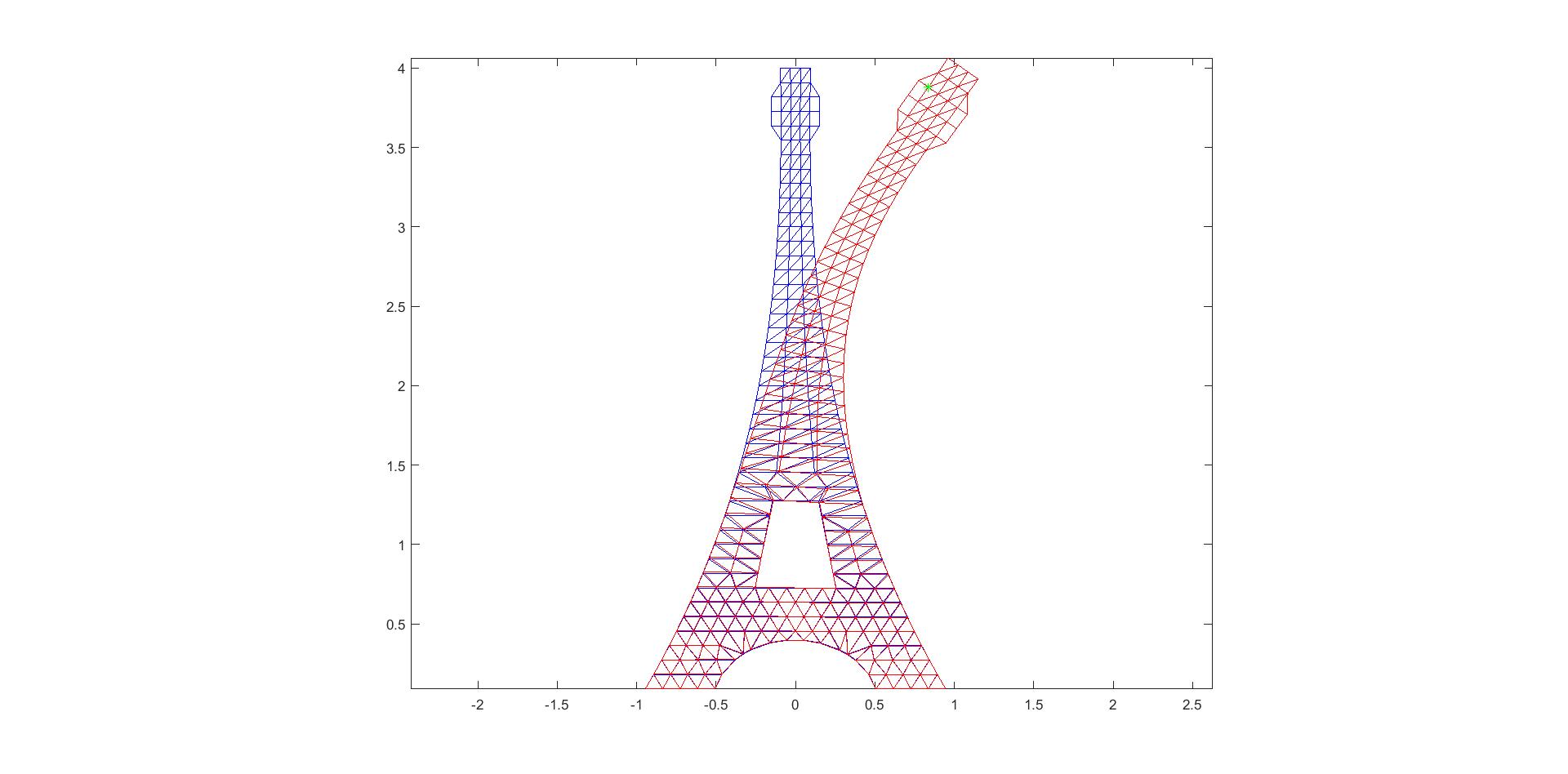
HW 4

(MATH/CS 375)

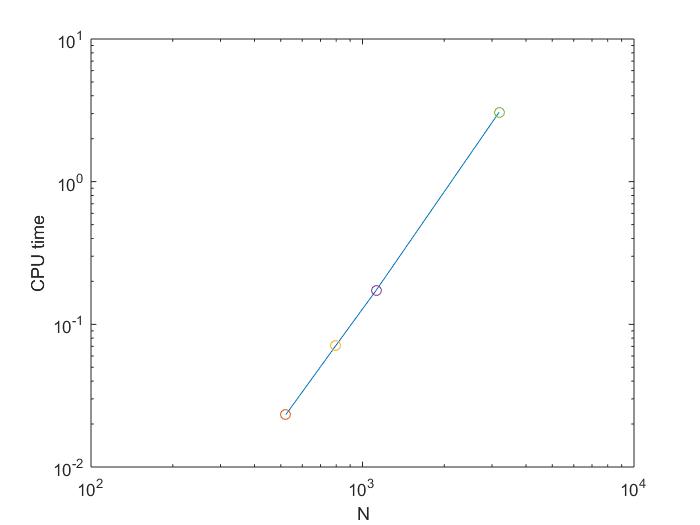
Jacob Hurst

03/01/18

Problem 1:

B @ 499 loaded with force of magnitude 1/5.

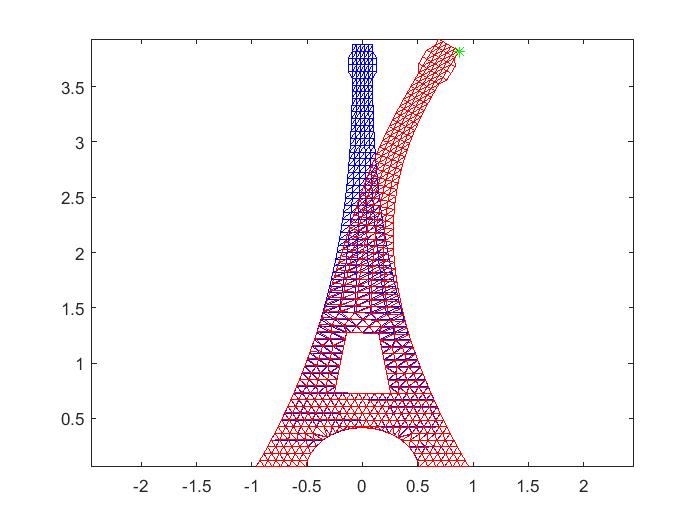
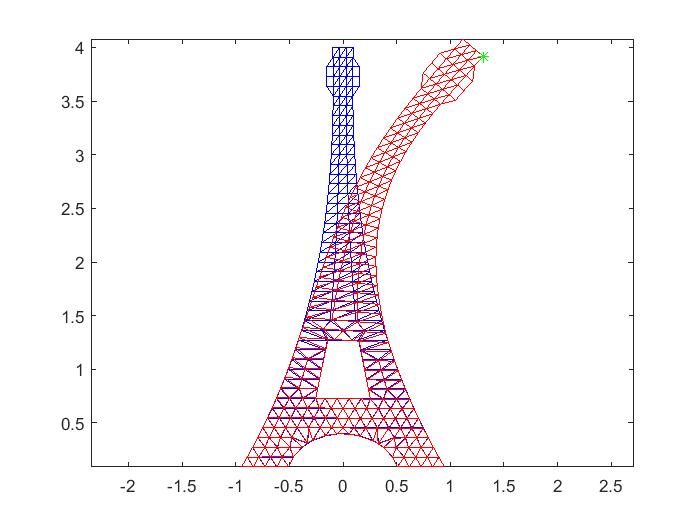
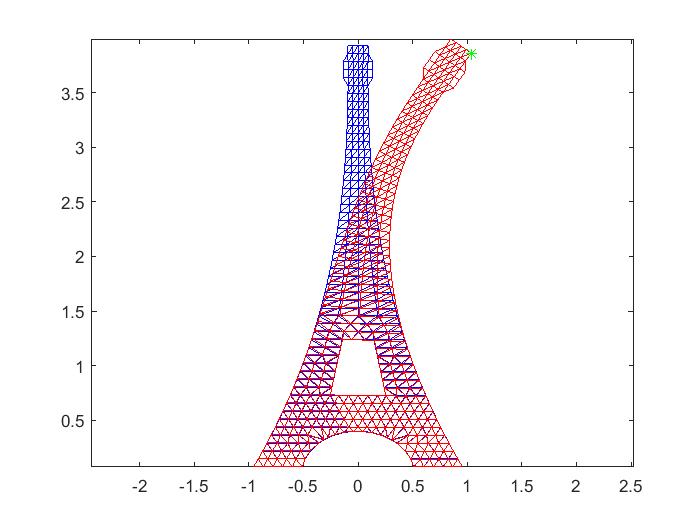
Problem 2:

From theory, we can expect the cost to be roughly m\*O(n^3). From the plot, we can observe that the CPU running time is approximately of order of n^3.

Problem 3a:

Eiffel2.mat

Eiffel1.mat

Eiffel1.mat  
Max displacement at node: 521  
CPU time: 7.109375

Eiffel3.mat

Eiffel2.mat  
Max displacement at node: 797  
CPU time: 39.312500

Eiffel3.mat  
Max displacement at node: 1121  
CPU time: 123.937500

Eiffel4.mat - personal computer reached its limit here.

Problem 3b:

Optimizing with LU-decomposition, we see, from our results, a significant improvement in CPU time roughly by a factor of n.

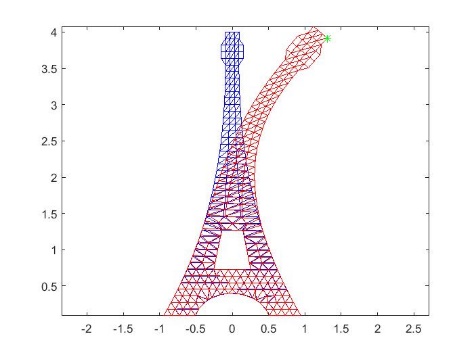
Eiffel1.mat  
Max displacement at node: 521  
CPU time: 0.750000

Eiffel2.mat  
Max displacement at node: 797  
CPU time: 2.578125

Eiffel3.mat  
Max displacement at node: 1121  
CPU time: 6.984375

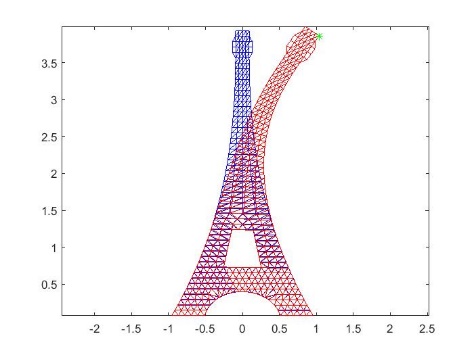
Eiffel4.mat  
Max displacement at node: 3183  
CPU time: 133.125000

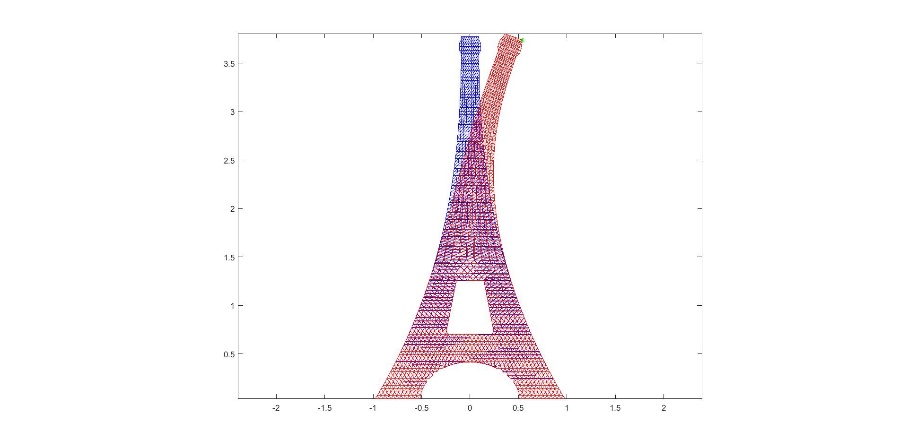
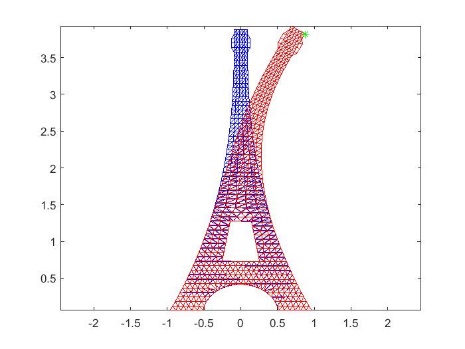
Problem 4:

**Sparse A without LU-decomposition:**

Eiffel 2

Eiffel 1

Eiffel1.mat  
Max displacement at node: 521  
CPU time: 0.921875

Eiffel2.mat  
Max displacement at node: 797  
CPU time: 2.000000

Eiffel3.mat  
Max displacement at node: 1121  
CPU time: 4.640625

Eiffel4.mat  
Max displacement at node: 3183  
CPU time: 43.718750

Problem 4 (cont’d):

**Sparse A with LU-decomposition:**

Eiffel1.mat  
Max displacement at node: 521  
CPU time: 0.468750

Eiffel2.mat  
Max displacement at node: 797  
CPU time: 1.531250

Eiffel3.mat  
Max displacement at node: 1121  
CPU time: 4.812500

Eiffel4.mat  
Max displacement at node: 3183  
CPU time: 86.718750

On average, the sparse method in MATLAB is approximately 1.5 times faster. Oddly, for a large A, say eiffel4.mat, the CPU time was 1.5 times faster without sparse(A).