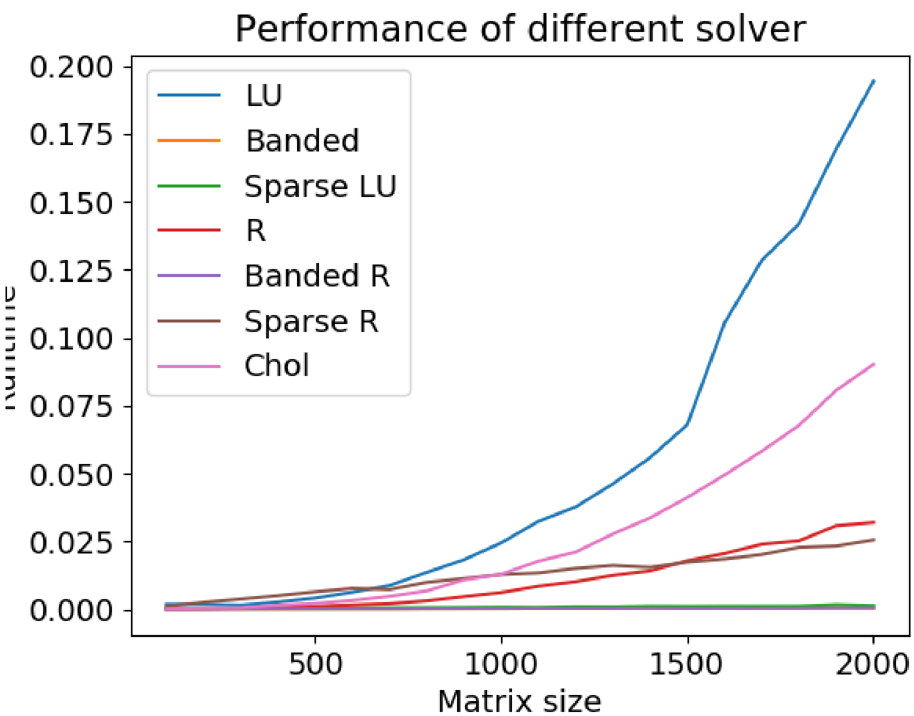
Q1:

a - g):

The code for these sub questions are written in a2-report.py as *timing()*.

h):

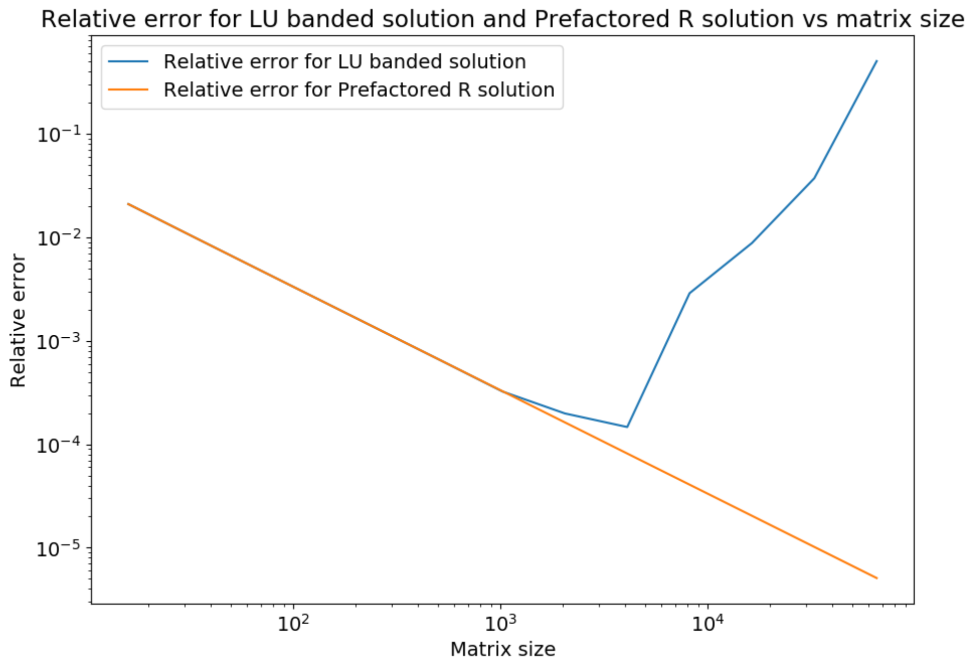
The runtime performance of different solvers is shown below:



i): The graph we produce from in part h) generally follows the theoretical bound given in the textbook and lectures. The process of Gaussian elimination (LU in the picture) will take $\frac{n^3}{3} + \frac{5n^2}{2} + n$ flops to finish the calculation, which is the highest flops among all the other algorithm. Thus, it takes the longest time to compute and this is reflected on the graph. The LU factorization algorithm is followed by Cholesky factorization. The from the graph, the Cholesky solver complete the calculation in half of the time compare to the LU solver, which perfectly matches its theoretical bound about half the flops of the Gaussian solver. At the lower end, we have the Banded solvers, all of the banded solvers have a theoretical bound of $O(\beta^2 n)$, thus they should have linear runtime. From the graph, we can see that all our banded solver has a very low runtimes compare to other solvers since their run time is linear. For the triangular matrix solvers, the sparce solver seems to have around the same runtime compare to the dense solver, the triangular solvers agree with their theoretical bounds of $O(n^2)$, which is clearly shown in the graph(R and Sparce R).

Q2:

a):



b):

From the plot, we can observe that the relative accuracy for both algorithm changes with respect to different matrix sizes. Before size 1000, the two algorithms have comparable precision. However, after size 1000, the prefactored R solution is significantly more accurate than direct calculation. One possible reason for the large relative error in direct calculation is that the direct calculation is that the pre-factored algorithm will just need to perform back and forward substitution on matrices that are already in lower or upper triangle form (This is accurate since we calculate it by hand) instead of doing Gaussian elimination where numerical errors (like rounding and truncation error) can accumulate in the elimination process.