LECTURE NOTES
David F. Gleich
September 29, 2011

Homework 1

Due September 20th, 2011

Problem 1: Norms

a) Show that $f(\mathbf{x}) = \|\boldsymbol{A}\mathbf{x}\|_p$ is a vector-norm, where $\boldsymbol{A}$ is a non-singular matrix.

Solution Because $\boldsymbol{A}$ is non-singular, $\boldsymbol{A}\mathbf{x} = 0$ implies that $\mathbf{x} = 0$. Consequently, by the standard properties of a norm, we know that $f(\mathbf{x}) \geq 0$, and $f(\mathbf{x}) = 0$ if and only $\mathbf{x} = 0$. The other two properties follow immediately from the properties of the vector norms and the properties of matrix multiplication.

b) Show that $f(\mathbf{x}) = \|\boldsymbol{A}\mathbf{x}\|_p$ is not a vector-norm if $\boldsymbol{A}$ is singular.

Solution When $\boldsymbol{A}$ is singular, there is a vector $\mathbf{x}$ such that $\boldsymbol{A}\mathbf{x} = 0$. This vector violates the first property of being a norm.

These norms will arise in our study of spectral graph theorem. In those cases, the matrix $\boldsymbol{A}$ is usually the diagonal matrix of degrees for each node – commonly written $\boldsymbol{D}$.

Problem 2

There are a tremendous number of matrix norms that arise. An interesting class are called the *orthgonally invariant norms.* Norms in this class satisfy:

$$\|\boldsymbol{A}\| = \|\boldsymbol{U}\boldsymbol{A}\boldsymbol{V}\|$$

for *square orthogonal matrices* $\boldsymbol{U}$ and $\boldsymbol{V}$. Recall that a square matrix is orthogonal when $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$, i.e. $\boldsymbol{U}^{-1} = \boldsymbol{U}^T$.

a) Show that $\|\boldsymbol{A}\|_F$ is orthogonally invariant. (Hint: use the relationship between $\|\boldsymbol{A}\|_F$ and $\operatorname{trace}(\boldsymbol{A}^T\boldsymbol{A})$.)

Solution For the trace operator, $\operatorname{trace}(AB) = \operatorname{trace}(BA)$ so, we have

$$\|\boldsymbol{U}\boldsymbol{A}\boldsymbol{V}\|_F^2 = \operatorname{trace}(\boldsymbol{V}^T\boldsymbol{A}^T\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{A}\boldsymbol{V}) = \operatorname{trace}(\boldsymbol{V}^T(\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{V})) = \operatorname{trace}((\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{V})\boldsymbol{V}^T) = \operatorname{trace}(\boldsymbol{A}^T\boldsymbol{A}) = \|\boldsymbol{A}\|_F^2.$$

b) Show that $\|\boldsymbol{A}\|_2$ is orthogonally invariant. (Hint: first show that $\|\boldsymbol{U}\mathbf{x}\|_2 = \|\mathbf{x}\|_2$ using the relationshp between $\|\mathbf{x}\|$ and $\mathbf{x}^T\mathbf{x}$.)

Solution Note that $\|\mathbf{x}\|^2 = \sum_i x_i{}^2 = \mathbf{x}^T\mathbf{x}$. Consequently, $\|\boldsymbol{U}\mathbf{x}\| = \sqrt{\mathbf{x}^T\boldsymbol{U}^T\boldsymbol{U}\mathbf{x}} = \|\mathbf{x}\|$.

Hence,

$$\|\boldsymbol{U}\boldsymbol{A}\boldsymbol{V}^T\|_2 = \max_{\mathbf{x}} \frac{\|\boldsymbol{U}\boldsymbol{A}\boldsymbol{V}^T\mathbf{x}\|_2}{\|\mathbf{x}\|_2} = \max_{\mathbf{x}} \frac{\|\boldsymbol{A}\boldsymbol{V}^T\mathbf{x}\|_2}{\|\mathbf{x}\|_2} = \max_{\mathbf{x}} \frac{\|\boldsymbol{A}\boldsymbol{V}^T\mathbf{x}\|_2}{\|\boldsymbol{V}^T\mathbf{x}\|_2} = \max_{\mathbf{y}=\boldsymbol{V}^T\mathbf{x}} \frac{\|\boldsymbol{A}\mathbf{y}\|_2}{\|\mathbf{y}\|_2} = \|\boldsymbol{A}\|_2$$

where the second to last expression follows because $\mathbf{y}$ can be any vector because $\boldsymbol{V}$ is a square orthogonal matrix.

Problem 3

In this problem, we'll work through the answer to the challenge question on the introductory survey.

Let $\boldsymbol{A}$ be the adjacency matrix of a simple, undirected graph.

a) **An upper bound on the largest eigenvalue**
Show that $\lambda_{\max}(\boldsymbol{A})$ is at most, the maximum degree of the graph. Show that this bound is tight.

Solution $\lambda_{\max} \le \rho(\boldsymbol{A}) \le \|\boldsymbol{A}\|$ where $\rho(\boldsymbol{A})$ is the spectral radius, the largest magnitude of any eigenvalue. The bound follows because the 1-norm of the $\boldsymbol{A}$ is the largest degree.

Any constant degree graph, e.g. a clique, has this as the largest eigenvalue.

b) **A lower bound on the largest eigenvalue** Show that $\lambda_{\max}(\boldsymbol{A})$ is at least, the square-root of the maximum degree of the graph. Show that this bound is tight. (Hint: try and find a lower-bound on the Rayleigh-Ritz characterization $\lambda_{\max} = \max \mathbf{x}^T \boldsymbol{A}\mathbf{x}/\mathbf{x}^T\mathbf{x}$.)

Solution Let $\boldsymbol{A}_S$ be the adjacency matrix for a graph with fewer edges than $\boldsymbol{A}$. Note that

$$\lambda_{\max} = \max_{\mathbf{x}} \mathbf{x}^T \boldsymbol{A}\mathbf{x}/(\mathbf{x}^T\mathbf{x}) \ge \max_{\mathbf{x}} \mathbf{x}^T \boldsymbol{A}_S\mathbf{x}/(\mathbf{x}^T\mathbf{x}) \ge \mathbf{y}^T \boldsymbol{A}_S\mathbf{y}/(\mathbf{y}^T\mathbf{y}).$$

for any vector $\mathbf{y}$. Let r be the vertex with maximum degree. Set $\boldsymbol{A}_S$ to be the adjacency matrix only for the edges that constitute the maximum edgree, then $\boldsymbol{A}_S$ is the matrix for a star-graph centered at r. Also set

$$[\mathbf{y}]_i = \begin{cases} 0 & i \ne r, i \notin \Gamma(r) \\ \sqrt{d_{\max}} & i = r \\ 1 & i \in \Gamma(r) \end{cases}.$$

Equivalently, $\mathbf{y} = \mathbf{e}_S - (1 - \sqrt{d_{\max}})\mathbf{e}_r$ (where $\mathbf{e}_S$ has 1s only on the set of vertices in the star.

Then $\mathbf{y}^T \boldsymbol{A}_S \mathbf{y} = \underbrace{\mathbf{e}_S^T \boldsymbol{A}_S \mathbf{e}_S}_{=2d_{\max}} -2(1-\sqrt{d_{\max}}) \underbrace{\mathbf{e}_r^T \boldsymbol{A}_S \mathbf{e}_S}_{=d_{\max}}, and \mathbf{y}^T\mathbf{y} = 2d_{\max}$

by a direct calculation.
Taking these ratios gives the lower-bound of $\sqrt{d_{\max}}$.

Problem 4

In this question, we'll show how to use these tools to solve a problem that arose when Amy Langville and I were studying ranking algorithms.

a) **the quiz from class** Let $\boldsymbol{A}$ be an $n \times n$ matrix of all ones:

$$\boldsymbol{A} = \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & & \vdots \\ 1 & \cdots & 1 \end{bmatrix}.$$

What are the eigenvalues of $\boldsymbol{A}$? What are the eigenvectors for all non-zero eigenvalues? Given a vector $\mathbf{x}$, how can you tell if it's in the *nullspace* (i.e. it's eigenvector with eigenvalue 0) without looking at the matrix?

Solution The eigenvalues are n and 0. A null-vector must have sum 0 because the eigenvalue n is associated with the vector of all constants, and all other vectors must be orthogonal, e.g. $\mathbf{e}^T\mathbf{x} = 0$ for any vector in the nullspace.

b) **my problem with Amy** Amy and I were studying the $n \times n$ matrix:

$$\boldsymbol{A} = \begin{bmatrix} n & -1 & \cdots & -1 \\ -1 & \ddots & & \vdots \\ \vdots & & \ddots & -1 \\ -1 & \cdots & -1 & n \end{bmatrix}$$

that arose when we were looking at ranking problems like we saw in http://www.cs.purdue.edu/homes/dgleich/nmcomp/lectures/lecture-1-matlab.m What we noticed was that Krylov methods to solve

$$\boldsymbol{A}\mathbf{x} = \mathbf{b}$$

worked incredibly fast.
Usually this happens when $\boldsymbol{A}$ only has a few *unique* eigenvalues. Show that this is indeed the case. What are the *unique* eigenvalues of $\boldsymbol{A}$?

Note There was a typo in this question. It should have been an $n \times n$ matrix, which makes it non-singular. Anyway, we'll solve the question as written.

Solution The eigenvalues of this matrix are just a shift away. We start with a single eigenvalue equal to $n+1$, and we shift all the eigenvalues in a positive direction by n+1, e.g. we write $\boldsymbol{A} = (n+1)I - \boldsymbol{E}$ where $\boldsymbol{E} = \mathbf{e}\mathbf{e}^T$ is the matrix of all ones.
Hence, we'll have $n+1$ eigenvalues equal to $n+1$.

c) **solving the system** Once we realized that there were only a few unique eigenvalues and vectors, we wanted to determine if there was a closed form solution of:

$$\boldsymbol{A}\mathbf{x} = \mathbf{b}.$$

There is such a form. Find it. (By closed form, I mean, given $\mathbf{b}$, there should be a simple expression for $\mathbf{x}$.)

Solution If the sum of $\mathbf{b}$ is non-zero, then there isn't a solution. i.e. we need $\mathbf{e}^T\mathbf{b} = 0$ to have a solution. Now we just have to determine $\mathbf{x}$ where

$$[(n+1)\boldsymbol{I} - \mathbf{e}\mathbf{e}^T]\mathbf{x} = \mathbf{b}.$$

Let $\mathbf{e}^T\mathbf{x} = \gamma$, then

$$\mathbf{x} = (\mathbf{b} - \gamma\mathbf{e})/(n+1).$$

So we already know that $\mathbf{x}$ is given by a rescaled $\mathbf{b}$. Note that $\mathbf{x}$ is a solution for any value of γ, so there is an infinite family of solutions. The simplest is just $\mathbf{b}/(n+1)$.

Problem 5

In this question, you'll implement codes to convert between triplet form of a sparse matrix and compressed sparse row.

You may use any language you'd like.

a) Describe and implement a procedure to turn a set of triplet data this data into a one-index based set of arrays: `pointers,␣columns,␣and␣values` for the compressed sparse form of the matrix. Use as little additional memory as possible. (Hint: it's doable using *no* extra memory.)

```
function [pointers, columns, values] = sparse_compress(m, n, triplets)
% SPARSE_COMPRESS Convert from triplet form
%
% Given a m-by-n sparse matrix stored as triplets:
%   triplets(nzi,:) = (i,j,value)
% Output the  the compressed sparse row arrays for the sparse matrix.

% SOLUTION from https://github.com/dgleich/gaimc/blob/master/sparse_to_csr.m

pointers = zeros(m+1,1);
nz = size(triplets,1);
values = zeros(nz,1);
```

```
columns = values(nz,1);
% build pointers for the bucket-sort
for i=1:nz
    pointers(triplets(i,1)+1)=pointers(triplets(i,1)+1)+1;
end
rp=cumsum(rp);
for i=1:nz
    values(pointers(triplets(i,1))+1)=triplets(i,3);
    columns(pointers(triplets(i,1))+1)=triplets(i,2);
    pointers(triplets(i,1))=pointers(triplets(i,1))+1;
end
for i=n:-1:1
    pointers(i+1)=pointers(i);
end
pointers(1)=0;
pointers=pointers+1;
```

b) Describe and implement a procedure to take in the one-indexed compressed sparse row form of a matrix: `pointers,␣columns,␣and␣values` and the dimensions `m,␣n` and output the compressed sparse row arrays for the transpose of the matrix:

```
function [pointers_out, columns_out, values_out] = sparse_transpose(...
m, n, pointers, columns, values)
% SPARSE_TRANSPOSE Compute the CSR form of a matrix transpose.
%
%

triplets = zeros(pointers(end),3);

% SOLUTION
for row=1:m
  for nzi=pointers(row):pointers(row+1)-1
    triplets(nzi,1) = columns(nzi);
    triplets(nzi,2) = row;
    triplets(nzi,3) = values(nzi);
  end
end

[pointers_out, columns_out, values_out] = sparse_compress(n, m, triplets);
```

Problem 6: Make it run in Matlab/Octave/Scipy/etc.

In this problem, you'll just have to run three problems on matlab. The first one will be to use the Jacobi method to solve a linear system. The second will be to use a Krylov method to solve a linear system. The third will be to use ARPACK to compute eigenvalues on Matlab.

For this problem, you'll need to use the 'minnesota' road network. It's available on the website: http://www.cs.purdue.edu/homes/dgleich/nmcomp/matlab/minnesota.mat The file is in Matlab format. If you need another format, let me know.

a) Use the `gplot` function in Matlab to draw a picture of the Minnesota road network.

Solution

```
load minnesota
gplot(A,xy)
```

b) Check that the adjacency matrix A has only non-zero values of 1 and that it is symmetric. Fix any problems you encouter.

Solution

```
all((nonzeros(A)) == 1)
A = spones(A);
all((nonzeros(A)) == 1)
nnz(A-A')
```

c) We'll do some work with this graph and the linear system described in class:

$$\boldsymbol{I} - \gamma \boldsymbol{L}$$

where $\boldsymbol{L}$ is the combinatorial Laplacian matrix.

```
% In Matlab code
L = diag(sum(A)) - A;
S = speye(n) - gamma*L;
```

For the right-hand side, label all the points above latitude line 47 with 1, and all points below latitude line 44 with -1.

```
% In Matlab code
b = zeros(n,1);
b(xy(:,2) > 47) = 1;
b(xy(:,2) < 44) = -1;
```

Write a routine to solve the linear system using the Jacobi method on the compressed sparse row arrays. You should use your code from 5a to get these arrays by calling

```
[src,dst,val] = find(S);
T = [src,dst,val];
[pointers,columns,values] = sparse_compress(size(A,1), size(A,2), T);
```

Show the convergence, in the relative residual metric:

$$\|\mathbf{b} - \boldsymbol{A}\mathbf{x}^{(k)}\| / \|b\|$$

when `gamma␣=␣1/7` (Note that $\boldsymbol{A}$ is the matrix in the linear system, not the adjacency matrix.)

Show what happens when `gamma=1/5`

Solution (No plots here)

```
n = size(A,1);
L = diag(sum(A)) - A;
S = speye(n) - 1/7*L;
b = zeros(n,1);
b(xy(:,2) > 47) = 1;
b(xy(:,2) < 44) = -1;

[i j v] = find(S);
[pointers,columns,values] = sparse_compress(size(S,1), size(S,2),[i,j,v])
[x,resvec]=jacobi(pointers,columns,values,b);
semilogy(resvec);
```

Jacobi sketch

```
function [x,resvec] = jacobi(pointers,columns,values,b,tol,maxiter)
x = zeros(n,1);
for i=1:maxiter
  y = zeros(n,1);
  for row=1:length(b)
    yi = b(row); di = 0;
    for nzi=pointers(row):pointers(row+1)-1
      if columns(nzi) ~= row, yi = yi - values(nzi)*x(columns(nzi));
      else di=values(nzi);
      end
    end
    y(row) = yi/di;
  end
  % compute the residual
  r = zeros(n,1);
  for row=1:length(b)
    ri = b(row);
    for nzi=pointers(row):pointers(row+1)-1
      ri = ri - values(nzi)*y(columns(nzi));
    end
  end
  resvec(i)=norm(ri);
  if resvec(i) < tol, break; end
end
resvec = resvec(1:i);
if resvec(end) > tol, warning('did not converge'); end
```

d) Try using Conjugate Gradient `pcg` and `minres` in Matlab on this same system with `gamma=1/7` and `gamma=1/5`. Show the convergence of the residuals.

Solution Both work for gamma=1/7, neither work for gamma=1/5.

```
S = speye(n) - 1/5*L;
b = zeros(n,1);
b(xy(:,2) > 47) = 1;
b(xy(:,2) < 44) = -1;

%%
[x,flag,relres,iter,resvec] = pcg(S,b);
semilogy(resvec);

%%
[x,flag,relres,iter,resvec] = minres(S,b,1e-8,500);
semilogy(resvec);
```

The `semilogy` was how to show the convergence.

e) Use the `eigs` routine to find the 18 smallest eigenvalues of the Laplacian matrix $\boldsymbol{L}$.

```
>> [V,D] = eigs(L,18,'SA'); diag(D)

ans =

   -0.0000
    0.0000
    0.0008
    0.0021
    0.0023
```

0.0031
0.0051
0.0055
0.0068
0.0073
0.0100
0.0116
0.0123
0.0126
0.0134
0.0151
0.0165
0.0167