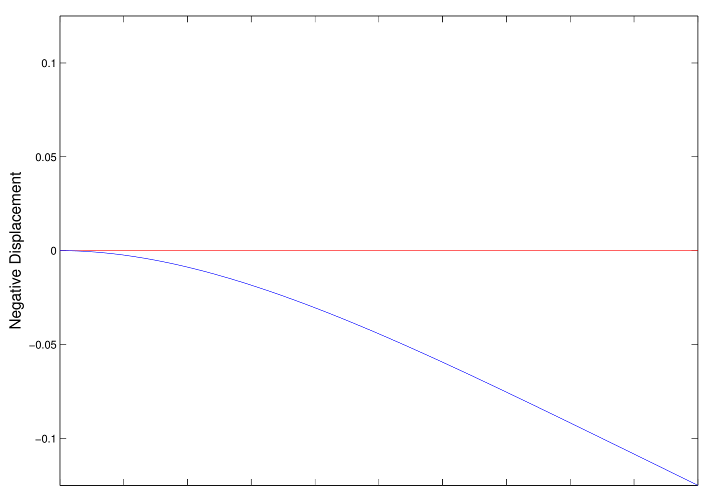
Define (by example) the n-by-n matrices A_n and U_n by

$$A_n = \begin{bmatrix} 9 & -4 & 1 & 0 & 0 & 0 & 0 & 0 \\ -4 & 6 & -4 & 1 & 0 & 0 & 0 & 0 \\ 1 & -4 & 6 & -4 & 1 & 0 & 0 & 0 \\ 0 & 1 & -4 & 6 & -4 & 1 & 0 & 0 \\ 0 & 0 & 1 & -4 & 6 & -4 & 1 & 0 \\ 0 & 0 & 0 & 1 & -4 & 6 & -4 & 1 \\ 0 & 0 & 0 & 0 & 1 & -4 & 5 & -2 \\ 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 \end{bmatrix} \qquad (n = 8)$$

$$U_n = \begin{bmatrix} 2 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad (n = 8)$$

Thus, the diagonal of A is all 6's except $A_n(1,1) = 9$, $A_n(n-1, n-1) = 5$, and $A_n(n,n) = 1$. It can be shown that $A_n = U_n U_n^T$ and thus, A_n is positive definite.

The solution to the linear system $A_n d = \texttt{ones}(n,1)/n^4$ approximates the displacement of a length-1 cantilevered beam that is fixed at the origin. It looks something like this:



In particular, $d(k)$ is the downward displacement of the beam at $x = k/n$ for $k = 1{:}n$. In this problem you solve the linear system $A_n d = \texttt{ones}(n,1)/n^4$ two ways.

In the first method you are to make use of the factorization $A_n = U_n U_n^T$ given above. In particular, you are to write a function `d = Beam1(n)` that uses this factorization to solve the linear system $A_n d = \texttt{ones}(n,1)/n^4$. Your implementation should require just a single n-vector of storage. Intuition tells us that $0 < d_1 < \cdots < d_n$. That is, the displacement increases as we walk out to the end of the beam. Add comments at the bottom of `Beam1` that explains why the output vector d has this property assuming exact arithmetic.

For the second method you are to use `chol`. In particular, you are to write a function `[d,condA] = Beam2(n)` that (a) sets up the $\texttt{triu}(A_n)$ in sparse format, (b) uses `chol` to compute the sparse Cholesky factorization $A_n = R^T R$ where R is upper triangular, (c) uses R to solve $A_n d = \texttt{ones}(n,1)/n^4$, and (d) uses `condest` to estimate the condition of A_n. For full credit, your implementation of `Beam2` should call `sparse` just once. (You may want to check out some of the "sparse" demos from the second week of class.)

The test script `ShowBeam` can be used to check things out. Submit `Beam1` and `Beam2` to CMS.

2 A Direction of Negative Curvature

If $A \in \mathbb{R}^{n\times n}$ is symmetric positive definite, then for all nonzero vectors $x \in \mathbb{R}^n$ we have $x^T Ax > 0$. If A is not positive definite, then there is a nonzero vector $x \in \mathbb{R}^n$ such that $x^T Ax \le 0$. These properties are preserved under *congruence transformations*. If A is symmetric and Z is nonsingular, then $B = Z^T AZ$ is said to be *congruent* to A. Noting that

$$x^T Bx = (Zx)^T A(Zx)$$

we see that $x^T Bx > 0$ for all nonzero x if A is positive definite. Likewise, if A is not positive definite, then we can find an x so that $x^T Bx \le 0$. In this problem you are to implement the following function:

```
  function x = NegativeCurvature(A)
% A is an nxn symmetric matrix. If A is positive definite then
% x is the empty vector. Otherwise, x is a unit 2-norm n-vector with the
% property that x'*Ax <= 0.
```

Your implementation should exploit the above facts about congruence transformations and it must make effective use of the following function that can be downloaded from the website:

```
  function [L,D,P,k] = LDLTpiv(A)
% A is an nxn symmetric matrix.
% k is an integer that satisfies 1<=k<=n+1.
% If k==n+1, then A is positive definite and PAP' = LDL' where P is a permutation
%    matrix, L is unit lower triangular, and D is diagonal.
% Otherwise, A is not positive definite and PAP' = LDL' where P is a
%    permutation matrix, L is unit lower triangular with L(k:n,k:n) = eye(n-k+1,n-k+1),
%    D(1:k-1,1:k-1) is diagonal with positive diagonal entries, and D(k,k)<=0.
```

Hints. If A is not positive definite, then `LDLTpiv` computes the factorization

$$PAP^T = LDL^T = \left[\begin{array}{c|c} L_{11} & 0 \\ \hline L_{21} & I_{n-k+1} \end{array}\right] \left[\begin{array}{c|c} \text{diag}(d_1,\ldots,d_{k-1}) & 0 \\ \hline 0 & D(k{:}n,k{:}n) \end{array}\right] \left[\begin{array}{c|c} L_{11} & 0 \\ \hline L_{21} & I_{n-k+1} \end{array}\right]^T$$

and $D(k,k) <= 0$. Here is what the D matrix might look like in this situation:

$$D = \left[\begin{array}{cc|ccc} 3 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ \hline 0 & 0 & -1 & 5 & 7 \\ 0 & 0 & 5 & -4 & 6 \\ 0 & 0 & 7 & 6 & -8 \end{array}\right] \qquad (n = 5,\ k = 3)$$

Note that A and D are congruent so you will want to think about $w^T Dw \le 0$. Submit your implementation of `NegativeCurvature` to CMS. A test script `ShowNegCurve` can be downloaded from the course website.

3 Sparse Cholesky and Sparse LU

Download `SparseFactorizations.zip` from the syllabus page. Unzip the file and move the resulting folder into your Matlab workspace. Study and run the demos `ShowSparseChol` and `ShowSparseLU`. These demos feed off of some `.mat` files that were downloaded from the UF Sparse Matrix Collection. Study the `README` file which tells you how to interact with the UF Collection. In this problem you are to use that same collection to produce two demo files of your own.

Find a symmetric positive definite example in the collection with the property that $n > 2000$. (Your example must be different from those that are used by `ShowSparseChol`.) Name the downloaded file `MySPD.mat`. Write a script `ShowMySPD` that produces the same display for your example as `ShowSparseChol` does for each of the given examples. To do this just copy `ShowSparseChol` into `ShowMySPD.m`, remove the loop, and set `A = GetMatrix('MySPD.mat')`. In addition your script should

- Print the UF id of your example and the value of n.
- Solve $Ax = A \cdot \texttt{ones}(\texttt{n}, 1)$ using the Cholesky factorization of A. Report the relative error, the time required to compute the factorization, and the time required to solve the system using the factors.
- Solve $Ax = A \cdot \texttt{ones}(\texttt{n}, 1)$ using the Cholesky factorization of A with minimum degree pivoting. Report the relative error, the time required to compute the factorization, and the time required to solve the system using the factors.
- Solve $Ax = A \cdot \texttt{ones}(\texttt{n}, 1)$ using the Cholesky factorization of A with reverse Cuthill-McKee ordering. Report the relative error, the time requitred to compute the ordering and the factorization, and the time required to solve the system using the factors.

Submit `MySPD.mat` and `ShowMySPD.m` to CMS. When we test your code, `GetMatrix` will be in the working directory.

Find a general unsymmetric example in the collection with the property that $n > 2000$. (Your example must be different from those that are used by `ShowSparseLU`.) Name the downloaded file `MyGEN.mat`. Write a script `ShowMyGEN` that produces the same display for your example as `ShowSparseLU` does for each of the given examples. To do this just copy `ShowSparseLU` into `ShowMyGEN.m`, remove the loop, and set `A = GetMatrix('MyGEN.mat')`. In addition your script should

- Print the UF id of your example and the value of n.
- Solve $Ax = A \cdot \texttt{ones}(\texttt{n}, 1)$ using the LU factorization of A with threshold pivoting. Report the relative error, the time required to compute the factorization, and the time required to solve the system using the factors.

Submit `MyGEN.mat` and `ShowMyGEN.m` to CMS. When we test your code, `GetMatrix` will be in the working directory.