Manoli Tramoutanas
AMATH 301 A - Autumn 2020
October 29, 2020

Homework 3 Solutions

1. Speed and Accuracy for Multiple Solves of a Special 2000×2000 Matrix using Various Methods.

f)

	Time for 100 Vectors (s)	$\|\|r\|\|$
Backslash	10.0436	$4.7672 * 10^{-12}$
PLU Decomposition	0.8155	$2.9647 * 10^{-12}$
Inverse	0.3284	$1.0159 * 10^{-10}$

g) The method with the fastest time for solving 100 random vectors is the inverse matrix multiplication method.

h) The method with the most residual error is the inverse matrix multiplication method.

i) For a task where the wrong answer means life or death but computing speed is still important, I would choose to use PLU decomposition. This is because, when solving with 100 random vectors, the error is of the same low magnitude as the backslash solve (10^{-12}), but the time is of the same low magnitude as the inverse method (10^{-1} seconds).

Problem 1 Code

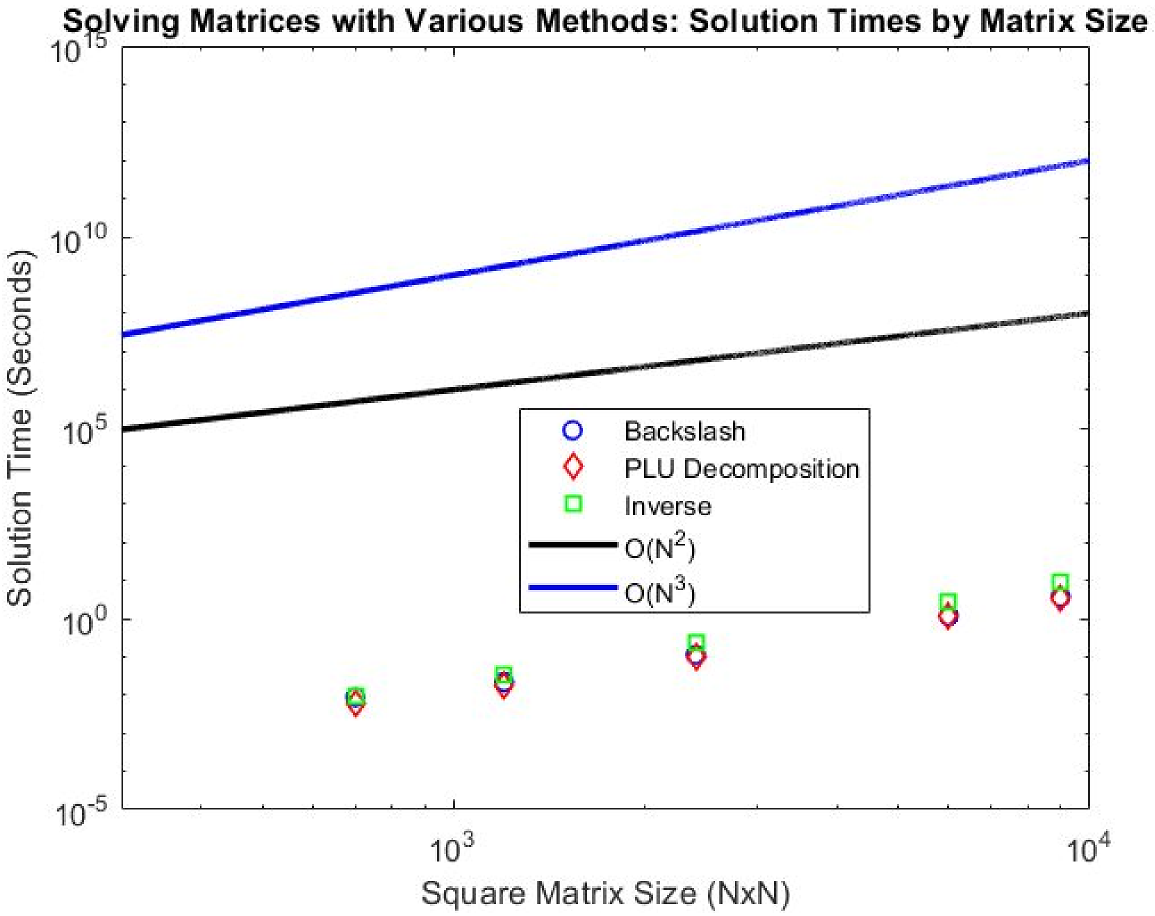
```
% a) Create Special Matrix
N = 2000;
A = -1*diag(ones(N,1)) + 4*diag(ones(N-1,1),1) + 4*diag(ones(N-1,1),-1);

% b) Run Tests for the Backslash Method
bsRes = 0;
tic;
for i = 1:100
   b = rand(2000, 1);
   x = A\b;
   % e) Record total residual magnitude for the Backslash Method
   bsRes = bsRes + norm(A*x-b);
end
bsTime = toc;

% c) Run Tests for PLU Decomposition
lupRes = 0;
tic
[L, U, P] = lu(A);
for i = 1:100
   b = rand(2000, 1);
   y = L\(P*b);
   x = U\y;
   % e) Record total residual magnitude for PLU Decomposition
   lupRes = lupRes + norm(A*x-b);
end
lupTime = toc;

% d) Run Tests for the Inverse Method
invRes = 0;
tic
invA = inv(A);
for i = 1:100
   b = rand(2000, 1);
   x = invA*b;
   % e) Record total residual magnitude for the Inverse Method
   invRes = invRes + norm(A*x-b);
end
invTime = toc;
```

2. The code produced the following figure:



j) The inverse method was consistently the slowest solution method with matrix size, while backslash and PLU decomposition were roughly equal in solution time as matrix size increased. In addition, it can be observed that the times for the inverse method grew further and further apart from the backslash and PLU times as matrix size increased, which suggests that the inverse method has a higher operation count per matrix size than both of the other methods (likely $O(N^3)$ as opposed to $O(N^2)$, as the sample curves on the graph appear to suggest).

Problem 2 Code

```
nVals = [700, 1200, 2400, 6000, 9000];

% b) Find Backslash solution times
bsTimes = [ bsRandSolve(700),...
          bsRandSolve(1200),...
          bsRandSolve(2400),...
          bsRandSolve(6000),...
          bsRandSolve(9000)];

% c) Find PLU Decomposition solution times
pluTimes = [pluRandSolve(700),...
          pluRandSolve(1200),...
          pluRandSolve(2400),...
          pluRandSolve(6000),...
          pluRandSolve(9000)];

% d) Find Inverse solution times
invTimes = [invRandSolve(700),...
          invRandSolve(1200),...
          invRandSolve(2400),...
          invRandSolve(6000),...
          invRandSolve(9000)];

n = 300:10000;
set(gca, 'Fontsize', 15);
% e-g) Plot solution times for all 3 solution methods
loglog(nVals, bsTimes, 'bo',...
   nVals, pluTimes, 'rd',...
   nVals, invTimes, 'gs',...
   'Linewidth', 1);
hold on
% h) Add trendlines for O(N^2) and O(N^3)
loglog(n, n.^2, 'k-',...
   n, n.^3, 'b-',...
   'Linewidth', 2);
% i) Make the graph look nice (title, axis labels, legend)
title('Solving Matrices with Various Methods: Solution Times by Matrix Size');
xlabel('Square Matrix Size (NxN)');
ylabel('Solution Time (Seconds)');
```

```
legend('Backslash', 'PLU Decomposition', 'Inverse', 'O(N^2)', 'O(N^3)',...
    'Location', 'Best');

% Random Solve Functions
function time = bsRandSolve(n)
    A = rand(n);
    b = rand(n, 1);
    tic
    A\b;
    time = toc;
end

function time = pluRandSolve(n)
    A = rand(n);
    b = rand(n, 1);
    tic
    [L, U, P] = lu(A);
    y = L\(P*b);
    U\y;
    time = toc;
end

function time = invRandSolve(n)
    A = rand(n);
    b = rand(n, 1);
    tic
    inv(A)*b;
    time = toc;
end
```