

CAAM 551 Problem set 1

Due Fri. 2 Sep. 2011.

Problem 1

Let A be an $n \times n$ symmetric positive definite matrix.

- a. Prove that MAM^T is symmetric and positive definite for any nonsingular matrix M of order n .

Note: MAM^T is called a congruence (transformation) of A .

- b. Let P be any permutation matrix of order n . Let

$$PAP^T = \begin{bmatrix} \alpha & a^T \\ a & \hat{A} \end{bmatrix}$$

Prove that $\alpha > 0$ and also that $A^{(2)} := \hat{A} - \frac{1}{\alpha}aa^T$ is symmetric and positive definite of order $n - 1$.

Problem 2

Suppose that A is an $n \times n$ symmetric positive definite matrix.

- a. Suppose that $A = LL^T$ is the cholesky factorization of A . For each $i = 1, 2, \dots, n$, prove that $|L(i, j)| \leq \sqrt{A(i, i)}$ for $1 \leq j \leq i$.
- b. Prove for all i, j that $|A(i, j)| \leq \frac{1}{2}(A(i, i) + A(j, j))$ and conclude that the maximum element of A (in abs. value) occurs on the diagonal of A .

Problem 3

Do problems 4 a,b,c,d page 98 of Saad (2nd ed.).

You can save some time by creating the matrices A and B in Matlab putting the numerical value 1 in place of *. Then you can do the matrix multiplications in Matlab. You may display the matlab results instead of writing them out by hand.

PROBLEM 4

Note: If you are a CAAM major, you MUST do problem 4b **instead** of problem 4a. If you are not a CAAM major, you may do problem 4a or problem 4b (DO NOT DO BOTH).

Problem 4a

- a. Write a Matlab code that will convert a sparse matrix in Matlab format $[i, j, A_{ij}]$ to CSR storage. You should store the pointers to row starts in an array IA , the column indices of the nonzero elements in an array JA and the nonzero matrix elements of A in an array AA .

For example the matrix of EX3.7 p 89 of Saad is:

```
% CSR Storage
%
% AA = [ 1.  2.  3.  4.  5.  6.  7.  8.  9. 10. 11. 12. ]
% JA = [ 1  4  1  2  4  1  3  4  5  3  4  5]
%      ^      ^      ^      ^      ^
% IA = [ 1      3      6      10     12    13]
```

- b. Write a Matlab code that will multiply a square matrix A in CSR storage times a vector x . i.e. code $w = Ax$.

- c. Repeat parts 1 and 2 using Jagged Diagonal Storage

For the Saad example this would be

```
% Jagged Diag Storage
%
% Ajag = [1.  2.  0.  0.  ;    J = [1 4  *  *  ;
%       3.  4.  5.  0.  ;      1 2  4  *  ;
%       6.  7.  8.  9.  ;      1 3  4  5  ;
%      10. 11.  0.  0.  ;      3 4  *  *  ;
%      12.  0.  0.  0.  ]      5 *  *  * ]
%
```

Hint: You might find the matlab command

```
[rows,cols,vals] = find(A(i,:));
```

to be useful.

You should write a test routine for each of the codes above and give computational evidence that they are working correctly.

- d. Write a Matlab code that will first solve $Lc = b$ and will then solve $L^T x = c$, where L is a sparse non-singular real $n \times n$ lower triangular matrix. (b is a given n -vector and x is solution vector).

The result x will be the solution to $Ax = b$ where $A = LL^T$. The matrix L should be traversed by rows in both cases. Thus, the first code (involving L) should be row oriented and the second code (involving L^T) should be column oriented.

Your Matlab function should be of the form

```
function [x] = CHsolve(AA,JA,IA,b);
```

where the lower triangular matrix L is stored in the AA, JA, IA CSR data structure. (You can use L in place of A in the notation if you wish.)

Write a test function to demonstrate your solver works correctly and test it on a random sparse lower triangular L (set the diagonals to 1 to assure non-singularity, but don't assume this in the code).

Problem 4b

The following function will create a struct in which you can create the CSR storage scheme and then just refer to a single entity instead of having to pass three arrays.

```
function [s] = csr(nz,m,n);
%
% This sets up a struct s to hold a matrix A in CSR format
%
    s = struct('nz',nz,'m',m,'n',n,'ia',[1:n+1],'ja',[1:nz],'aa',randn(nz,1));
%
% s has fields
%
%     s.nz    % length of arrays ja and aa
%     s.m     % number of rows
%     s.n     % number of columns
%     s.ia    % array to hold pointers to row starts
%     s.ja    % array to hold row indices
%     s.aa    % array to hold matrix values
```

Thus if you call

```
L = csr(nz,n,n)
```

you may fill in the values $L.ia \leftarrow IA$, $L.ja \leftarrow JA$, $L.aa \leftarrow AA$ and refer to the nonzero entries in i -th row of L as

```
for jj = L.ia(i): L.ia(i+1) - 1,
    j = L.ja(jj);
    value_ij = L.aa(jj);
end
```

Use this structure to write all of the the codes of Problem 4a above. Thus the function `CHsolve` will be re-written so that you can call it in the form

```
x = CHsolve(L,b)
```

and this should produce the solution to $LL^T x = b$.