Homework 1 - STAT 540

Amal Agarwal

Answer 1

(a) The Monte Carlo estimate of the computational cost is the sample mean of the time for several (M) iterations for some vector length n. To choose an appropriate M, note that the approximate confidence interval for the MC estimate is of the form $\overline{t_n} \pm \frac{z_{\alpha/2} s}{\sqrt{M}}$ where s is the sample standard deviation. For a fixed n, we can define the relative error as the ratio of the length of the interval to the point estimate. If we bound this relative error by some tolerance (tol) then we obtain the following inequality:

$$\frac{2 z_{\alpha/2} s/\sqrt{M}}{\overline{t_n}} \leq tol$$

Solving the above inequality yields:

$$M \geq \frac{4 z_{\alpha/2}^2 s^2}{tol^2 \overline{t_n}^2}$$

Now note that the estimates $\overline{t_n}$ and s^2 must converge to the true values for sufficiently large M. Thus we can pick up M=1000 (say) to calculate $\overline{t_n}$ and s^2 which should give the lower bound of M required from above inequality. Note that bounding the relative errors like this ensures comparibilty of errors across different n's. We can then conclude that this M is optimal.

The following 2 plots showing the convergence of means and variances with number of iterations suggests that we can use law of large numbers to conclude the M=1000 is more than sufficient for getting true value of mean and variance. We can use these values of sample mean and variance and an appropriate tolerance to get optimal M.

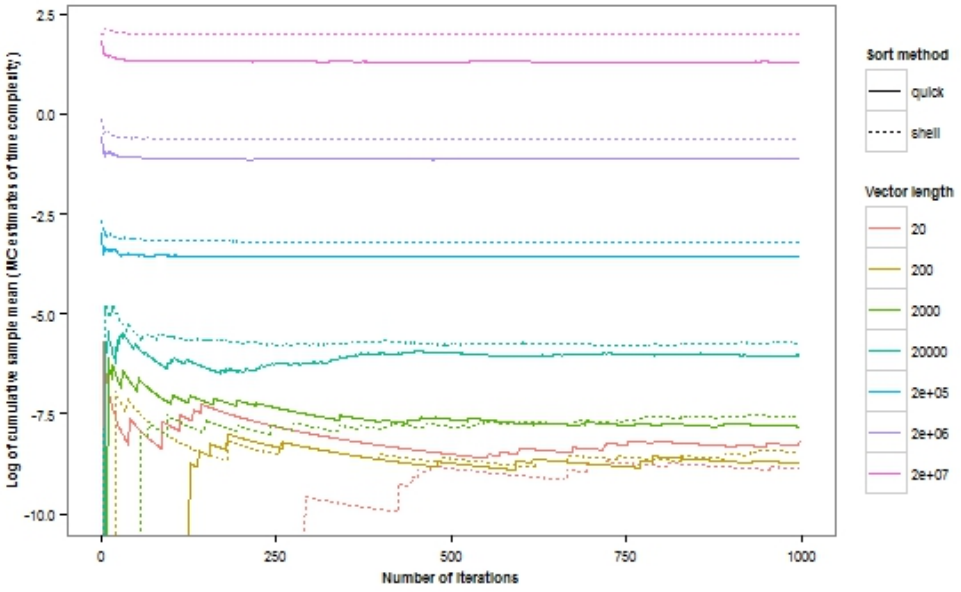

Figure 1

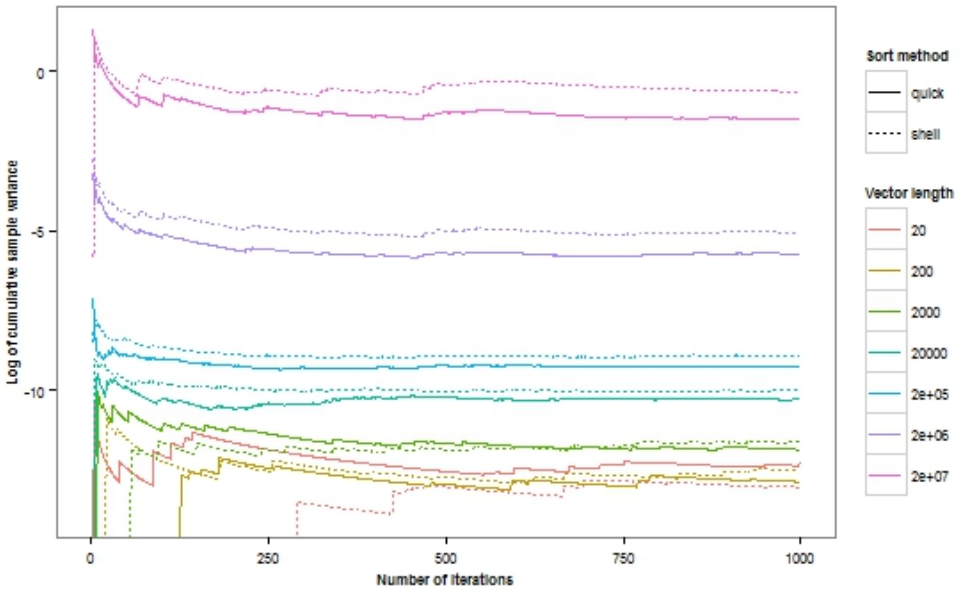

Figure 2

The variation of optimal M chosen for each n vs. n is shown below:

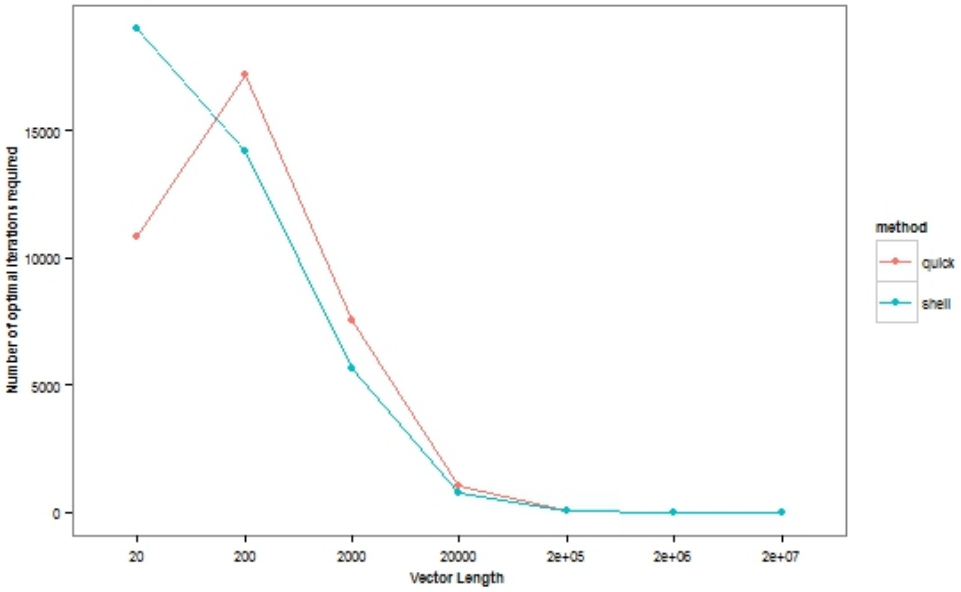

Figure 3

(b) The following table gives the Monte Carlo estimates and standard errors for time complexity of both quicksort and shellsort algorithms. Note that we are using total system elapsed time in seconds to calculate the estimates.

Vector length (n)	Quicksort		Shellsort	
	Monte Carlo estimates	Monte Carlo standard errors	Monte Carlo estimates	Monte Carlo standard errors
20	2.7×10^{-4}	4.63×10^{-6}	1.4×10^{-4}	2.18×10^{-6}
200	1.6×10^{-4}	2.57×10^{-6}	2.1×10^{-4}	3.66×10^{-6}
2000	4.0×10^{-4}	7.05×10^{-6}	5.2×10^{-4}	8.95×10^{-6}
20000	2.41×10^{-3}	3.49×10^{-5}	3.25×10^{-3}	4.5×10^{-5}
200000	2.8×10^{-2}	9.46×10^{-5}	4.1×10^{-2}	1.3×10^{-4}
2000000	3.23×10^{-1}	3.21×10^{-3}	5.32×10^{-1}	6.29×10^{-3}
20000000	3.7	2.17×10^{-1}	7.43	5.18×10^{-1}

(c) The following plot shows the variation of Monte Carlo estimates of computational cost with error bars vs. the vector length n for both the methods.

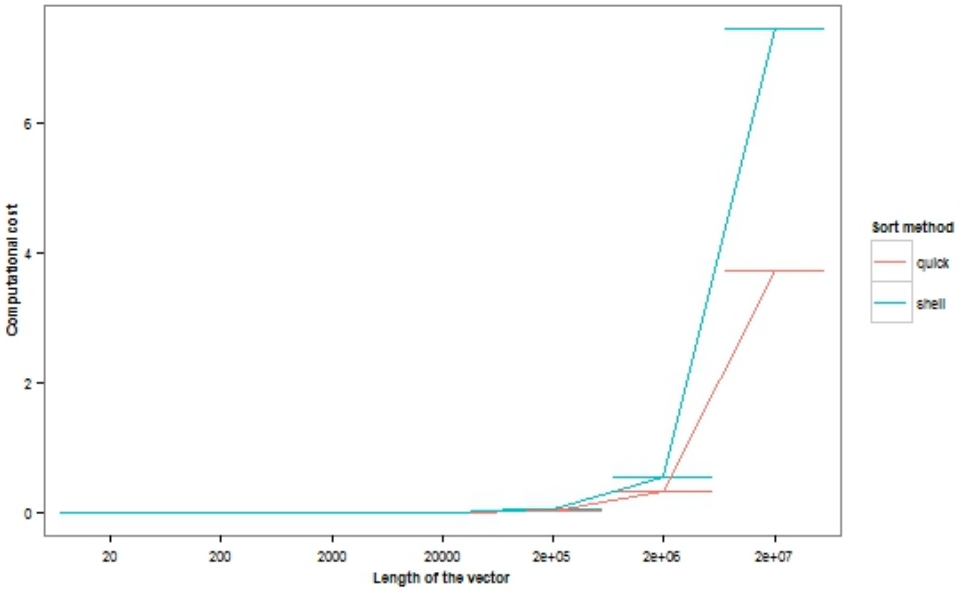

Figure 4

Clearly the above plot does not show relative increase for lower n. Therefore we plot the variation of log of Monte Carlo estimates of computational cost with error bars vs. the number of iterations for different n and both the methods.

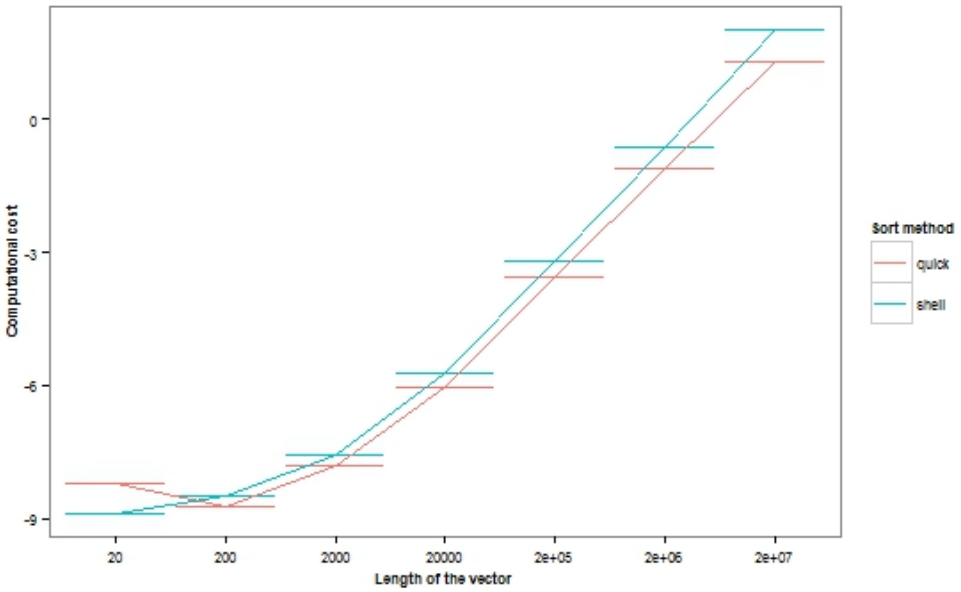

Figure 5

(d) Based on the above study, we can draw following conclusions:

- This simulation study goes upto 20 million only for the vector length and does not take the case of 200 million since my humble computer did not allow that due to memory allocation problems for a 1.5GB vector. I still don't have access to clusters and if there is an efficient way to do it on a slow computer, I failed to figure it out.
- M decreases as function of n. This can be attributed to our assumption that we are bounding the relative errors and the sample variance decreases at a faster rate compared to the rate of increase of sample mean.
- With increase in n, the time complexity increases faster for shellsort compared to quicksort.
- We can plot the computational cost vs. number of iterations for different n as follows:

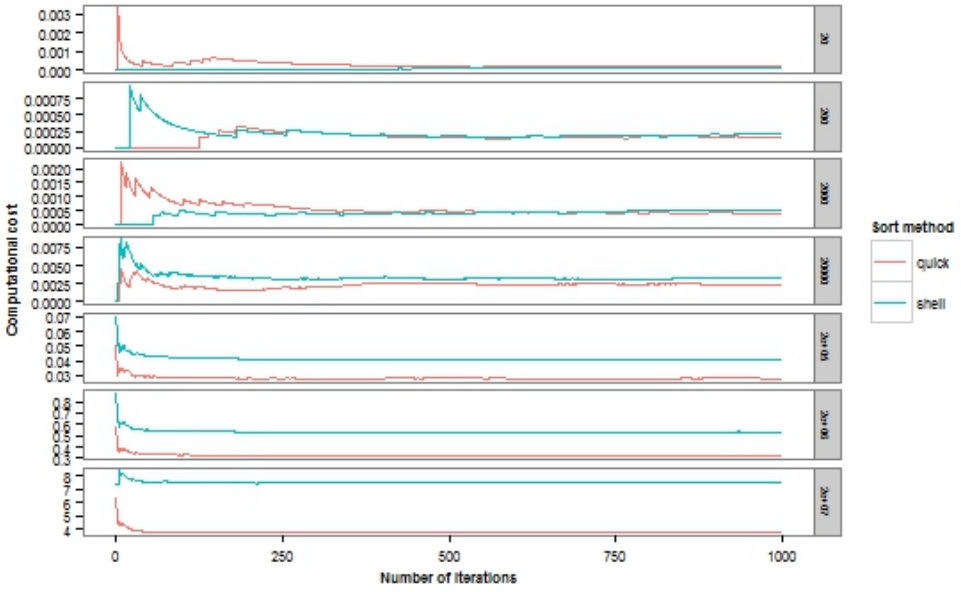

Figure 6

The above plot shows that for $n \leq 20000$, estimates of time complexities for both quicksort and shellsort converge to similar values. In fact shellsort converges faster compared to quicksort for low n. But as n increases this difference becomes wider and wider and shellsort takes considerably more time. We will see in part (e) that this behaviour agrees with the analytical results on run times for these algorithms.

(e) The worst case scenario for the time complexity of both quick sort and shell sort is $O(n^2)$. The best case scenario for quicksort and shellsort are $O(nlog(n))$ and $O(nlog^2(n))$ respectively. Plotting the estimates as a function of n and comparing this with the theoretical time complexities for both quicksort and shellsort. Here we plot the the ratio of our MC estimates with theoretical time complexities (for best case) vs. n for both the methods.

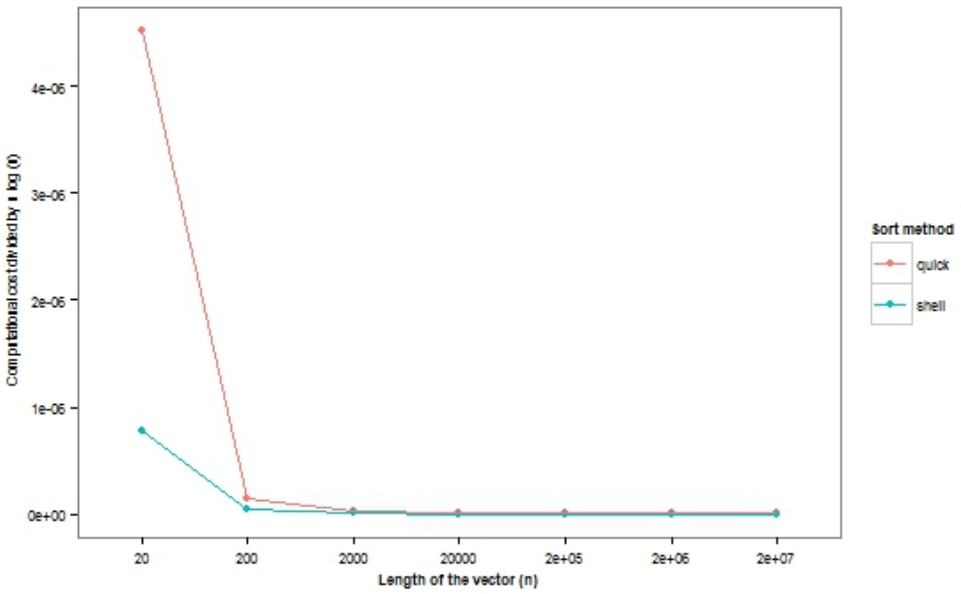

Figure 7

The above plot shows that for quicksort since this ratio converges to zero, $\frac{\overline{t_n}}{nlog(n)} = o(1)$. Clearly this implies we can choose any value as an upper bound i.e. $\frac{\overline{t_n}}{nlog(n)} = O(1)$. Similar reasoning for shellsort leads us to conclude that $\frac{\overline{t_n}}{nlog^2(n)} = O(1)$. Thus we can conclude that results from simulation agree with the analytical results.

Answer 2

Please see the file "aua257HW1.py" for Python code.

Answer 3

(a) Please see the file "aua257HW1.py" for Python code.

(b) The value of the sum of squares is 5103133479.22.