Computing Assignment 2: GE Timing Test

By Kai Sackville-Hii Jan 28, 2019

ca2_demo.m -- timing exercise (macm316, hl -- 13 jan 2019)

Purpose: This script serves as a demo for students to build on in completing computing assignment 2. This script builds three types of NxN matricies: dense, upper triangular, and permuted upper triangular. It then performs a matrix solve with each, Nex number of times. The time it takes for Nex number of solves is used to estimate the time of one solve .

Instructions: Start by running the script once, and see the output for the esimtated times. Note: the choice of Nex here may not give accurate results for all three matrix types. Next, copy and paste this code into your own Matlab file. Follow the assignment sheet for further instructions to complete your report.

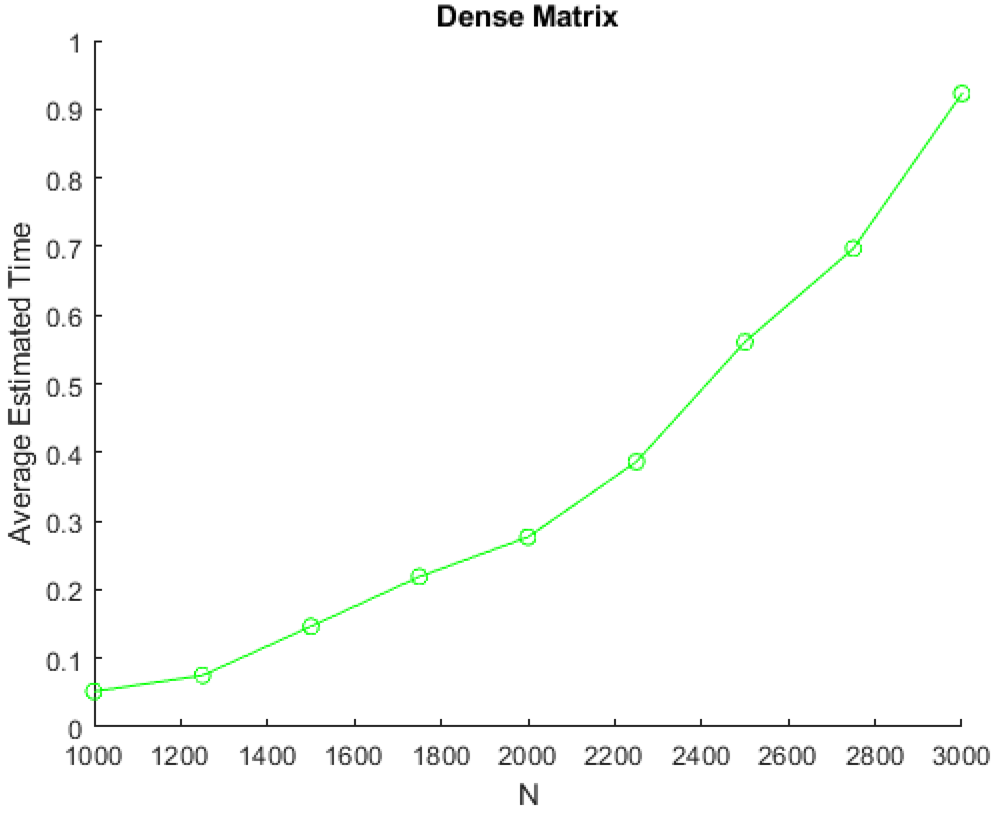
```
clear;

%  experimental parameters
% N   = 1000;
Nex = 0;

% NArr = [10 ^3, 11^3, 12^3, 13^3, 14^3, 15^3, 16^3];
% NArr = [35^2, 36^2, 37^2, 38^2, 39^2 40^2];
NArr = 1000:250:3000;

Nex_dense_func = @(n) -0.025*n + 100;
Nex_tri_func = @(n) -0.025*n + 1000;
Nex_perm_func = @(n) -0.025*n + 300;
Nex_tridiag_func = @(n) -0.025*n + 200;
Nex_sparse_func = @(n) -0.025*n + 10000;

tri_time_array = zeros(1,length(NArr));
perm_time_array = zeros(1,length(NArr));
dense_time_array = zeros(1,length(NArr));
tridiag_time_array = zeros(1,length(NArr));
sparse_time_array = zeros(1,length(NArr));


avg_tri_time_array = zeros(1,length(NArr));
avg_perm_time_array = zeros(1,length(NArr));
avg_dense_time_array = zeros(1,length(NArr));
avg_tridiag_time_array = zeros(1,length(NArr));
avg_sparse_time_array = zeros(1,length(NArr));

for iter = 1:length(NArr)

    fprintf("\nN = %f", NArr(iter));

    N = NArr(iter);

    Nex_dense = Nex_dense_func(N);
    Nex_tri = Nex_tri_func(N);
    Nex_perm = Nex_perm_func(N);
    Nex_tridiag = Nex_tridiag_func(N);
    Nex_sparse = Nex_sparse_func(N);

    % ----- Initilize matrices ----- %

    %  dense matrix (no zeros)
    Md = randn(N,N);

    %  upper triangular
    Mt = triu(Md);

    %  randomly row-exchanged upper triangular (these are tricky array
 commands,
    %  but if you run a small sample, it is clear they do the right
 thing)
    idx=randperm(N);
    Mp = Mt(idx,:);

    %Tri-diagonal and sparse tri-diagonal
    M3 = diag(diag(Md))+diag(diag(Md,-1),-1)+diag(diag(Md,1),1);
    M3s = sparse(M3);

    %  exact solution of all ones
    x = ones(N,1);

    %  right-side vectors
    bd = Md*x;
    bt = Mt*x;
    bp = bt(idx);
    b3 = M3*x;
    b3s = M3s*x;

    % ----- Solving Matrices ----- %

    %  dense test
    tic
    for jj = 1:Nex_dense
        xd = Md\bd;
    end
    dense_time=toc;
    dense_time_array(iter)=toc;

    %  upper tri test
    tic
    for jj = 1:Nex_tri
        xt = Mt\bt;
    end
    tri_time=toc;
    tri_time_array(iter)=toc;

    % permuted upper tri test
    tic
    for jj = 1:Nex_perm
```

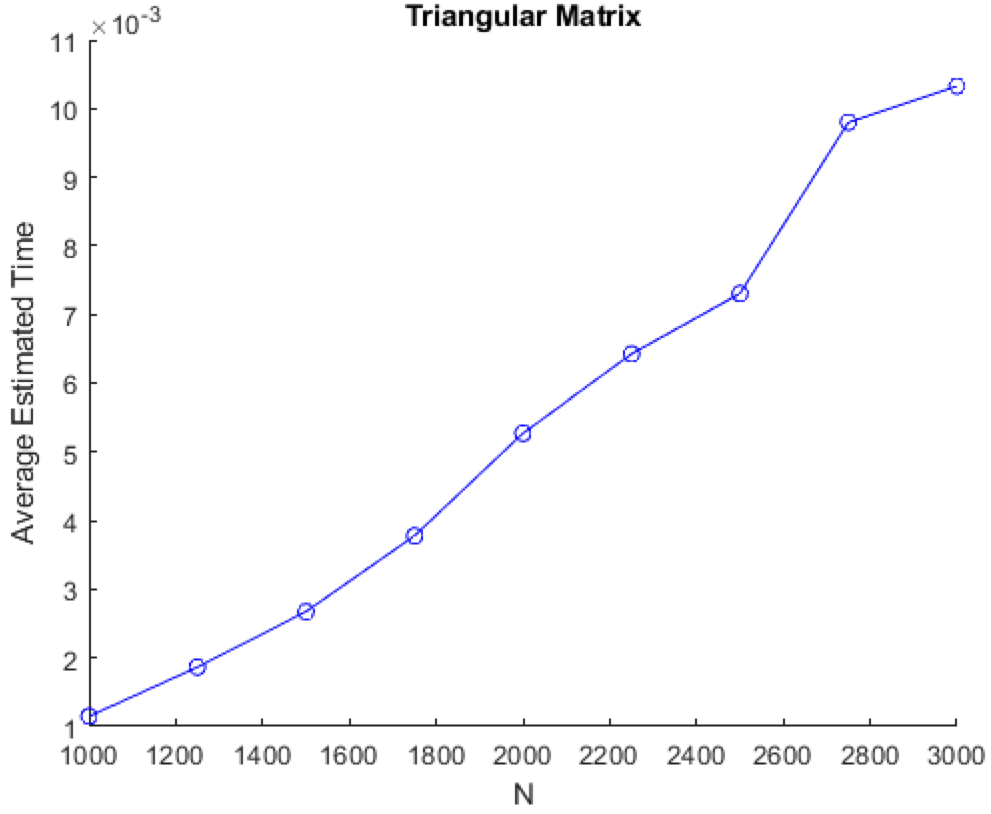
```
        xp = Mp\bp;
    end
    perm_tri_time=toc;
    perm_time_array(iter)=toc;

    % tridiagonal matrix
    tic
    for jj = 1:Nex_tridiag
        xtd = M3\b3;
    end
    tridiag_time=toc;
    tridiag_time_array(iter)=toc;

    % sparse matrix
    tic
    for jj = 1:Nex_sparse
        xs = M3s\b3s;
    end
    sparse_time=toc;
    sparse_time_array(iter)=toc;

    % ----- Computing avgerage solve times ----- %
    avg_tri_time = tri_time/(Nex_tri);
    avg_perm_time = perm_tri_time/(Nex_perm);
    avg_dense_time = dense_time/(Nex_dense);
    avg_tridiag_time = tridiag_time/(Nex_tridiag);
    avg_sparse_time = sparse_time/(Nex_sparse);

    % ----- Add estimation to array -----
    avg_tri_time_array(iter) = avg_tri_time;
    avg_perm_time_array(iter) = avg_perm_time;
    avg_dense_time_array(iter) = avg_dense_time;
    avg_tridiag_time_array(iter) = avg_tridiag_time;
    avg_sparse_time_array(iter) = avg_sparse_time;

    % ----- Display results ----- %
    % You may find the following code helpful for displaying the
 results
    % of this demo.
    type_times = {
        'Dense',avg_dense_time ...
        'Upper Triangular', avg_tri_time ...
        'Permuted Upper Triangular', avg_perm_time ...
        'Tri-diagonal' , avg_tridiag_time ...
        'Sparse', avg_sparse_time ...
        };
    fprintf(' \n')
    fprintf('Estimated time for a %s matrix is %f seconds.
 \n',type_times{:})

end

% ----- Average Time Plots ----- %
const_x_label = 'N';
```

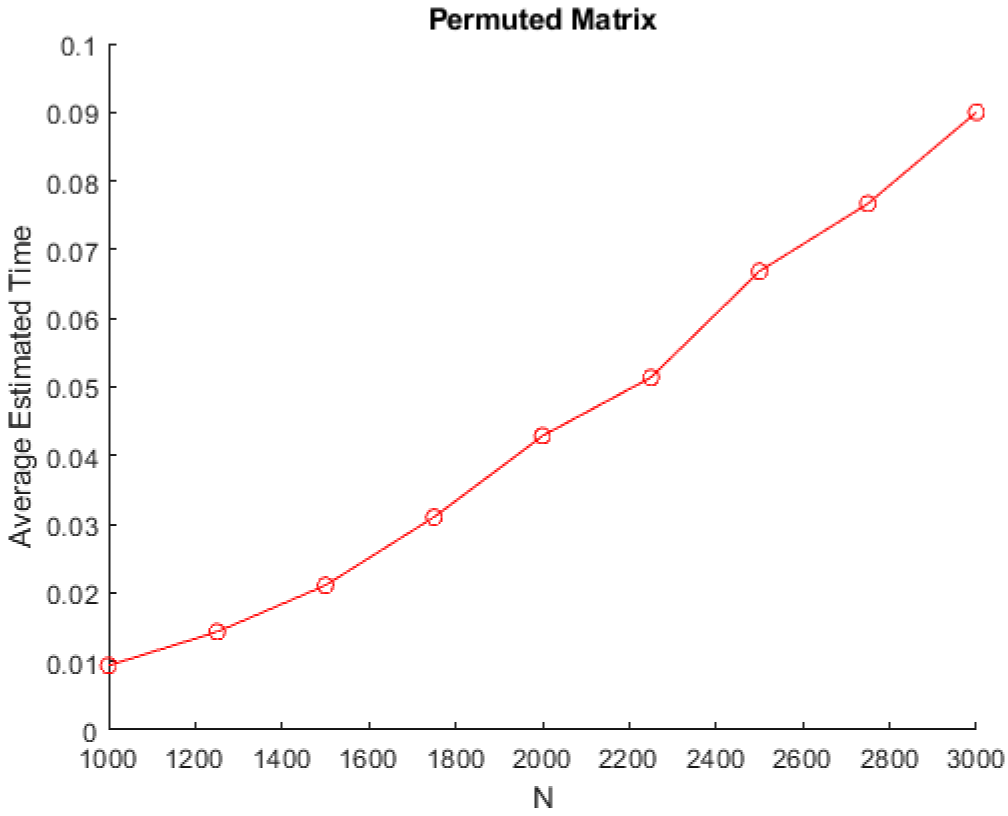
```
const_y_label = 'Average Estimated Time';

figure
hold on
title('Dense Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, avg_dense_time_array, 'g-o')
hold off

figure
hold on
title('Triangular Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, avg_tri_time_array, 'b-o')
hold off

figure
hold on
title('Permuted Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, avg_perm_time_array, 'r-o')
hold off

figure
hold on
title('Tri-diagonal Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, avg_tridiag_time_array, 'y-o')
hold off

figure
hold on
title('Sparse Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, avg_sparse_time_array, 'c-o')
hold off

% ----- Total Time Plots ----- %
const_x_label = 'N';
const_y_label = 'Total Experiment Time';

figure
hold on
title('Dense Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, dense_time_array, 'g-o')
hold off
```

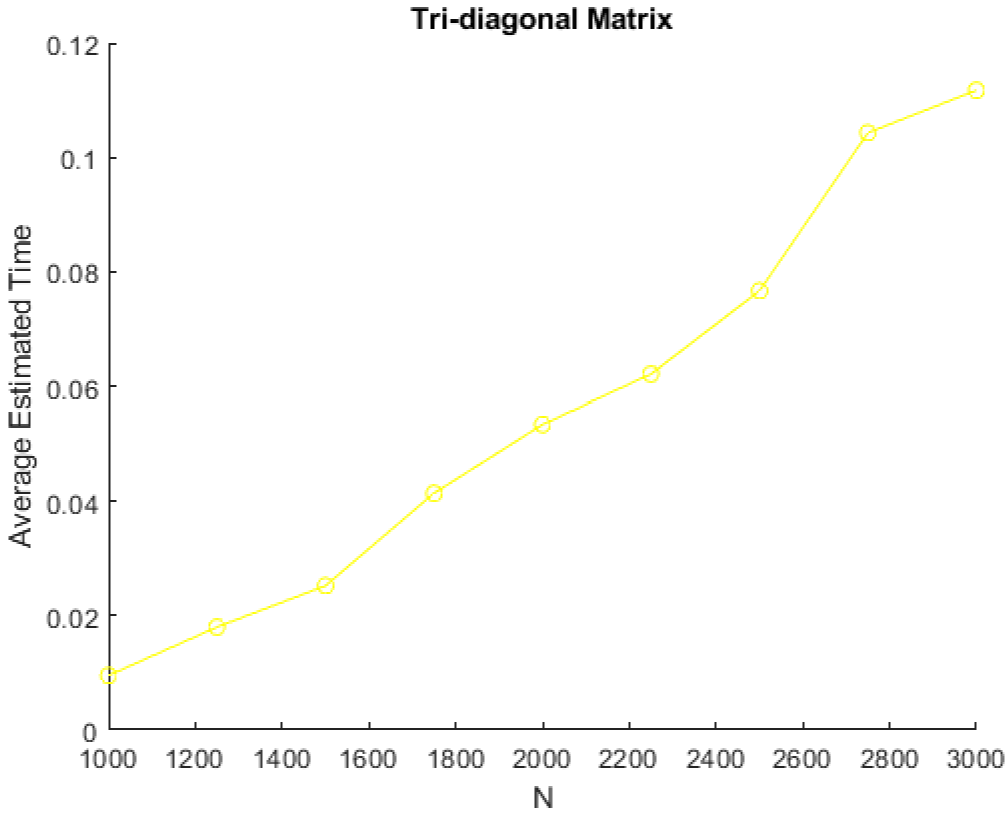
```
figure
hold on
title('Triangular Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, tri_time_array, 'b-o')
hold off

figure
hold on
title('Permuted Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, perm_time_array, 'r-o')
hold off

figure
hold on
title('Tri-diagonal Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, tridiag_time_array, 'y-o')
hold off

figure
hold on
title('Sparse Matrix')
xlabel(const_x_label)
ylabel(const_y_label)
plot(NArr, sparse_time_array, 'c-o')
hold off

save('times.mat', 'avg_dense_time_array', 'avg_tri_time_array', 'avg_perm_time_arr
```

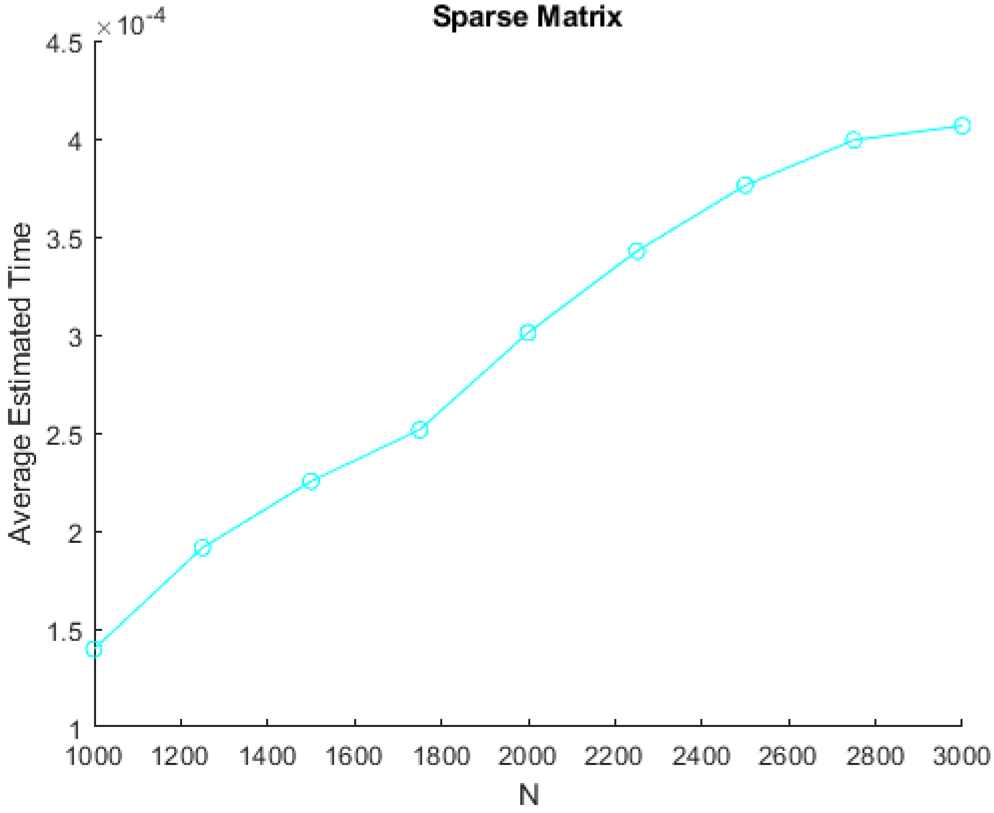
```
N = 1000.000000
Estimated time for a Dense matrix is 0.051673 seconds.
Estimated time for a Upper Triangular matrix is 0.001148 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.009400
 seconds.
Estimated time for a Tri-diagonal matrix is 0.009531 seconds.
Estimated time for a Sparse matrix is 0.000140 seconds.

N = 1250.000000
Estimated time for a Dense matrix is 0.075035 seconds.
Estimated time for a Upper Triangular matrix is 0.001865 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.014309
 seconds.
Estimated time for a Tri-diagonal matrix is 0.018007 seconds.
Estimated time for a Sparse matrix is 0.000191 seconds.

N = 1500.000000
Estimated time for a Dense matrix is 0.146504 seconds.
Estimated time for a Upper Triangular matrix is 0.002674 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.021093
 seconds.
Estimated time for a Tri-diagonal matrix is 0.025232 seconds.
Estimated time for a Sparse matrix is 0.000225 seconds.

N = 1750.000000
Estimated time for a Dense matrix is 0.218892 seconds.
Estimated time for a Upper Triangular matrix is 0.003779 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.031036
 seconds.
Estimated time for a Tri-diagonal matrix is 0.041415 seconds.
Estimated time for a Sparse matrix is 0.000252 seconds.

N = 2000.000000
Estimated time for a Dense matrix is 0.276628 seconds.
Estimated time for a Upper Triangular matrix is 0.005274 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.042879
 seconds.
Estimated time for a Tri-diagonal matrix is 0.053420 seconds.
Estimated time for a Sparse matrix is 0.000301 seconds.

N = 2250.000000
Estimated time for a Dense matrix is 0.386335 seconds.
Estimated time for a Upper Triangular matrix is 0.006423 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.051390
 seconds.
Estimated time for a Tri-diagonal matrix is 0.062099 seconds.
Estimated time for a Sparse matrix is 0.000343 seconds.

N = 2500.000000
Estimated time for a Dense matrix is 0.561016 seconds.
Estimated time for a Upper Triangular matrix is 0.007305 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.066854
 seconds.
Estimated time for a Tri-diagonal matrix is 0.076640 seconds.
Estimated time for a Sparse matrix is 0.000376 seconds.

N = 2750.000000
Estimated time for a Dense matrix is 0.697016 seconds.
Estimated time for a Upper Triangular matrix is 0.009801 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.076700
 seconds.
Estimated time for a Tri-diagonal matrix is 0.104308 seconds.
Estimated time for a Sparse matrix is 0.000399 seconds.

N = 3000.000000
Estimated time for a Dense matrix is 0.922737 seconds.
Estimated time for a Upper Triangular matrix is 0.010329 seconds.
Estimated time for a Permuted Upper Triangular matrix is 0.090004
 seconds.
Estimated time for a Tri-diagonal matrix is 0.111733 seconds.
Estimated time for a Sparse matrix is 0.000407 seconds.
```

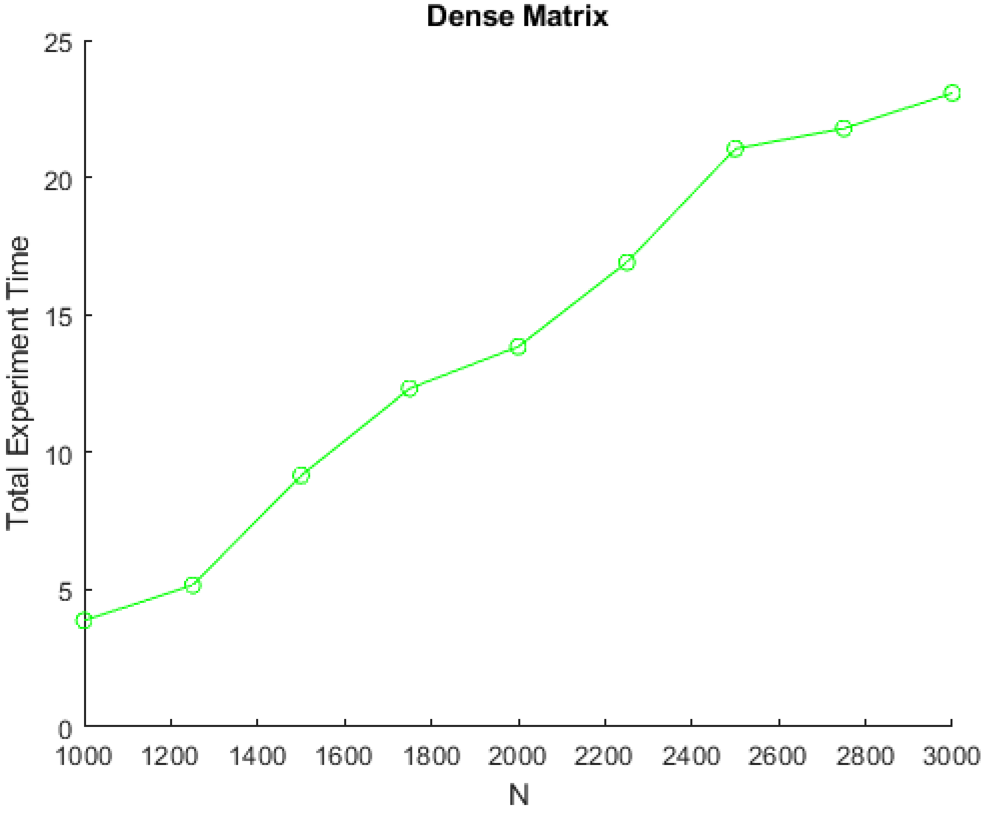

Dense Matrix
Average Estimated Time
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000
N

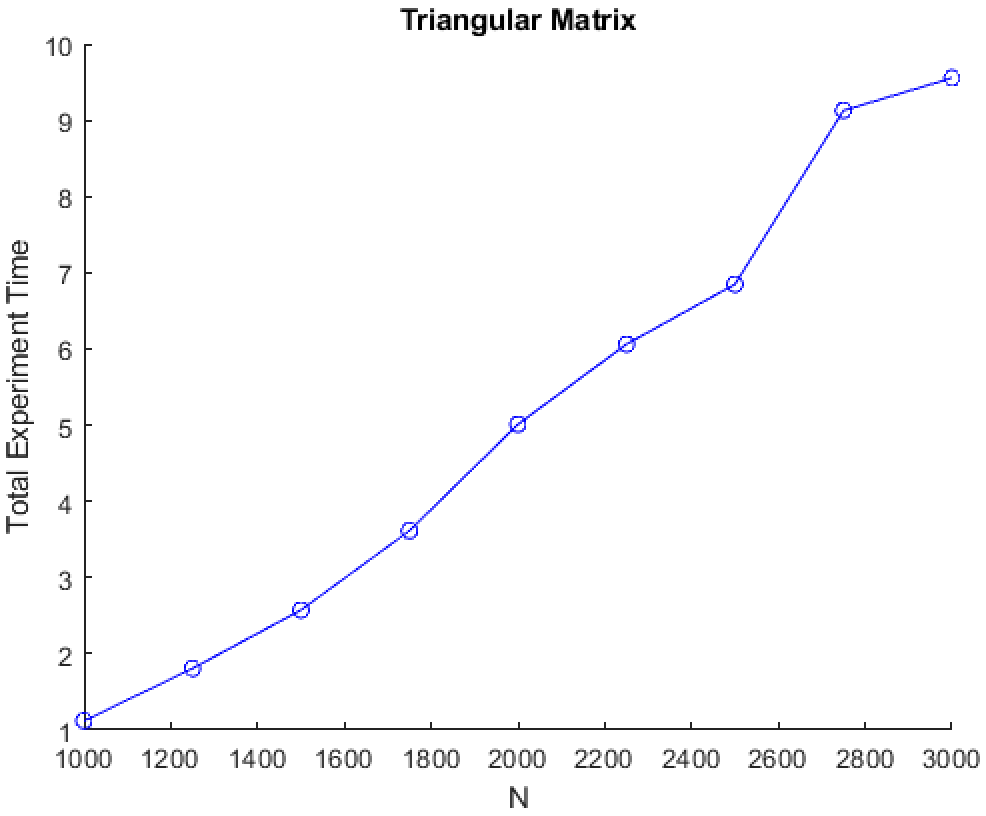

Triangular Matrix
× 10^-3
Average Estimated Time
1
2
3
4
5
6
7
8
9
10
11
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000
N

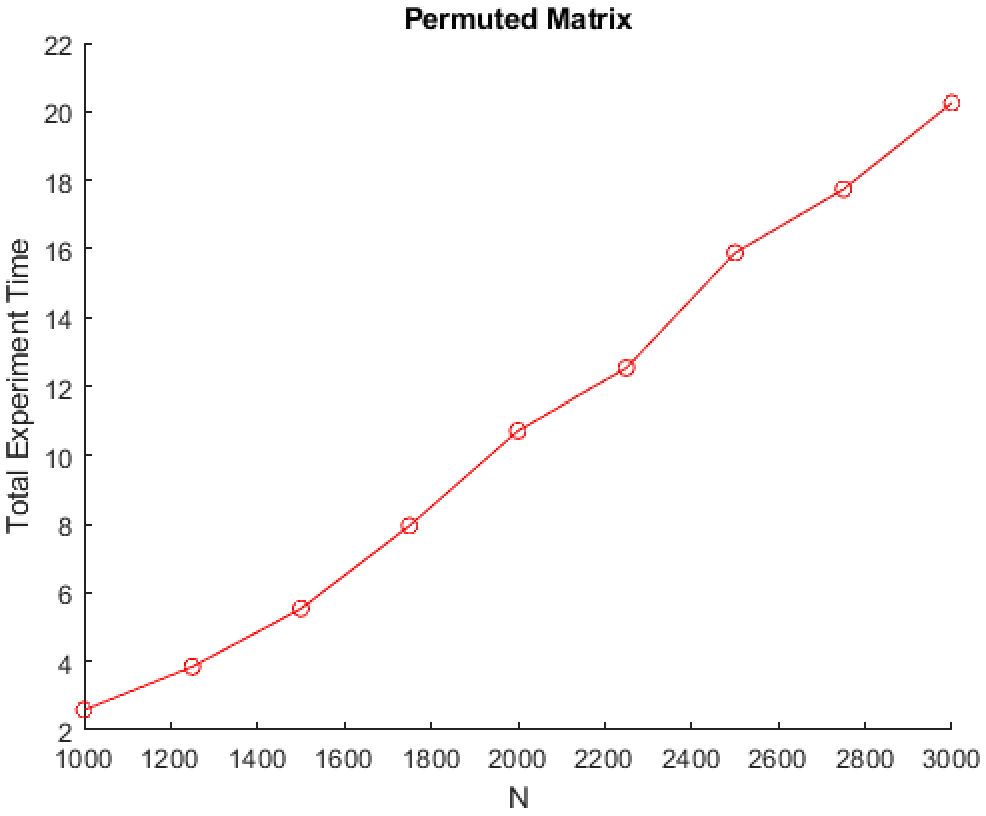

Permuted Matrix
Average Estimated Time
0.1
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000
N

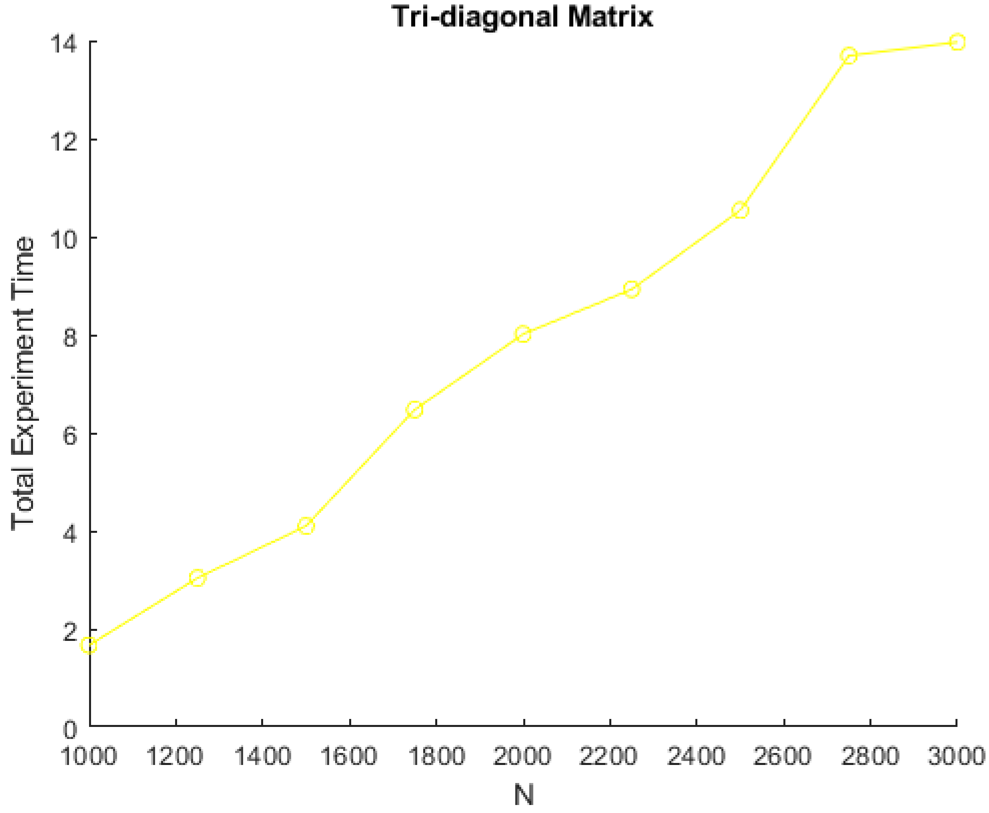

Tri-diagonal Matrix
Average Estimated Time
0.12
0.1
0.08
0.06
0.04
0.02
0
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000
N

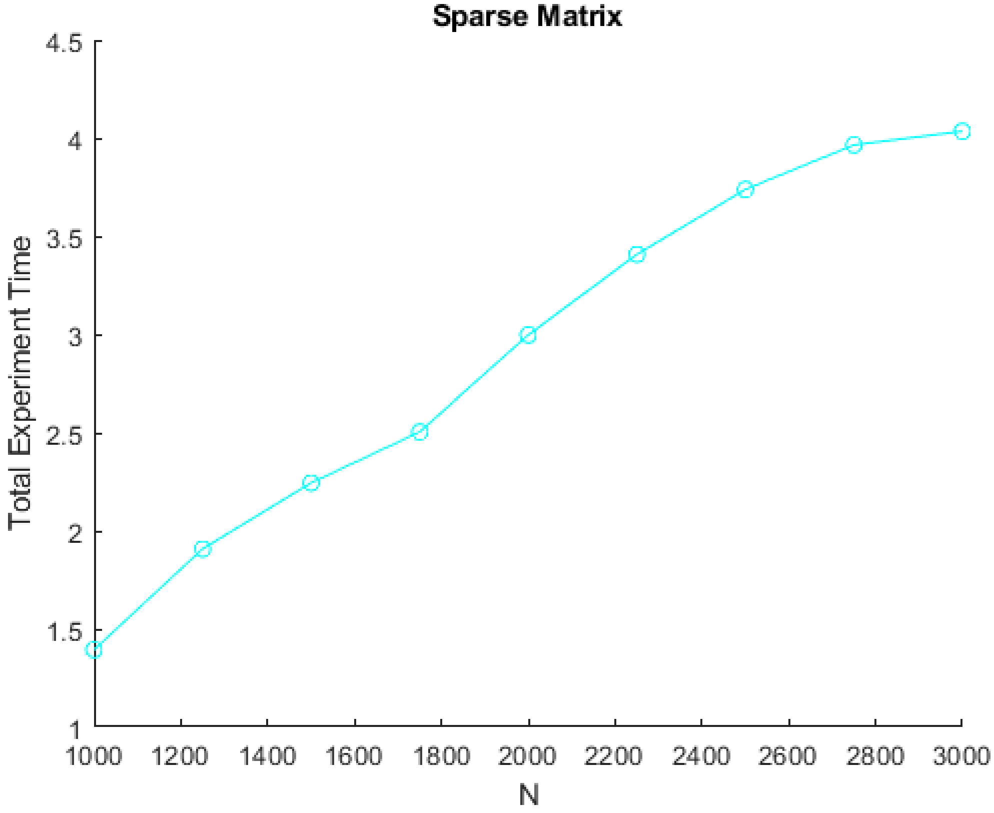

Sparse Matrix
$\times 10^{-4}$
4.5
4
3.5
3
2.5
2
1.5
1
Average Estimated Time
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000
N


Dense Matrix
25
20
15
10
5
0
Total Experiment Time
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000
N


Triangular Matrix


Permuted Matrix


Tri-diagonal Matrix
Total Experiment Time
N
0
2
4
6
8
10
12
14
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000


Sparse Matrix
Total Experiment Time
N
1
1.5
2
2.5
3
3.5
4
4.5
1000
1200
1400
1600
1800
2000
2200
2400
2600
2800
3000

Published with MATLAB® R2018b