Homework 1

CS 290 H : Sparse Matrix Algorithms : Spring 2014

Due by Friday, April 18

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

Problem 1. Incidence matrix. Let G be an undirected graph with n vertices and m edges. An *incidence matrix* of G (sometimes called a *vertex-edge incidence matrix*) is an n-by-m matrix U, not square in general, with one column for each edge of G. The column corresponding to edge (i, j) has just two nonzeros, one in row i and one in row j, one with value 1 and one with value -1. Notice that the incidence matrix isn't unique—I didn't specify which entry in each column is positive and which is negative, and I also didn't specify which column corresponds to each edge.

1(a) Choose any 5-vertex graph. Draw a picture of that graph, with vertices labeled 1 through 5. Display both the Laplacian matrix and an incidence matrix of that graph.

1(b) Prove that if U is any incidence matrix for a graph G, then $L = UU^T$ is the (unique) Laplacian matrix of G.

1(c) Suppose graph G is connected. What is the rank of U? Justify your answer.

1(d) A *tree* is a connected, undirected graph with no cycles. Any tree with n vertices has $n-1$ edges. A *spanning tree* of a connected graph G is a subgraph of G that is a tree and that includes all the vertices of G.

Let G and U be as above, let T be some spanning tree of G, and let V be an incidence matrix of T. Note that V is n-by-$(n-1)$. Give two proofs that there exists an $(n-1)$-by-m matrix W with $U = VW$: first, by reasoning about the ranks and column spaces of the matrices; and, second, by an explicit combinatorial construction of W, one column at a time, using the fact that an edge of G corresponds to both a column of U and a column of W.

Problem 2. Augmented matrix. Let G be an undirected graph with n vertices and m edges, let L be the Laplacian matrix of G, and let U be an incidence matrix of G. An *augmented matrix* of G is a square, symmetric $(n+m)$-by-$(n+m)$ matrix B defined in block form as

$$B = \begin{pmatrix} I & U^T \\ U & 0 \end{pmatrix}.$$

Suppose we are given an n-vector b and we want to solve the Laplacian linear system $Lx = b$ for the n-vector x. Give a linear system $By = c$ with the augmented matrix whose solution y contains the solution to $Lx = b$.

Problem 3. Matlab. 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.)

If you've never used Matlab before, it's pretty simple. Try running it and reproducing some of the demos I've done in class; the transcripts are on the class web site, as is the Matlab code I was using. Saying "help" gets you some general help; "help foo" gets you information on the function `foo()`. Try saying "help meshpart", and then take a look at the Matlab source code in the `matlab/graphs/meshpart` directory (from the course web site) to see what it looks like.

If there is enough demand I will set up a Matlab tutorial session outside of class one day next week.

3(a) In Matlab, type in the Laplacian matrix L of your 5-vertex graph from problem (1a). Use `eig()` to find its eigenvalues and eigenvectors (say "`help eig`" for instructions). Either by hand or by using Matlab functions, create the incidence matrix U of L. Verify in Matlab that $UU^T = L$.

Use the following Matlab incantation to form the augmented matrix B:

```
I = speye(something);   % sparse identity matrix, you choose "something"
Z = sparse(something);  % sparse zero matrix, you choose "something"
B = [I U' ; U Z];       % the augmented matrix
B                       % print B in sparse format
full(B)                 % print B in dense format
```

Use `eig(full(B))` to compute the eigenvalues and eigenvectors of B, and compare them to the eigenvalues and eigenvectors of L.

3(b) Go to the University of Florida Sparse Matrix Collection at

`http://www.cise.ufl.edu/research/sparse/matrices/`

The UF Collection has groups of matrices from many sources; some of them come from graphs. Browse the collection and find a few matrices that come from undirected graphs. You might try looking in the Pajek group,

`http://www.cise.ufl.edu/research/sparse/matrices/Pajek/index.html`

The UF website has tools to search and download the matrices in several formats, including "UFget", which imports matrices directly into Matlab, a Java GUI interface, and some Java and Python tools. Set up your preferred download technique (I just use UFget in Matlab), and download a few undirected graphs of various sizes to play with.

3(c) Try using Matlab to find the Fiedler value and vector (second smallest eigenvalue of the Laplacian matrix and its eigenvector) for the graphs you downloaded. You can try both Matlab's `eigs()` routine and the `fiedler()` routine from the meshpart toolbox on the course web site. What's the biggest graph from the collection you can find a Fiedler vector for within a minute of time on Matlab on a workstation?

A couple of hints: The Pajek group contains both undirected and directed graphs (symmetric and unsymmetric matrices); stick to the undirected ones, and ignore any edge weights. You can use the meshpart toolbox `laplacian()` routine to convert any square matrix to the Laplacian matrix of the corresponding unweighted undirected graph. If one of your matrices is not connected, its Fielder value will be 0; you might try getting its largest connected component (with meshpart routine `components()`) and finding the Fiedler value and vector for that.