# ME766 HW1

Anmol Mishra

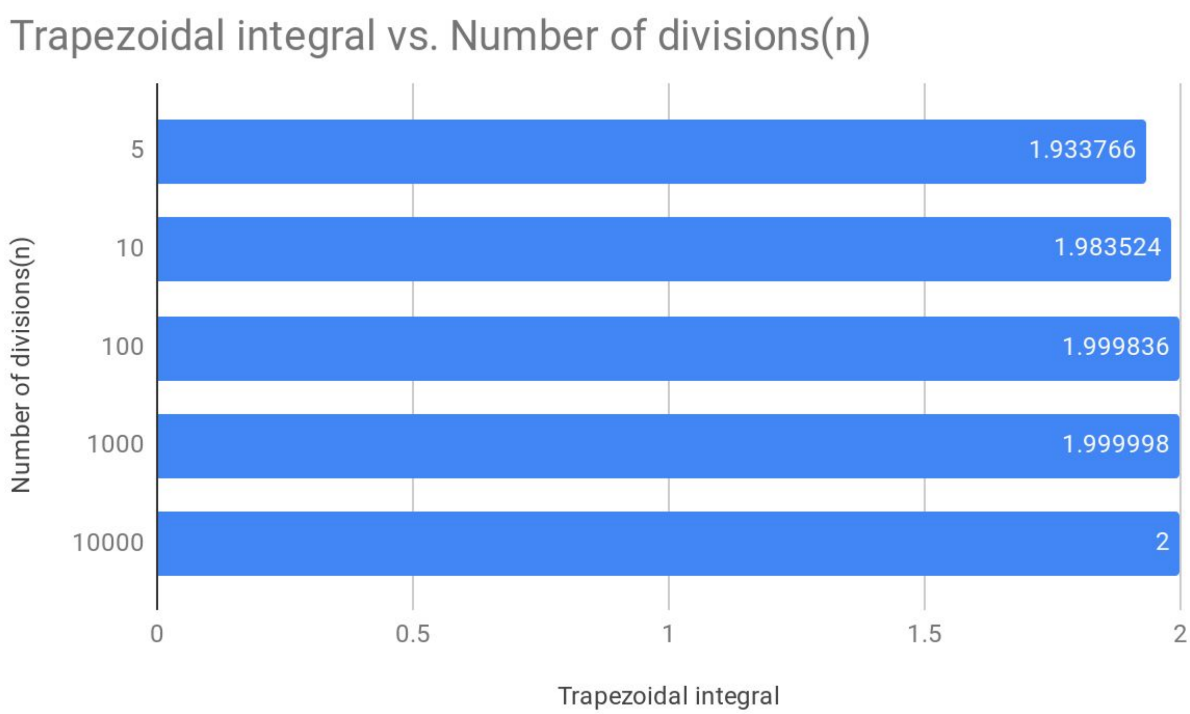
150010041

1. a) trap_s(serial) and trap_o(openmp) executables have been provided. The respective codes are in trap_s.c and trap_o.c

   b) mont_s(serial) and mont_o(openmp) executables have been provided. The respective codes are in mont_s.c and mont_o.c

2. a) We compute the integral using trapezoidal method for 5 different divisions. The results are as shown below -

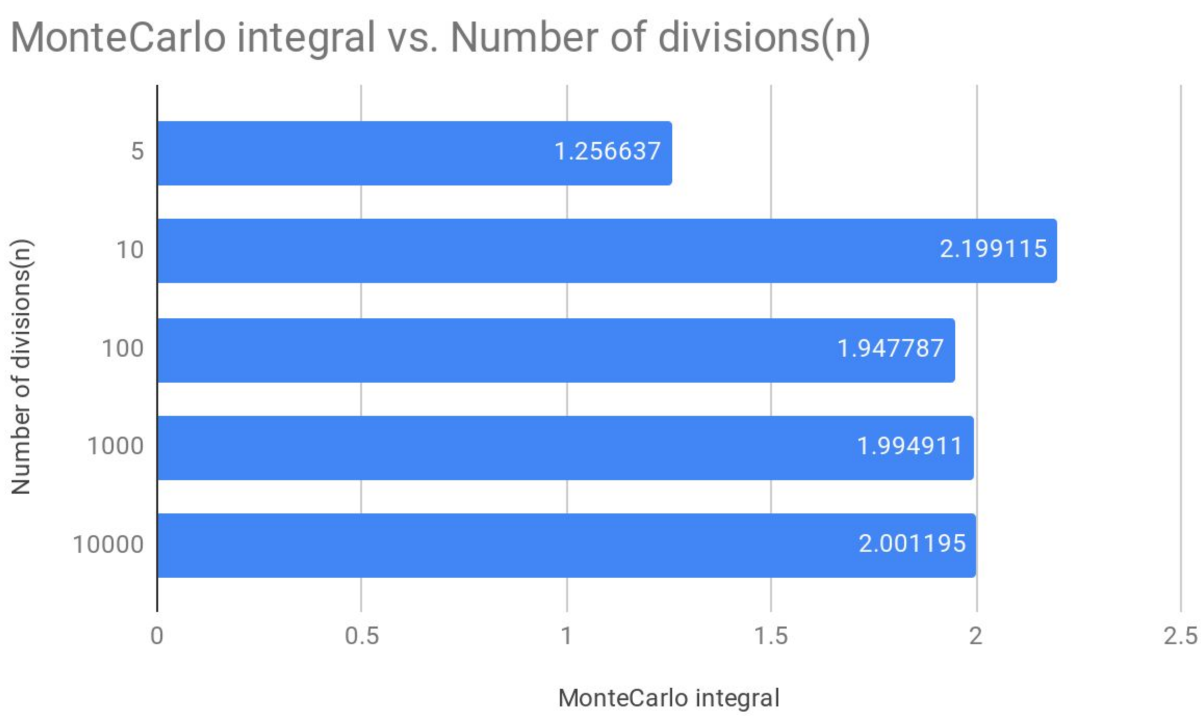
| Number of divisions(n) | Trapezoidal integral | Analytical Value | Percentage Error |
|---|---|---|---|
| 5 | 1.933766 | 2 | -3.3117 |
| 10 | 1.983524 | 2 | -0.8238 |
| 100 | 1.999836 | 2 | -0.0082 |
| 1000 | 1.999998 | 2 | -0.0001 |
| 10000 | 2 | 2 | 0 |




The results obtained are according to expectations. The trapezoidal integral approaches the analytical value 2 from left hand side and the percentage error is decreasing with increasing value of n.

b) We compute the integral using Monte Carlo method for 5 different divisions. The results are as shown below -

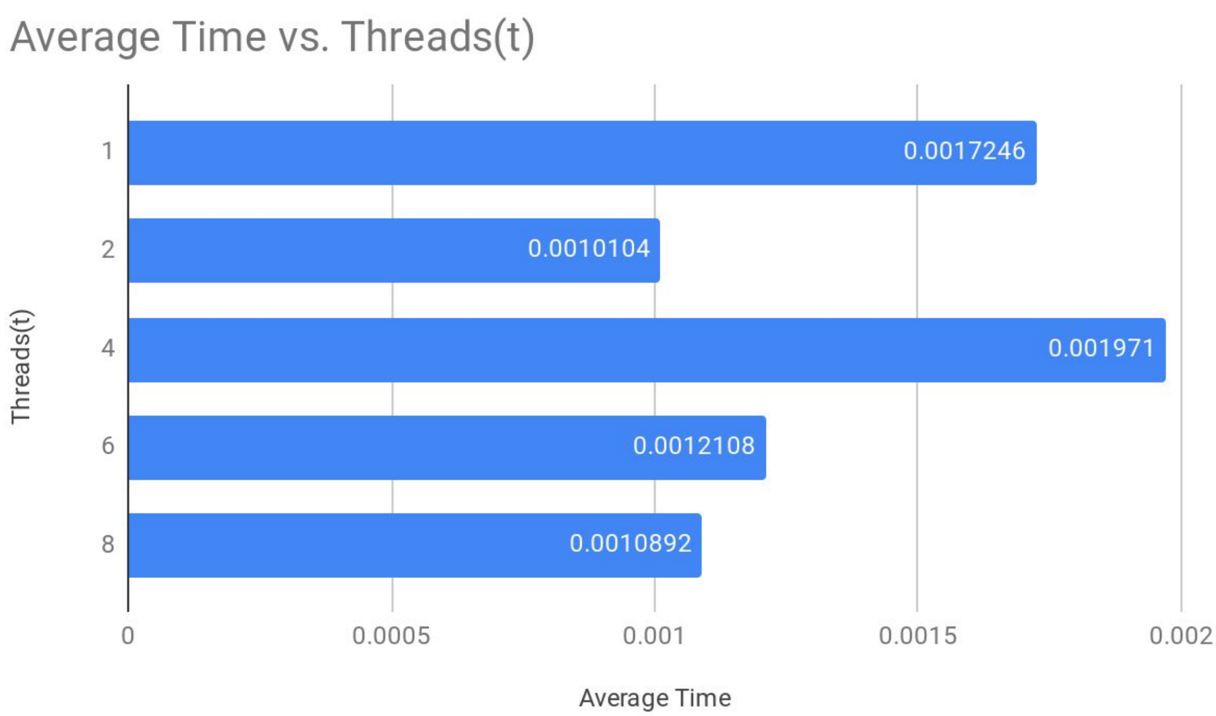
| Number of divisions(n) | MonteCarlo integral | Analytical Value | Percentage Error |
|---|---|---|---|
| 5 | 1.256637 | 2 | -37.16815 |
| 10 | 2.199115 | 2 | 9.95575 |
| 100 | 1.947787 | 2 | -2.61065 |
| 1000 | 1.994911 | 2 | -0.25445 |
| 10000 | 2.001195 | 2 | 0.05975 |




The results obtained are according to expectations. The Monte Carlo integral oscillates about the analytical value 2 with increasing n but the absolute value of percentage error keeps on decreasing with increasing n.

3. a) We compute the integral using Trapezoidal method for n = 10000 and different number of threads -

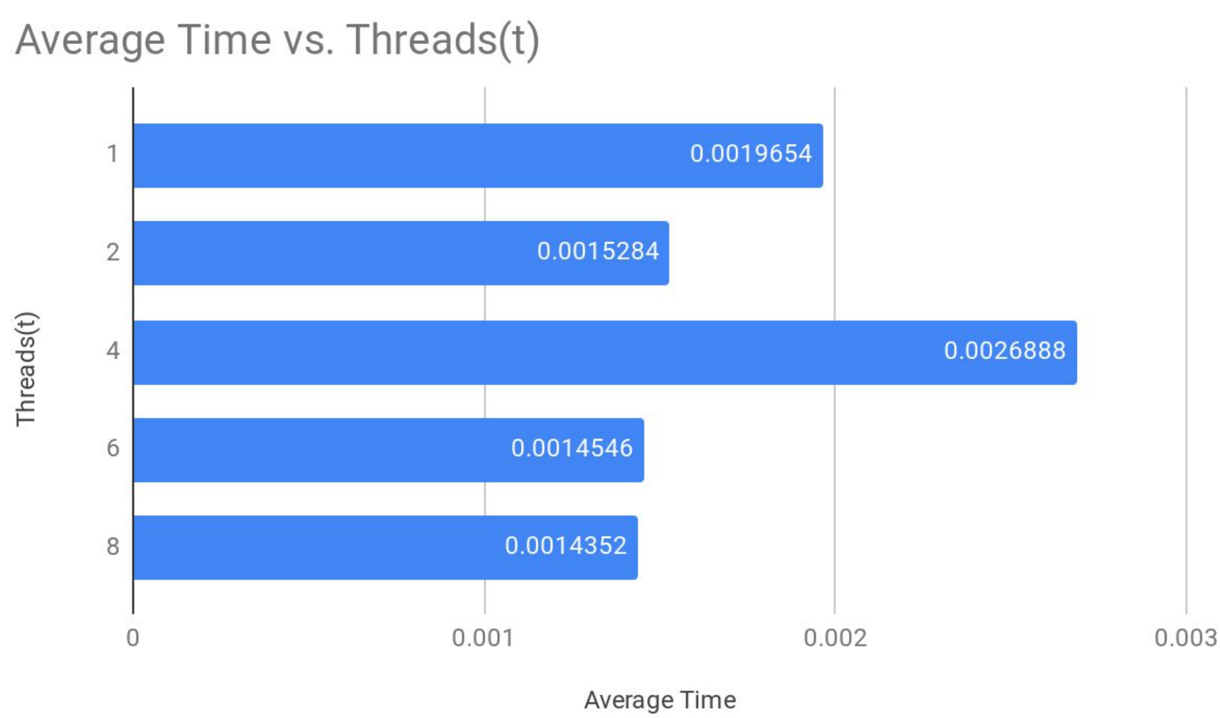
| Threads(t) | Run1 | Run2 | Run3 | Run4 | Run5 | Average Time |
|---|---|---|---|---|---|---|
| 1(Serial Code) | 0.001747 | 0.001713 | 0.0017 | 0.001715 | 0.001748 | 0.0017246 |
| 2 | 0.000679 | 0.001096 | 0.001086 | 0.001101 | 0.00109 | 0.0010104 |
| 4 | 0.001874 | 0.003504 | 0.002114 | 0.00156 | 0.000803 | 0.001971 |
| 6 | 0.001075 | 0.001177 | 0.001331 | 0.001244 | 0.001227 | 0.0012108 |
| 8 | 0.001188 | 0.000876 | 0.000961 | 0.001218 | 0.001203 | 0.0010892 |




While the parallelised code takes less time than serial for all cases except t = 4. As the number of thread increases above 4 however, we can see a downward trend of average time due to more efficient work division.

b) We compute the integral using Monte Carlo method for n = 10000 and different number of threads -

| Threads(t) | Run1 | Run2 | Run3 | Run4 | Run5 | Average Time |
|---|---|---|---|---|---|---|
| 1(Serial Code) | 0.002115 | 0.002007 | 0.001966 | 0.001992 | 0.001747 | 0.0019654 |
| 2 | 0.001388 | 0.001653 | 0.001437 | 0.001544 | 0.00162 | 0.0015284 |
| 4 | 0.002921 | 0.00279 | 0.002688 | 0.002857 | 0.002188 | 0.0026888 |
| 6 | 0.001453 | 0.000954 | 0.001591 | 0.001638 | 0.001637 | 0.0014546 |
| 8 | 0.001351 | 0.001454 | 0.001542 | 0.001435 | 0.001394 | 0.0014352 |




While the parallelised code takes less time than serial for all cases except t = 4. As the number of thread increases above 4 however, we can see a downward trend of average time due to more efficient work division.