Table of Contents

Homework #3 1
Problem #1 : Diagonal matrices and their inverse 1
Problem #2 : Manipulating matrices 2
Problem #3 : Toeplitz matrices 5
Problem #4 : Max, min, and logical indexing 6
Problem #5 : Timing a linear solve 7
Problem #6 : Heat Transfer 9
Problem #7 : PageRank 11
Problem #8 : Reducing fill-in 13

Homework #3

```
        Ian Blackstone, Helena Fujishin
        Math 365, Fall 2016
```

```
function HW3_FujishinBlackstone()

hmwk_problem(@prob1,'prob1');
hmwk_problem(@prob2,'prob2');
hmwk_problem(@prob3,'prob3');
hmwk_problem(@prob4,'prob4');
hmwk_problem(@prob5,'prob5');
hmwk_problem(@prob6,'prob6');
hmwk_problem(@prob7,'prob7');
hmwk_problem(@prob8,'prob8');

end

function hmwk_problem(prob,msg)
```

This function should be included in every assignment

```
try
    prob()
    fprintf('%s : Success!\n',msg);
catch me
    fprintf('%s : Something went wrong.\n',msg);
    fprintf('%s\n',me.message);
end
fprintf('\n');
end
```

Problem #1 : Diagonal matrices and their inverse

```
function prob1()
% make our matrix:
D = diag(2.^(1:5))
```

```
% One line to compute inverse
D_inv= diag(1./diag(D))

% If D_inv is the inverse of D then multiplying the two together
% should result in an identity matrix:
X = D_inv*D

end
```

```
D =

     2     0     0     0     0
     0     4     0     0     0
     0     0     8     0     0
     0     0     0    16     0
     0     0     0     0    32


D_inv =

    0.5000         0         0         0         0
         0    0.2500         0         0         0
         0         0    0.1250         0         0
         0         0         0    0.0625         0
         0         0         0         0    0.0313


X =

     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1

prob1 : Success!
```

Problem #2 : Manipulating matrices

```
function prob2()
% part a
% defining my matricies:
% our first matrix
A1 = tril(reshape(1:25, [5 5]),-1) %#ok<*NASGU>
% our second matrix
A2 = triu(repmat(1:1:5,[5 1]),2)
% our third matrix
A3 = triu(repmat(1:1:5,[5 1]),2)+tril(reshape(1:25, [5 5]),-1)

% part b
B1 = repmat(2:2:10,[5,1])
```

```
B2 = reshape(25:-1:1,[5,5])

B3 = B2 - diag(diag(B2))

B4 = B3 - eye(5)

% part c
C = repmat((1:5).^2,[5,1]);

C1 = (C - triu(C,1)).' + tril(C,-1)

C2 = C - tril(C,-1) - tril(C.')

C3 = C - tril(C) + tril(C.',-1) - diag(diag(C))

% part d

D = repmat(reshape((1:9).^2,[3,3]),[2,2])

end
```

```
A1 =

     0     0     0     0     0
     2     0     0     0     0
     3     8     0     0     0
     4     9    14     0     0
     5    10    15    20     0


A2 =

     0     0     3     4     5
     0     0     0     4     5
     0     0     0     0     5
     0     0     0     0     0
     0     0     0     0     0


A3 =

     0     0     3     4     5
     2     0     0     4     5
     3     8     0     0     5
     4     9    14     0     0
     5    10    15    20     0


B1 =

     2     4     6     8    10
     2     4     6     8    10
```

```
     2     4     6     8    10
     2     4     6     8    10
     2     4     6     8    10
```

```
B2 =

    25    20    15    10     5
    24    19    14     9     4
    23    18    13     8     3
    22    17    12     7     2
    21    16    11     6     1
```

```
B3 =

     0    20    15    10     5
    24     0    14     9     4
    23    18     0     8     3
    22    17    12     0     2
    21    16    11     6     0
```

```
B4 =

    -1    20    15    10     5
    24    -1    14     9     4
    23    18    -1     8     3
    22    17    12    -1     2
    21    16    11     6    -1
```

```
C1 =

     1     1     1     1     1
     1     4     4     4     4
     1     4     9     9     9
     1     4     9    16    16
     1     4     9    16    25
```

```
C2 =

     0     4     9    16    25
    -4     0     9    16    25
    -9    -9     0    16    25
   -16   -16   -16     0    25
   -25   -25   -25   -25     0
```

```
C3 =

    -1     4     9    16    25
     4    -4     9    16    25
```

```
     9     9    -9    16    25
    16    16    16   -16    25
    25    25    25    25   -25


D =

     1    16    49     1    16    49
     4    25    64     4    25    64
     9    36    81     9    36    81
     1    16    49     1    16    49
     4    25    64     4    25    64
     9    36    81     9    36    81

prob2 : Success!
```

Problem #3 : Toeplitz matrices

```
function prob3()
% part A: Use toeplitz
A = toeplitz([-2, 1, zeros(1,8)])
B = toeplitz([-2, 1, zeros(1,7) ,1])
% Part B: Use Toeplitz
A = toeplitz([1,zeros(1,8)],(1:8))

n=7;

r = (1:3:3*n).*((-1).^(0:n-1));
c = (1:2:2*n).*((-1).^(0:n-1));

B = toeplitz(c,r)
end


A =

    -2     1     0     0     0     0     0     0     0     0
     1    -2     1     0     0     0     0     0     0     0
     0     1    -2     1     0     0     0     0     0     0
     0     0     1    -2     1     0     0     0     0     0
     0     0     0     1    -2     1     0     0     0     0
     0     0     0     0     1    -2     1     0     0     0
     0     0     0     0     0     1    -2     1     0     0
     0     0     0     0     0     0     1    -2     1     0
     0     0     0     0     0     0     0     1    -2     1
     0     0     0     0     0     0     0     0     1    -2


B =

    -2     1     0     0     0     0     0     0     0     1
     1    -2     1     0     0     0     0     0     0     0
```

```
     0     1    -2     1     0     0     0     0     0     0
     0     0     1    -2     1     0     0     0     0     0
     0     0     0     1    -2     1     0     0     0     0
     0     0     0     0     1    -2     1     0     0     0
     0     0     0     0     0     1    -2     1     0     0
     0     0     0     0     0     0     1    -2     1     0
     0     0     0     0     0     0     0     1    -2     1
     1     0     0     0     0     0     0     0     1    -2
```

```
A =

     1     2     3     4     5     6     7     8
     0     1     2     3     4     5     6     7
     0     0     1     2     3     4     5     6
     0     0     0     1     2     3     4     5
     0     0     0     0     1     2     3     4
     0     0     0     0     0     1     2     3
     0     0     0     0     0     0     1     2
     0     0     0     0     0     0     0     1
     0     0     0     0     0     0     0     0
```

```
B =

     1    -4     7   -10    13   -16    19
    -3     1    -4     7   -10    13   -16
     5    -3     1    -4     7   -10    13
    -7     5    -3     1    -4     7   -10
     9    -7     5    -3     1    -4     7
   -11     9    -7     5    -3     1    -4
    13   -11     9    -7     5    -3     1
```

```
prob3 : Success!
```

Problem #4 : Max, min, and logical indexing

```
function prob4()
A = rand(9);
maxrowA = max(transpose(A)) % a)Gives us the maximum in every row
maxcolumnA = max(A) % b)Gives us the maximum in every column
overallmax = max(A(:)) % c) Gives us the overall maximum
A(A<0.5) = 0
end
```

```
maxrowA =

  Columns 1 through 7

    0.9459    0.7821    0.8685    0.7266    0.8993    0.9392    0.9660
```

```
  Columns 8 through 9

    0.9814    0.8531


maxcolumnA =

  Columns 1 through 7

    0.9814    0.8582    0.7821    0.9445    0.7883    0.9459    0.8993

  Columns 8 through 9

    0.9660    0.9205


overallmax =

    0.9814


A =

  Columns 1 through 7

         0         0         0    0.7884         0    0.9459    0.5863
    0.7380         0    0.7821    0.6958    0.6425    0.7772         0
    0.6934    0.5274    0.6974    0.8685    0.7883         0         0
         0    0.5412         0         0         0         0         0
    0.8005    0.6371         0         0    0.7041    0.5326    0.8993
    0.9392    0.8582    0.6032         0    0.6134    0.7744    0.7701
         0    0.7974         0         0         0         0         0
    0.9814         0         0    0.9445    0.5185    0.8617         0
    0.7703         0    0.7518         0         0    0.5641    0.8531

  Columns 8 through 9

    0.5484    0.8454
         0         0
         0         0
         0    0.7266
         0         0
    0.5411         0
    0.9660         0
    0.6242    0.9205
         0    0.6819

prob4 : Success!
```

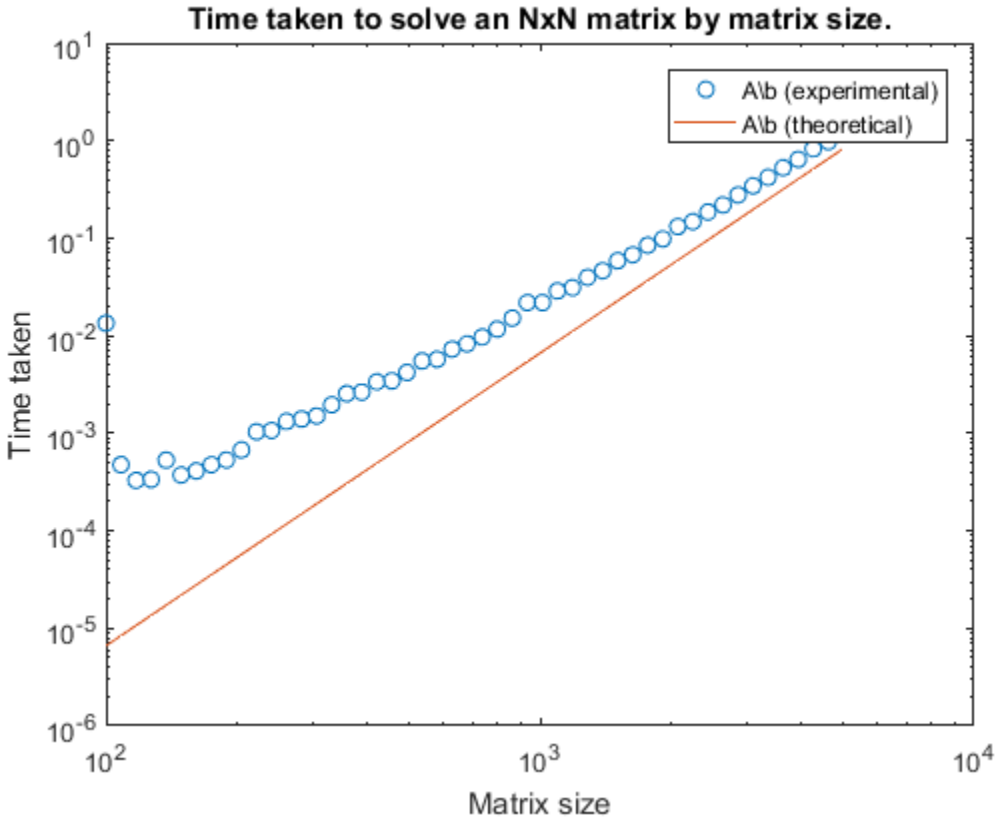
Problem #5 : Timing a linear solve

```
function prob5()
```

```
% Determine the maximum possible array that can fit in available
% memory.
Nmax = 5000;

% Create two random matrices, Nmax must be converted to a 16 bit
% integer for large memory systems. Larger memory system with more
% than 68 GB of memory available may need to convert to int32.
A = rand(int16(Nmax));
B = rand(int16(Nmax));

% Calculate the time taken to multiply the two random matrices.
tic
A*B;
t = toc;

% Calculate the number of operations performed in A*B and use the
% time it took to calculate that operation to find the floating
% point operations per second and then estimate the operating speed
% of the processor.
Nops = Nmax^2 * (Nmax-1) + Nmax^3;
Flops = Nops/t;
Hz = Flops/4;
GHz = Hz / 10e9;
fprintf('This computer operates at %.2f GHz \n',GHz)

% part c

% Initialize a list.
lutimes = zeros(50,1);

% create a list of logarithmically spaced points.
N = 2;
Nmaxlog = log10(Nmax);
Nvec = logspace(N,Nmaxlog);

% For each item in our list of points generate random matrices and a
% random vector and time how long it takes to solve each
% matrix/vector combination
for n = 1:numel(Nvec)
 A = rand(int16(Nvec(n)));
 b = rand(int16(Nvec(n)),1);
 tic
 A\b;
 lutimes(int16(n)) = toc;
end

% Generate a set of theoretical data points to be plotted based on the
% calculated speed of the processor.
Ttimes = 2/3 * Nvec.^3 / Flops;

% Plot both the theoretical line and the actual times over the matrix
% size on a log-log plot.
figure
```

```
loglog(Nvec,lutimes,'o',Nvec,Ttimes,'-')
title('Time taken to solve an NxN matrix by matrix size.')
xlabel('Matrix size')
ylabel('Time taken')
legend('A\\b (experimental)','A\\b (theoretical)')

% part e
% The fastest computer at this time is the Sunway TaihuLight, with
% more than 10 million cores and an operating capacity of 93
% petaflops. My computer operating at 99 gigaflops would have been
% the fastest computer in the world until late 1993 when it would
% have been overtaken by the Numerical Wind Tunnel.  It is likely
% that we will see exascale machines within 10 years given the pace
% that China has been on to reach this milestone.

end
```

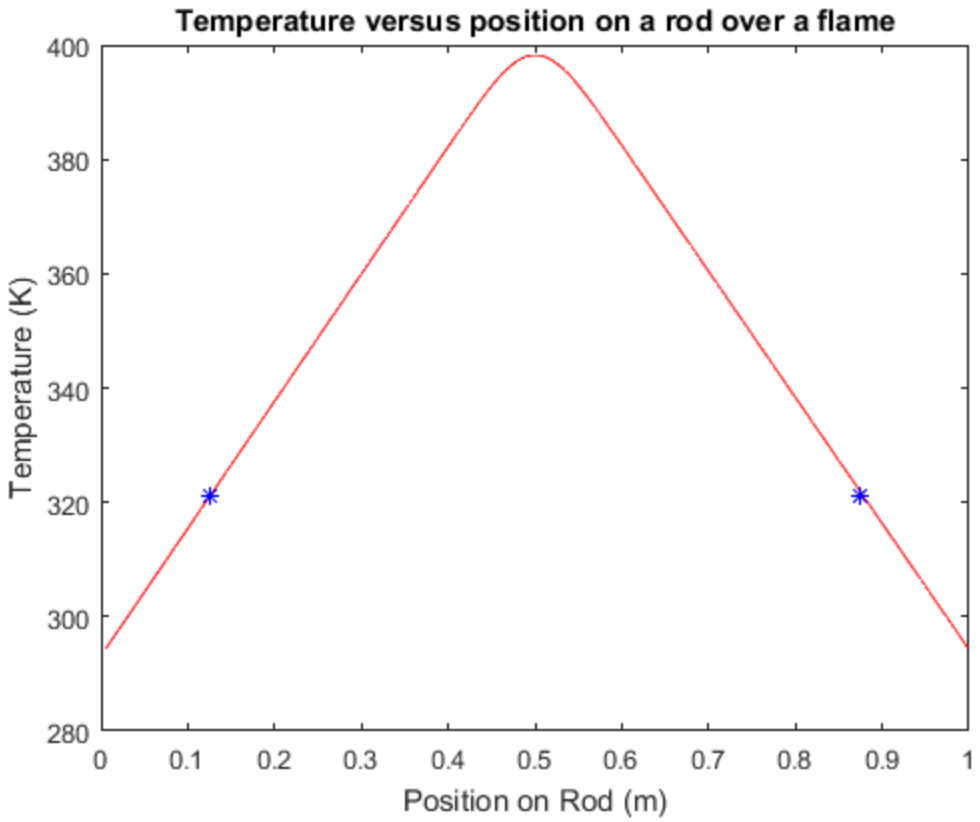
```
This computer operates at 2.52 GHz
prob5 : Success!
```



Problem #6 : Heat Transfer

```
function prob6()
N = 200;
L = 1;
```

```
B = 10^(-3);
S = 1*10^4;
Cp = 200;
rho = 10;
eps = 5*10^(-2);
T0 = 293.15;
f = @(x) (S/(2*rho*Cp*B))*(exp(-(((x-(L/2))/(eps)).^2)));
h = L/N;
A1 = 2*diag(ones(N,1));
A2 = (-1)*diag(ones(N-1,1),-1);
A3 = (-1)*diag(ones(N-1,1),1);
A = A1+A2+A3;
j = (1:N).';
x = h.*j;
B = 2*h^2.*f(x);
B(1)= B(1)+T0;
B(end)= B(end)+T0;
T = A\B;

% part c
figure
plot(x,T, 'r-')
title('Temperature versus position on a rod over a flame')
xlabel('Position on Rod (m)')
ylabel('Temperature (K)')
% part d (too hot is about 320.9K)
jouch = find(T > 320.9,1, 'first') %this should give us where j=25
fprintf('You can touch the rod %f m away from the center\n',0.5-
x(jouch))
hold on % I'm going to mark where you can touch the rod with a blue
 star
plot(x(jouch),T(jouch),'b*')
plot(1-x(jouch),T(jouch),'b*')
end


jouch =

    25

You can touch the rod 0.375000 m away from the center
prob6 : Success!
```



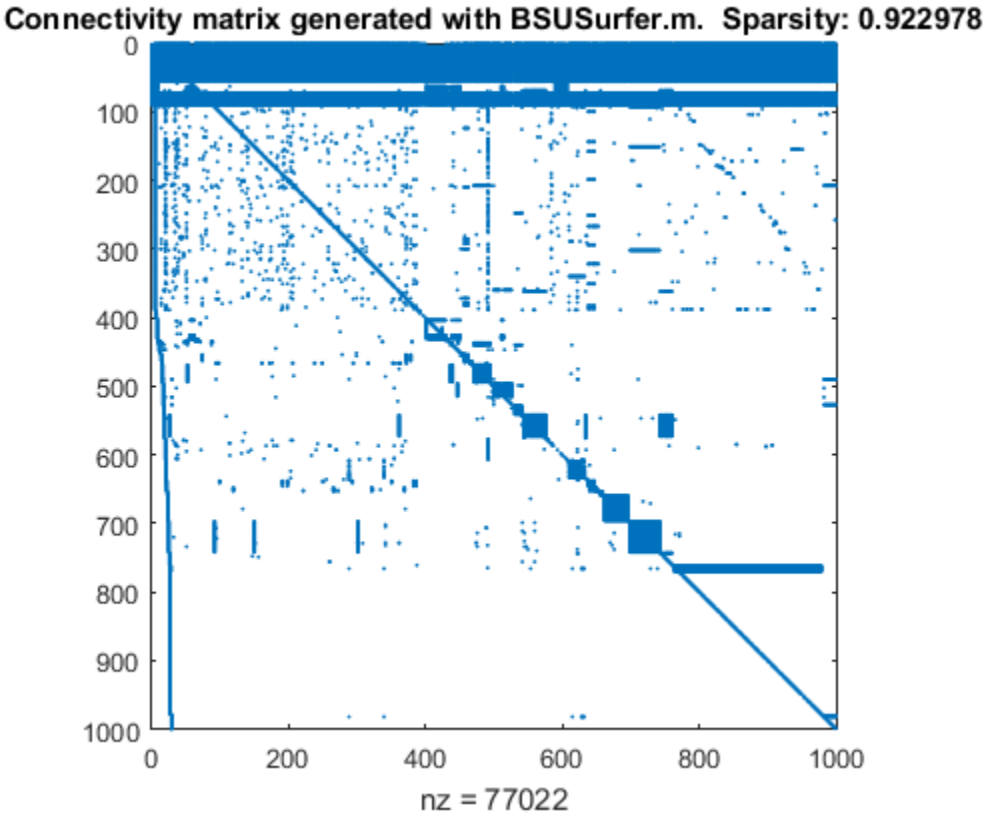
Problem #7 : PageRank

```
function prob7()
n= 1000;
% Load the results of BSUSurfer.m.
load BSUSurferResults;

% find the size of the matrix G.
[row,col] = size(G);

% Determine the sparsity of matrix G.
spar = 1 - nnz(G)/(row * col);

% Generate the connectivity matrix with title.
figure
spy(G)
title(sprintf('Connectivity matrix generated with BSUSurfer.m.
 Sparsity: %f',spar))

% The following code was added to the end of pagerank.m to output the
% bottom 10 pages as well as the top 10

% [~,id] = sort(x,1,'ascend');
% fprintf('\n#     PageRank     Page\n');
% for j=1:10
```

```
%    fprintf('%02d    %1.2e     %s\n',n - j + 1,x(id(j)),U{id(j)});
% end

% run pagerank twice, with two different weightings.
pagerank(U,G,0.85);

pagerank(U,G,0.95);

% The top rated sites are mostly unchanged between the two weightings
% with only two sites swapping positions but the bottom ten are almost
% completely different, with only index.boisestate.edu/feed remaining
% in the lowest ranked pages in both weightings.

end
```

```
#     PageRank     Page
01    1.45e-02     http://www.boisestate.edu
02    1.40e-02     http://go.boisestate.edu/privacy
03    1.06e-02     http://news.boisestate.edu/social
04    1.05e-02     http://alumni.boisestate.edu
05    1.02e-02     http://library.boisestate.edu
06    1.02e-02     http://admissions.boisestate.edu
07    1.01e-02     http://graduatecollege.boisestate.edu
08    1.00e-02     http://ecampus.boisestate.edu
09    1.00e-02     http://cid.boisestate.edu
10    9.96e-03     http://research.boisestate.edu

#     PageRank     Page
1000    2.20e-04     http://camps.boisestate.edu/athletics
999    2.20e-04     http://english.boisestate.edu/techcomm
998    2.20e-04     http://index.boisestate.edu/ttps:/
www.facebook.com/boisestatetechcomm
997    2.20e-04     http://communication.boisestate.edu/utp
996    2.20e-04     http://varsityb.boisestate.edu
995    2.20e-04     http://venturecollege.boisestate.edu
994    2.20e-04     http://index.boisestate.edu/www.facebook.com/
venturecollegeboisestateuniverstiy
993    2.20e-04     http://go.boisestate.edu/closure
992    2.20e-04     http://healthservices.boisestate.edu/services/
wellness
991    2.20e-04     http://index.boisestate.edu/feed

#     PageRank     Page
01    1.66e-02     http://www.boisestate.edu
02    1.59e-02     http://go.boisestate.edu/privacy
03    1.17e-02     http://news.boisestate.edu/social
04    1.16e-02     http://alumni.boisestate.edu
05    1.12e-02     http://admissions.boisestate.edu
06    1.11e-02     http://library.boisestate.edu
07    1.11e-02     http://graduatecollege.boisestate.edu
08    1.10e-02     http://ecampus.boisestate.edu
09    1.10e-02     http://cid.boisestate.edu
10    1.10e-02     http://research.boisestate.edu
```

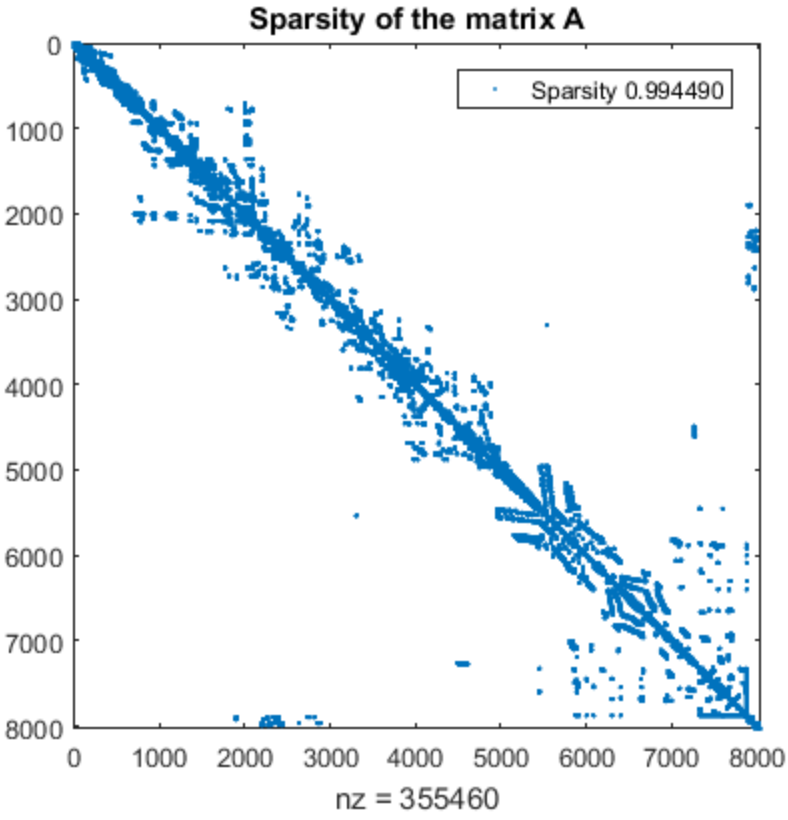
```
#      PageRank      Page
1000     1.31e-04      http://index.boisestate.edu/feed
999     1.31e-04     http://index.boisestate.edu/comments/feed
998     1.31e-04     http://accessibility.boisestate.edu
997     1.31e-04     http://assessment.boisestate.edu
996     1.31e-04     http://orgs.boisestate.edu/ace
995     1.31e-04     http://orgs.boisestate.edu/bsuaop
994     1.31e-04     http://preco.boisestate.edu
993     1.31e-04     http://orgs.boisestate.edu/bicyclecongress
992     1.31e-04     http://blackboardhelp.boisestate.edu
991     1.31e-04     http://thunder.boisestate.edu
prob7 : Success!
```

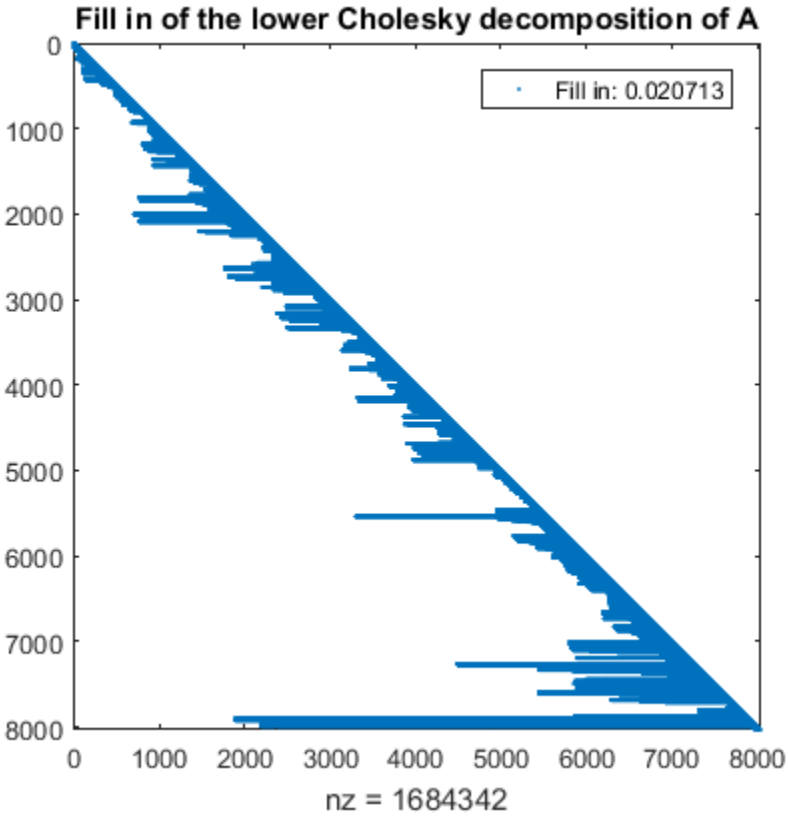


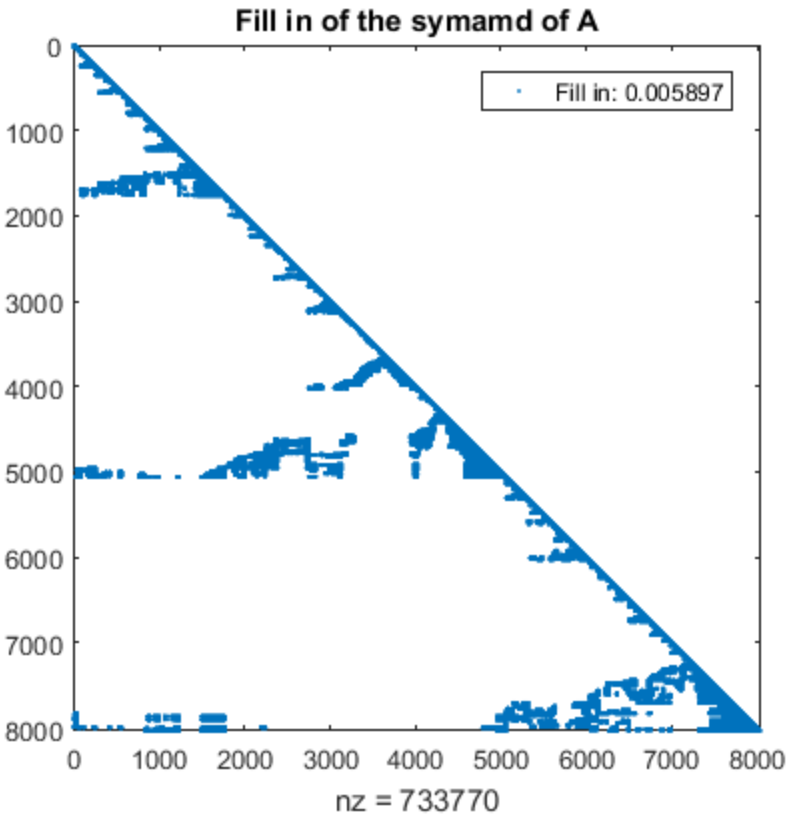
Problem #8 : Reducing fill-in

```
function prob8()

% Load the data from file, storing it as matrix A.
load('bcsstk38.mat')
A = Problem.A;

% Determine the size of matrix A.
[row,col] = size(A);
```

```
% Calculate the sparsity of A.
spar1 = 1 - nnz(A)/(row*col);

% Create the connectivity matrix.
figure
spy(A)
title('Sparsity of the matrix A')
legend(sprintf('Sparsity %f',spar1))

% Create the Cholesky decomposition of matrix A.
B = chol(A,'lower');

% Calculate the sparsity of B.
spar2 = 1 - nnz(B)/(row*col);

% Compare the sparsity of A and B.
compspar = spar2/spar1;

% Plot the connetivity matrix of B and display the fill in.
figure
spy(B)
title('Fill in of the lower Cholesky decomposition of A')
legend(sprintf('Fill in: %f',1 - compspar))

% Use the symand function and generate the Cholesky decomposition of
% the rearranged A to reduce fill in.
c = symamd(A);
C = chol(A(c,c),'lower');

% Calculate the sparsity of C and compare it to the sparsity of A.
spar3 = 1 - nnz(C)/(row*col);
compspar2 = spar3/spar1;

% Plot the connectivity matrix of C and display the fill in.
figure
spy(C)
title('Fill in of the symamd of A')
legend(sprintf('Fill in: %f',1 - compspar2))

end

prob8 : Success!
```


Sparsity of the matrix A
Sparsity 0.994490
0
1000
2000
3000
4000
5000
6000
7000
8000
nz = 355460


Fill in of the lower Cholesky decomposition of A
Fill in: 0.020713
0
1000
2000
3000
4000
5000
6000
7000
8000
nz = 1684342



Published with MATLAB® R2016a