CS 219: Sparse Matrix Algorithms // Homework 1

Due by class time Monday, April 8

Please turn in all your homework on paper, either in class or in the homework box in the Computer Science Department office. For programs, turn in (1) code listing; (2) sample output and any plots, showing thorough testing; (3) a transcript of a Matlab session showing how it's used.

Problem 1. The purpose of this problem is to prove that 0 is a simple eigenvalue of the Laplacian matrix of a connected graph. Let G be a connected, undirected graph, whose vertices V are the integers 1 through n and whose edges E are unordered pairs (i, j) of vertices/integers. Let $A = L(G)$ be the n-by-n Laplacian matrix of G.

1(a) Let $G^{(ij)}$ be a the graph with n vertices and only one edge (i, j), and $A^{(ij)} = L(G^{(ij)})$ be its n-by-n Laplacian matrix. Show that $A = \sum_{(i,j)\in E} A^{(ij)}$.

1(b) Let $x = (x_1, x_2, \ldots, x_n)^T$ be a column vector. What is the scalar $x^T A^{(ij)} x$, in terms of the x_i's?

1(c) Prove that the scalar $x^T Ax$ is zero if and only if all the elements of x are the same. (You will need to use the fact that G is connected.)

1(d) Use (1c) to conclude that (i) 0 is an eigenvalue of A and (ii) every eigenvector w with $Aw = 0w$ is a multiple of the vector $(1, 1, \ldots, 1)^T$, and therefore 0 is a simple eigenvalue.

1(e) Extra credit: Prove that, if G has $k > 0$ connected components, then 0 has multiplicity exactly k as an eigenvalue of $A = L(G)$. (That is, show that there are k, and not more, mutually orthogonal vectors $w^{(1)}, \ldots, w^{(k)}$ with $Aw^{(i)} = 0$ for $1 \le i \le k$. Recall that vectors u and v are *orthogonal* if $u^T v = 0$.)

Problem 2. The purpose of this problem is just to get your Matlab environment set up for playing with sparse matrices. You can run Matlab on any machine you want. (You can get an account on CSIL if you're not a CS student, as long as you're enrolled in a CS course.)

2(a) Download the mesh partitioning toolbox from `www.cerfacs.fr/algor/Softs/MESHPART/` and try running `meshdemo`.

2(b) Go to the UFL Sparse Matrix Collection at `www.cise.ufl.edu/research/sparse/matrices/`. The website has tools to download the matrices in several formats, using several interfaces (including "UFget", which imports matrices directly into Matlab, as well as a GUI interface and some

Java and Python tools). Set up your preferred method of download, and download a few matrices to play with.

2(c) Try using Matlab to solve linear systems with some of the matrices from the UFL collection. (A good right-hand-side vector to use is `b = sum(A')'`, the vector of row sums of A, because then you know that the right answer to $Ax = b$ is the vector of all 1's.) What's the biggest system from the collection that you can solve in reasonable time in Matlab?

Problem 3. Let G be the 9-vertex "grid graph" corresponding to the two-dimensional model problem on a 3-by-3 mesh. Suppose the vertices of G are numbered 1 through 9 by rows from upper left to bottom right.

3(a) Draw the sequence of ten so-called "elimination graphs" (beginning with G and ending with the empty graph) that result from playing the vertex elimination game on G with the given vertex numbering.

3(b) Draw the 9-vertex "filled graph" G^+ consisting of G plus fill edges.

3(c) In Matlab, create a 9-by-9 symmetric, positive definite matrix A that has G as its nonzero structure (that is, such that $G(A) = G$). Use the Matlab functions `spy()` and `chol()` to visualize the nonzero structure of A and of its Cholesky factor, verifying that the nonzero structure of the Cholesky factor is $G^+(A)$. Print your `spy` plots and turn them in with your homework.

3(d) Find a way to renumber the vertices so that there is less fill. That is, find a permutation p such that $G^+(A(p,p))$ has fewer edges than $G^+(A)$. Draw the filled graph $G^+(A(p,p))$, and the print the `spy` plots of `A(p,p)` and its Cholesky factor.