Table of Contents

Benchmark

The same input stack was scaled up and down in 14 steps by 10% (only in x-y plane, number of slices was unchanged) on each step.

Largest stack successfully procesed is [435 x 435 x 150] by CHOL solver.

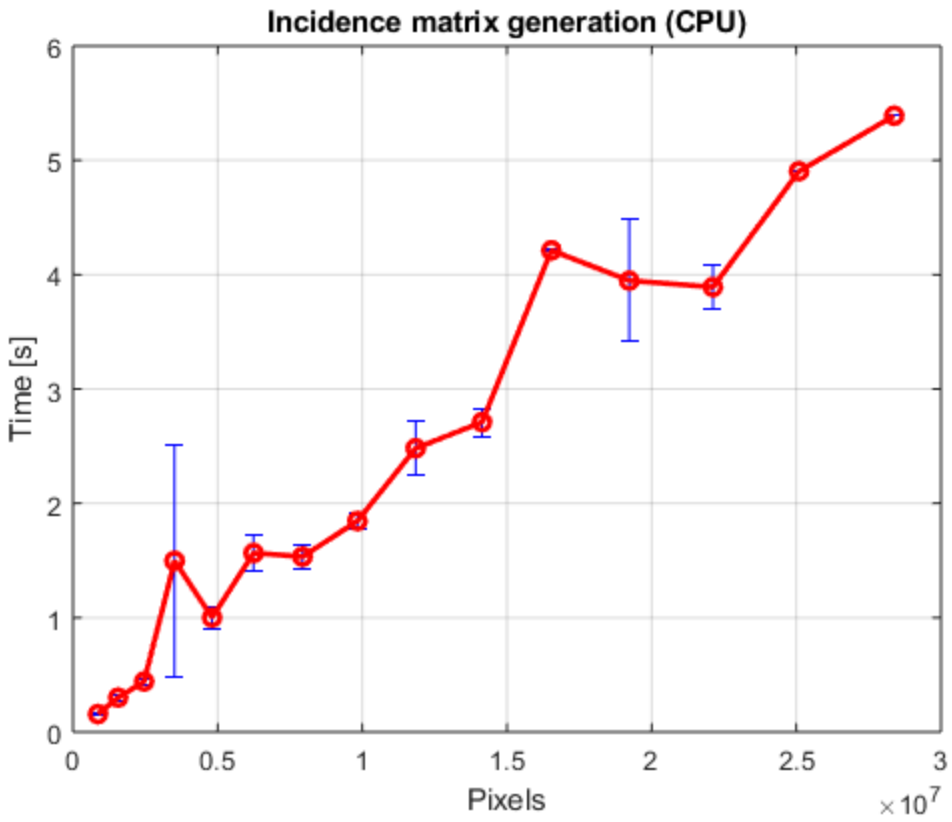
Seeds were computed for each input automatically with the same settings. Measurements were taken selectively for particular computational steps omitting image loading/saving, preprocessing and probability computation. The same set of images was tested twice for LU and CHOL solvers.

```
folder1 = 'CHOL';
folder2 = 'LU';

ff1 = dir(fullfile(folder1,'*.txt'));
ff2 = dir(fullfile(folder2,'*.txt'));

[hf1, dataf1, filesf1] = convertBench(fullfile(ff1.folder,ff1.name));
tf1 =
 array2table(dataf1,'RowNames',filesf1,'VariableNames',hf1(2:end));
tf1s = sortrows(tf1,'SSIZE');

[hf2, dataf2, filesf2] = convertBench(fullfile(ff2.folder,ff2.name));
tf2 =
 array2table(dataf2,'RowNames',filesf2,'VariableNames',hf2(2:end));
tf2s = sortrows(tf2,'SSIZE');
```

incidence (CPU)

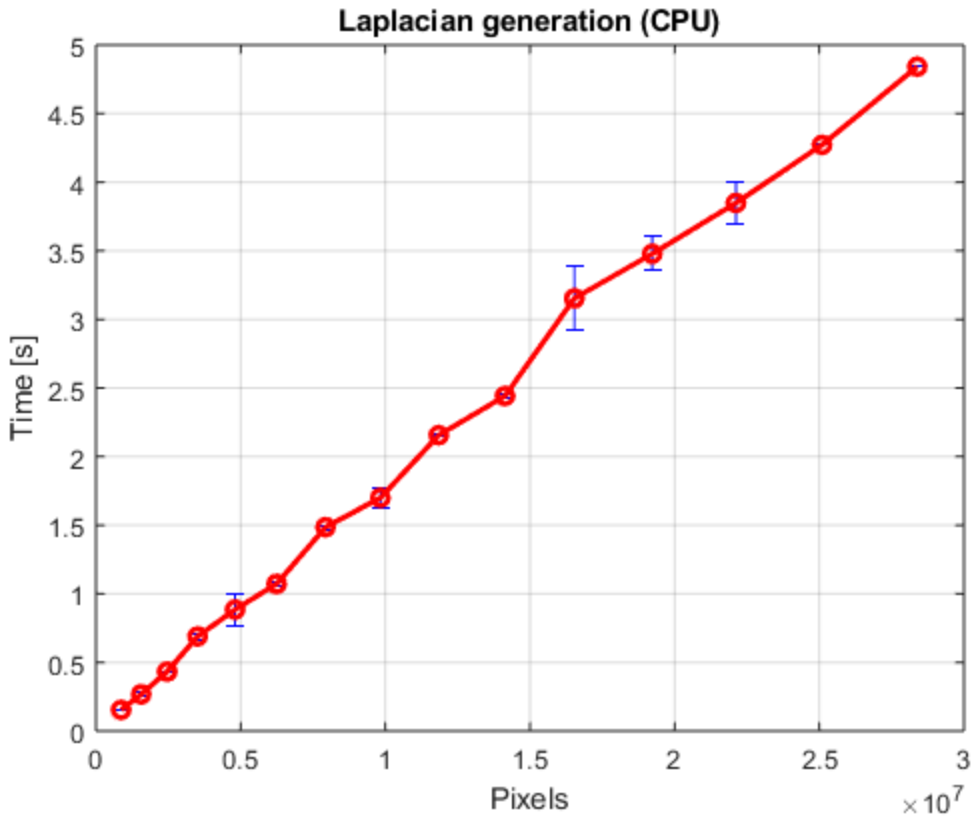
```
x = tf1s{:,'SSIZE'};
t1t2 = [tf1s{:,'INCIDENCE'}, [tf2s{:,'INCIDENCE'}; tf1s{:,'INCIDENCE'}
(end-1); tf1s{:,'INCIDENCE'}(end)]]';
t1t2 = t1t2/1000;
mm = mean(t1t2);
err = std(t1t2);

errorbar(x,mm,err,'-b');
```

```
hold on
plot(x,mm,'-ro', 'LineWidth',2);
grid on
xlabel('Pixels')
ylabel('Time [s]')
title('Incidence matrix generation (CPU)')
hold off
```



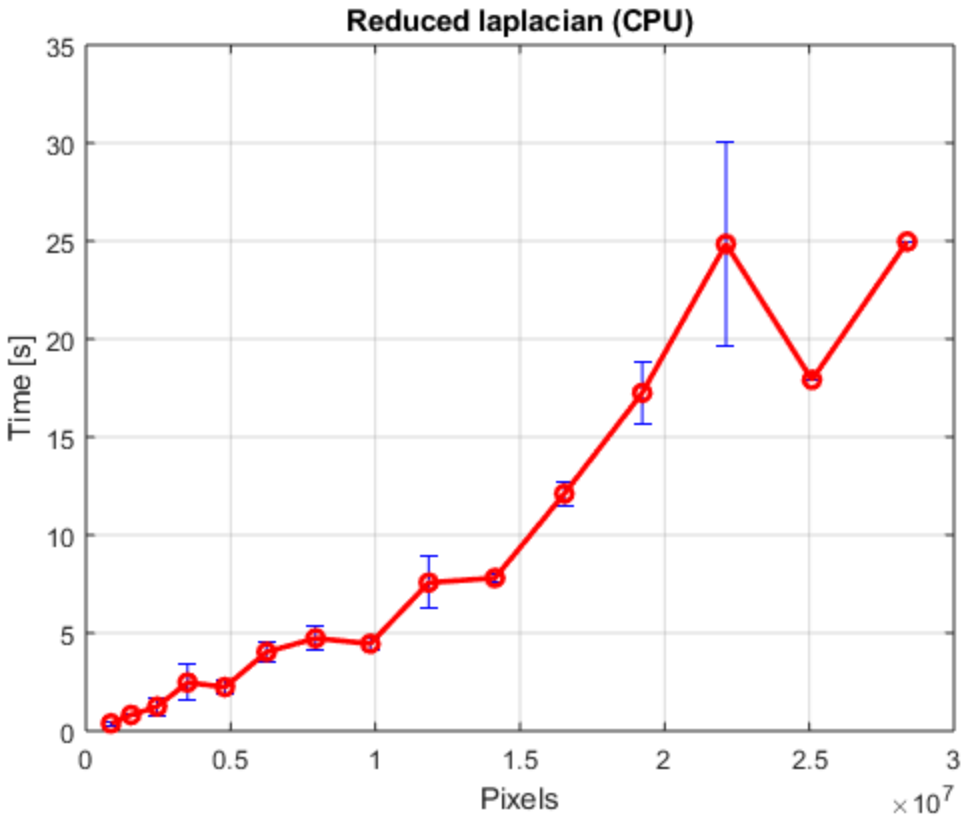
laplacian (CPU)

```
x = tf1s{:,'SSIZE'};
t1t2 = [tf1s{:,'LAPLACIAN'}, [tf2s{:,'LAPLACIAN'}; tf1s{:,'LAPLACIAN'}
(end-1); tf1s{:,'LAPLACIAN'}(end)]]';
t1t2 = t1t2/1000;
mm = mean(t1t2);
err = std(t1t2);

errorbar(x,mm,err,'-b');
hold on
plot(x,mm,'-ro', 'LineWidth',2);
grid on
xlabel('Pixels')
ylabel('Time [s]')
title('Laplacian generation (CPU)')
hold off
```



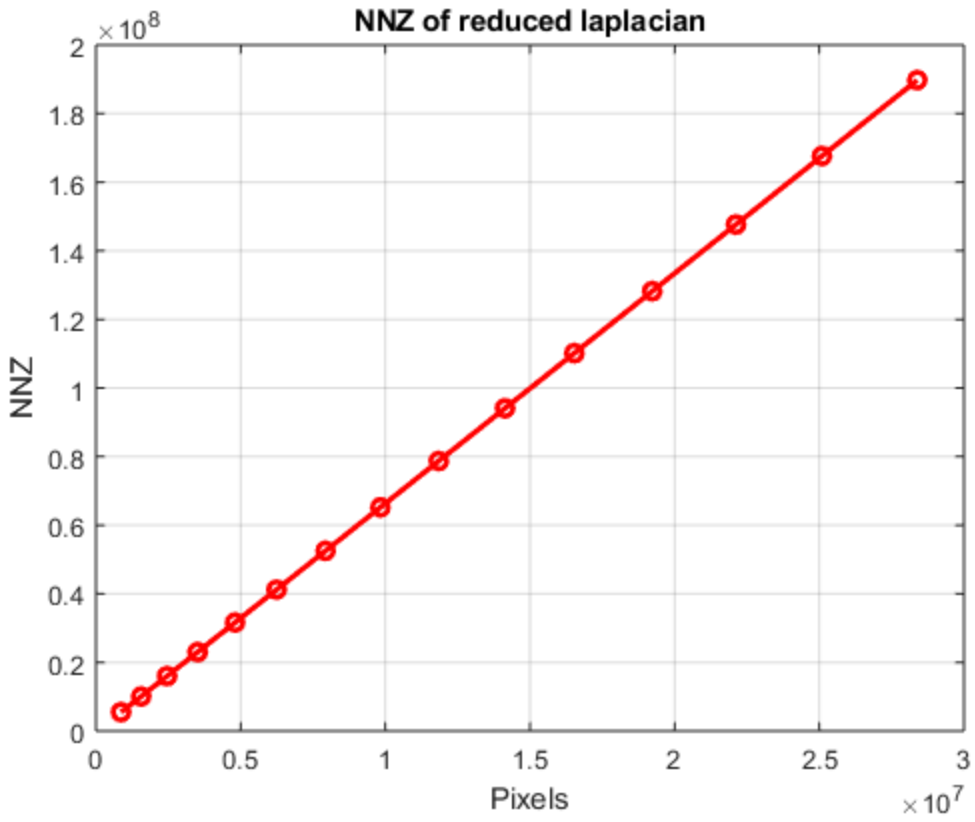
Reducing laplacian (CPU)

```
x = tf1s{:,'SSIZE'};
t1t2 = [tf1s{:,'RLAPLACIAN'}, [tf2s{:,'RLAPLACIAN'};
 tf1s{:,'RLAPLACIAN'}(end-1); tf1s{:,'RLAPLACIAN'}(end)]]';
t1t2 = t1t2/1000;
mm = mean(t1t2);
err = std(t1t2);

errorbar(x,mm,err,'-b');
hold on
plot(x,mm,'-ro', 'LineWidth',2);
grid on
xlabel('Pixels')
ylabel('Time [s]')
title('Reduced laplacian (CPU)')
hold off
```



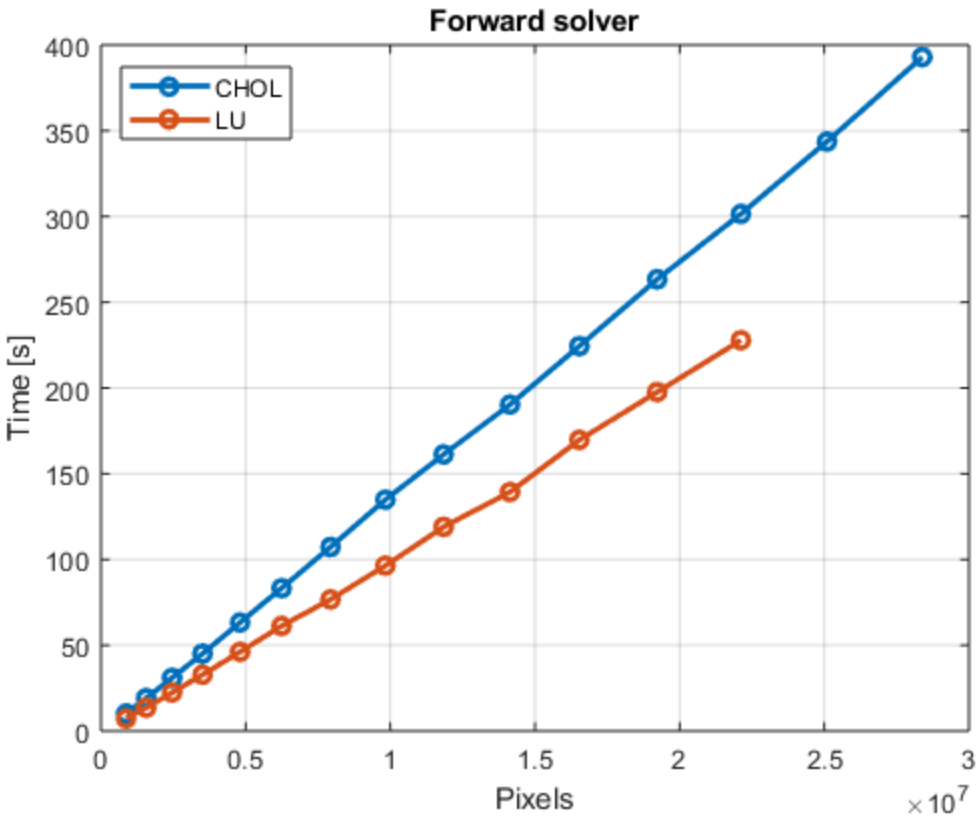
Size of R laplacian

```
x = tf1s{:,'SSIZE'};
t1t2 = tf1s{:,'RNNZ'}';

plot(x,t1t2,'-ro', 'LineWidth',2);
grid on
xlabel('Pixels')
ylabel('NNZ')
title('NNZ of reduced laplacian')
hold off
```



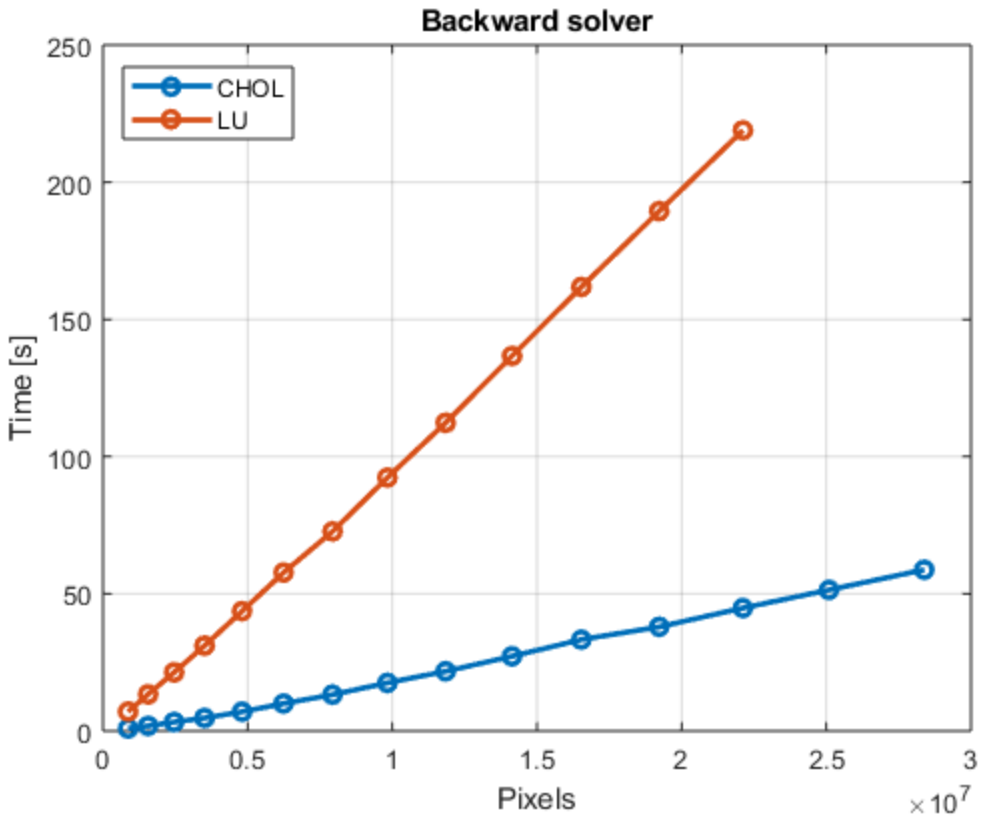
f-solve both

```
x = tf1s{:,'SSIZE'};
t1t2 = [tf1s{:,'FSOLVE'}, [tf2s{:,'FSOLVE'}; NaN; NaN]]';
t1t2 = t1t2/1000;

plot(x,t1t2,'-o', 'LineWidth',2);
grid on
xlabel('Pixels')
ylabel('Time [s]')
title('Forward solver')
hold off
legend('CHOL','LU','Location','northwest')
```



b-solve both

```
x = tf1s{:,'SSIZE'};
t1t2 = [tf1s{:,'BSOLVE'}, [tf2s{:,'BSOLVE'}; NaN; NaN]]';
t1t2 = t1t2/1000;

plot(x,t1t2,'-o', 'LineWidth',2);
grid on
xlabel('Pixels')
ylabel('Time [s]')
title('Backward solver')
hold off
legend('CHOL','LU','Location','northwest')
```



total-solve both

```
x = tf1s{:,'SSIZE'};
t1tb = [tf1s{:,'BSOLVE'}, [tf2s{:,'BSOLVE'}; NaN; NaN]]';
t1tf = [tf1s{:,'FSOLVE'}, [tf2s{:,'FSOLVE'}; NaN; NaN]]';
t1t2 = t1tb + t1tf;
t1t2 = t1t2/1000;

plot(x,t1t2,'-o', 'LineWidth',2);
grid on
xlabel('Pixels')
ylabel('Time [s]')
title('Full solution')
hold off
legend('CHOL','LU','Location','northwest')
```

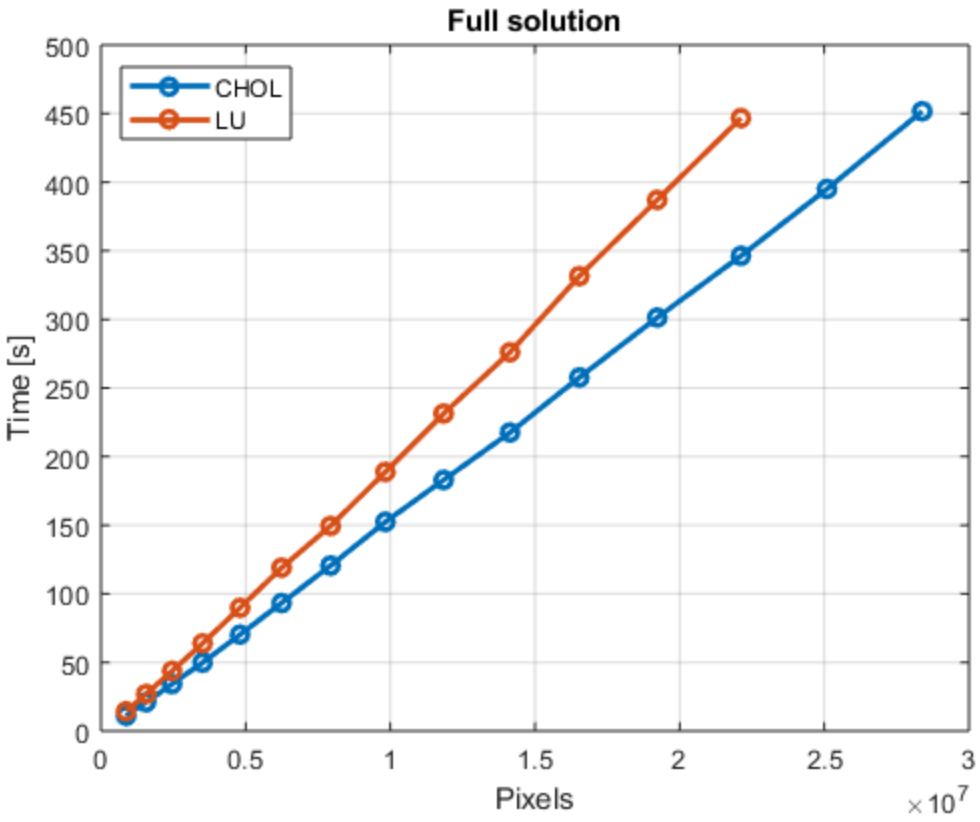
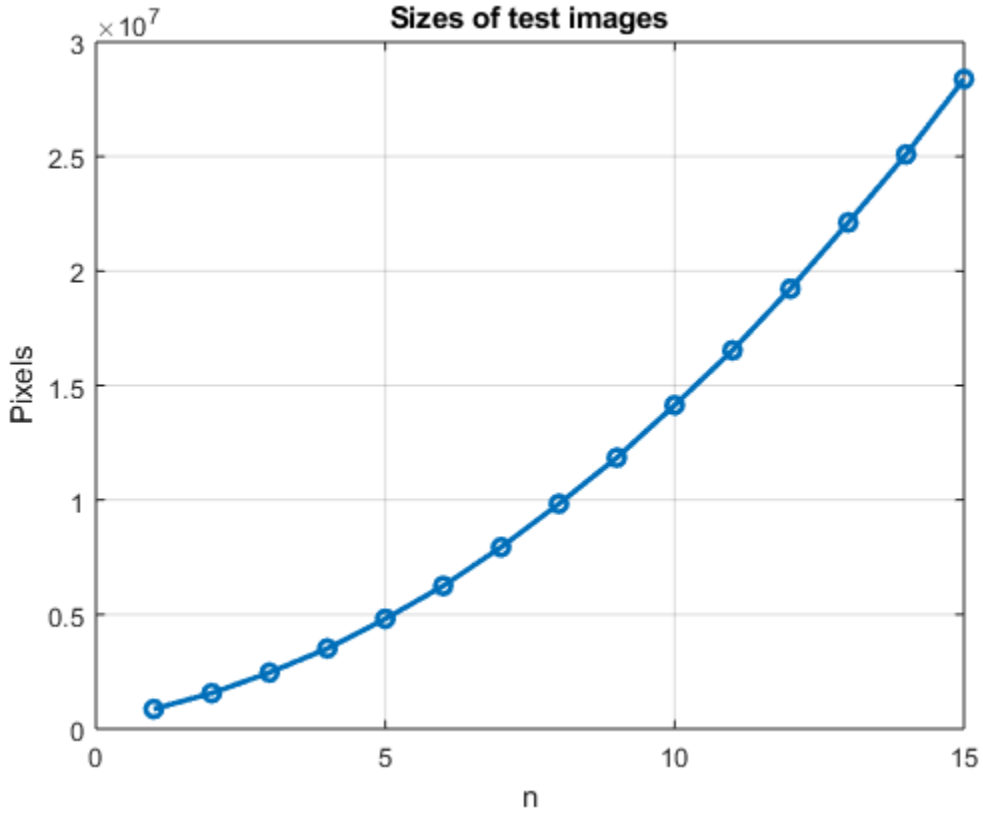


image size

```
plot(x,'-o', 'LineWidth',2);
grid on
xlabel('n')
ylabel('Pixels')
title('Sizes of test images')

function [headers, ret, files] = convertBench(file)

    im = importdata(file);
    imt = im.textdata;
    headers = {imt{1,1}, imt{1,3}, imt{1,5}, imt{1,7}, imt{1,9},
 imt{1,11}, imt{1,13}, imt{1,15}, imt{1,17}};

    cols = [4 6 8 10 12 14 16];
    ret = zeros(size(imt,1), length(cols)+1);
    l = 1;
    for i = cols
        cc = {imt{:,i}};
        aa = cellfun(@str2num, cc)';
        ret(:,l) = aa;
        l = l + 1;
    end
    ret(:,end) = im.data;
```

```
    files = imt(:,2);
end
```



Published with MATLAB® R2018a