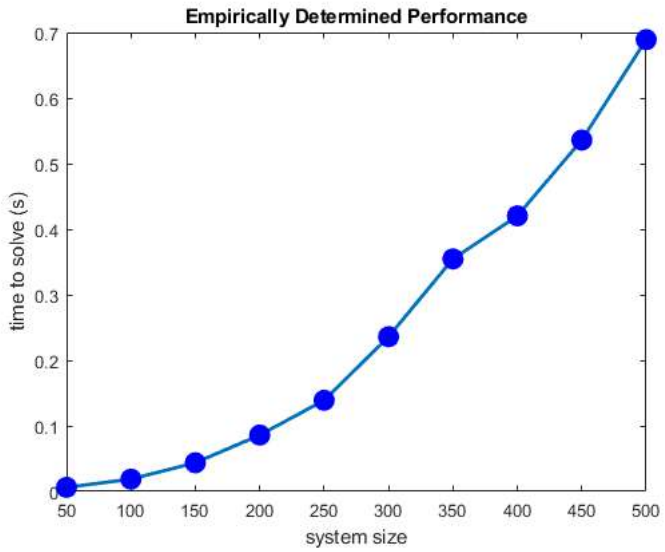
Contents


```
% Evaluate performance and scaling of Gaussian elimination, Jacobi
% iteration, and Thomas algorithm by solving systems of different
% size and timing the solves

clear
clc
close all

nvals=50:50:500;
testtimes=zeros(size(nvals));
lrep=10;     %how many times to repeat each test
```

Gaussian elimination

```
disp('Start of tests of Gaussian-elimination scaling');
for in=1:numel(nvals)
    nlarge=nvals(in);
    Blarge=diag(-1*ones(nlarge-1,1),-1)+diag(-1*ones(nlarge-1,1),1)+diag(4*ones(nlarge,1),0);    %this must be diagonally dominant or else the method won't converge
    blarge=ones(nlarge,1);

    for irep=1:lrep     %benchmark will repeat the same solution several times to eliminate random variations from CPU load, etc.
        tstart=cputime;
        [Blargemod,ordlarge]=Gauss_elim(Blarge,blarge);
        xlarge=backsub(Blargemod(ordlarge,:));
        tend=cputime;
        testtimes(in)=testtimes(in)+(tend-tstart)/lrep;
    end %for
    disp([' GE solution for system of size ',num2str(nlarge),' takes average time ',num2str(testtimes(in)),' s']);
end %for

TT(:,1) = testtimes';

figure(1);
plot(nvals,testtimes,'-o','LineWidth',2,'MarkerSize',10,'MarkerEdgeColor','blue','MarkerFaceColor','blue')
xlabel('system size');
ylabel('time to solve (s)');
title('Empirically Determined Performance');
```

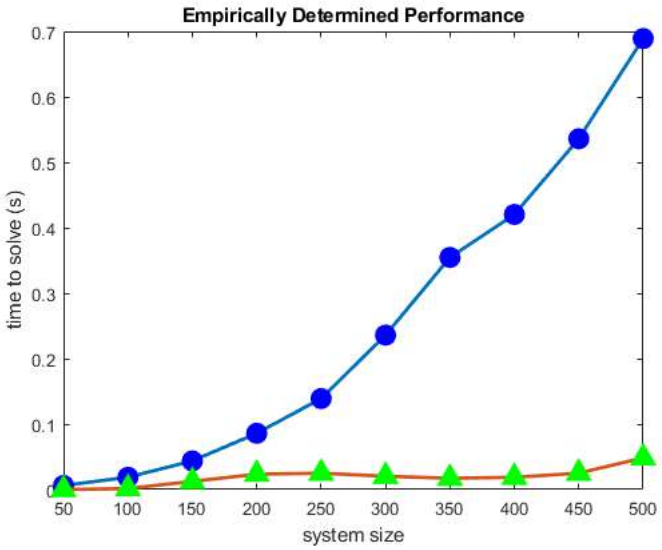
```
Start of tests of Gaussian-elimination scaling
 GE solution for system of size 50 takes average time 0.00625 s
 GE solution for system of size 100 takes average time 0.01875 s
 GE solution for system of size 150 takes average time 0.04375 s
 GE solution for system of size 200 takes average time 0.085938 s
 GE solution for system of size 250 takes average time 0.13906 s
 GE solution for system of size 300 takes average time 0.23594 s
 GE solution for system of size 350 takes average time 0.35469 s
 GE solution for system of size 400 takes average time 0.42031 s
 GE solution for system of size 450 takes average time 0.53594 s
 GE solution for system of size 500 takes average time 0.68906 s
```



Jacobi iteration

```
disp('Start of tests for Jacobi iteration');
tol=1e-9;
testtimes=zeros(size(nvals));
for in=1:numel(nvals)
    nlarge=nvals(in);
    Blarge=diag(-1*ones(nlarge-1,1),-1)+diag(-1*ones(nlarge-1,1),1)+diag(4*ones(nlarge,1),0);    %this must be diagonally dominant or else the method won't converge
    blarge=ones(nlarge,1);

    for irep=1:lrep     %benchmark will repeat the same solution several times to eliminate random variations from CPU load, etc.
        tstart=cputime;
        x0=randn(nlarge,1);
        [xit,iterations]=Jacobi(x0,Blarge,blarge,tol,false);
        tend=cputime;
        testtimes(in)=testtimes(in)+(tend-tstart)/lrep;
    end %for
    disp([' JI solution for system of size ',num2str(nlarge),' takes average time ',num2str(testtimes(in)),' s']);
end %for

TT(:,2) = testtimes';

figure(1);
hold on
plot(nvals,testtimes,'-^','LineWidth',2,'MarkerSize',10,'MarkerEdgeColor','green','MarkerFaceColor','green')
```

```
Start of tests for Jacobi iteration
 JI solution for system of size 50 takes average time 0 s
 JI solution for system of size 100 takes average time 0.0015625 s
 JI solution for system of size 150 takes average time 0.0125 s
 JI solution for system of size 200 takes average time 0.023438 s
 JI solution for system of size 250 takes average time 0.025 s
 JI solution for system of size 300 takes average time 0.020313 s
 JI solution for system of size 350 takes average time 0.017187 s
 JI solution for system of size 400 takes average time 0.01875 s
 JI solution for system of size 450 takes average time 0.025 s
 JI solution for system of size 500 takes average time 0.048438 s
```

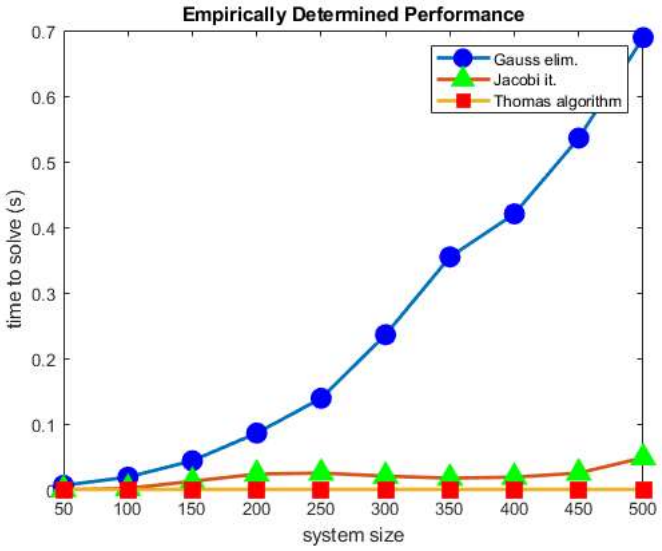


Thomas algorithm

```
disp('Start of tests for tridiag solver');
% tol=1e-9;
testtimes=zeros(size(nvals));
for in=1:numel(nvals)
    nlarge=nvals(in);
    Blarge=diag(-1*ones(nlarge-1,1),-1)+diag(-1*ones(nlarge-1,1),1)+diag(4*ones(nlarge,1),0);    %this must be diagonally dominant or else the method won't converge
    blarge=ones(nlarge,1);

    for irep=1:lrep     %benchmark will repeat the same solution several times to eliminate random variations from CPU load, etc.
        tstart=cputime;
        [xit,iterations]=tridiag(Blarge,blarge);
        tend=cputime;
        testtimes(in)=testtimes(in)+(tend-tstart)/lrep;
    end %for
    disp([' TA solution for system of size ',num2str(nlarge),' takes average time ',num2str(testtimes(in)),' s']);
end %for

TT(:,3) = testtimes';
disp('Time required for GE, JI, & TA (s)')
disp(TT)

figure(1);
```

```
hold on
plot(nvals,testtimes,'-s','LineWidth',2,'MarkerSize',10,'MarkerEdgeColor','red','MarkerFaceColor','red')
legend('Gauss elim.','Jacobi it.','Thomas algorithm')
```

```
Start of tests for tridiag solver
 TA solution for system of size 50 takes average time 0 s
 TA solution for system of size 100 takes average time 0 s
 TA solution for system of size 150 takes average time 0 s
 TA solution for system of size 200 takes average time 0 s
 TA solution for system of size 250 takes average time 0 s
 TA solution for system of size 300 takes average time 0 s
 TA solution for system of size 350 takes average time 0 s
 TA solution for system of size 400 takes average time 0 s
 TA solution for system of size 450 takes average time 0 s
 TA solution for system of size 500 takes average time 0 s
Time required for GE, JI, & TA (s)
    0.0063         0         0
    0.0187    0.0016         0
    0.0437    0.0125         0
    0.0859    0.0234         0
    0.1391    0.0250         0
    0.2359    0.0203         0
    0.3547    0.0172         0
    0.4203    0.0187         0
    0.5359    0.0250         0
    0.6891    0.0484         0
```



Published with MATLAB® R2020b