CENG 371 - Scientific Computing
Fall 2022
Homework 1

Anıl Utku Ilgın
2260073

October 30, 2022

---

1. (a) Plot of g(n) is given below 1

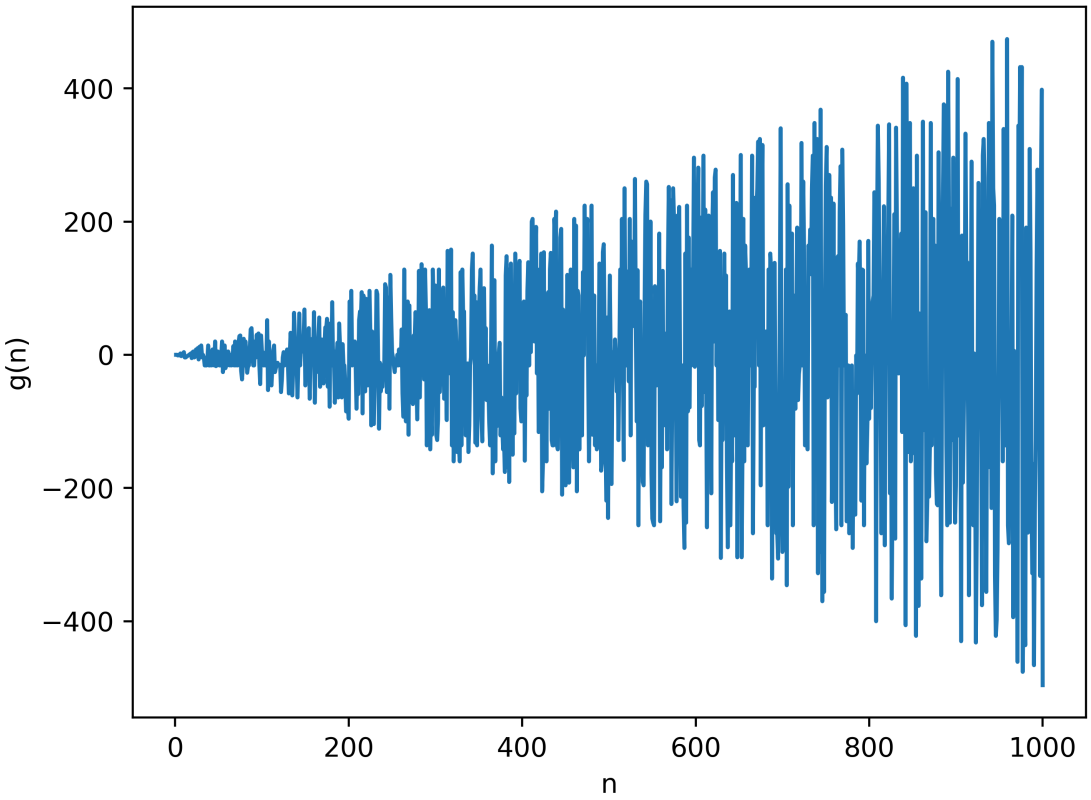

Figure 1: Plot of g(n)

(b) The values that makes g(n) = 0 are
2^n where $n \epsilon N < 1000$
$= \{1, 2, 4, 8, 16, 32, 64, 128, 256, 512\}$

(c) For values of n which are powers of 2, there are no representation error for them and since multiplication and division to powers of 2 are done by bit shifting in computers, there are no operational errors either. Hence, for values beside these, there could be representational or operational errors and thus g(n) would be other then zero because of these errors.

(d) In the f(n), there is a small error for values other than $n = \{1, 2, 4, 8, 16, 32, 64, 128, 256, 512\}$ and as n grows, the cancellation error in $n/(n-1) - 1$ part doesn't decrease as in same rate, resulting in a growing error in size.

2. (a)

$$\begin{aligned}\textbf{nums}[n] &= 1 + (10^6 + 1 - n) * 10^{-8} \\ &= \sum_{n=1}^{10^6} 1 + 10^{-2} + 10^{-8} - n * 10^{-8} \\ &= 10^6 + 10^4 + 10^{-2} - 10^{-8} * \frac{1 + 10^6}{2} \\ &= 1005000.005\end{aligned} \tag{1}$$

(b) Using divide and conquer algorithm, pairwise summation divides the arrays into smaller children arrays until a certain length is reached. Later then sums up the child array, and in return summing the result of this array with other child array, pairwise summation follows the recursion until the top.

(c) Naive sum of single precision is 1002466.6875
Naive sum of double precision is 1005000.0049999995

Compensated sum of single precision is 1005000.0
Compensated sum of double precision is 1005000.005

Pairwise sum of single precision is 1005000.0
Pairwise sum of double precision is 1005000.005

(d) The naive-sum error for S-P is 2544.3125
The naive-sum error for D-P is 0.0000000005
The compensated-sum error for S-P is 5.0
The compensated-sum error for D-P is 0.0
The pairwise-sum error for D-P is 5.0
The pairwise-sum error for S-P is 0.0
The compensated sum and pairwise sum errors are smaller than the naive sum and equal to each other, however if I increased the N value of the pairwise summation the error increase, hence the compensated sum is much more reliable.

The naive sum runtime is on average 0.09 sec,
The compensated sum runtime is on average 0.24 sec,
The pairwise sum runtime is on average 1.10 sec
The naive and compensated sum is much faster than the pairwise sum since they are $O(n)$ complexitiy whereas the pairwise is $O(nlogn)$

3. The errors of single precision is much higher than the double precision versions, because of the precision errors. Single precision can represent up to 6 or 7 significant digits, therefore when dealing with numbers such as 10^-8 it's high likely for errors to occur.

For the naive-sums, we simply add up all the numbers in an iterative manner. Therefore, our errors are also accumulating during the summation, hence the error is enormous considering the single precision's precision loss, resulting in ~ 2500.
However, when we change the precision to double precision, we only encounter a really small error, ~ 0.0000000005 which is probably caused summation operations error.

For the pairwise summation, since we use divide and conquer error, our summation operations are much less numerous than the naive-summation. Therefore, our errors are also really low even in single precision version. ~ 5 difference and exact precision in the double representation shows that. The error also seems to increase if we increase the array size of the summation part, hence it's not a reliable one compared to compensation algorithm.

For the compensated sum, since we eradicate the error of summation with a compensation variable, we only encounter the representation errors in this version. Therefore, a ~ 5 error in single precision, and exact precision in the double precision version yields a very good result with this algorithm.

For the time complexity,
Since summation and compensated sum algorithm is $O(n)$ complexity, they are the smaller one. However, since we do additional substraction and addition operations in compensated sum, it's slower than the naive-summation.
Since pairwise summation follows the divide and conquer algorithm, it has $O(nlogn)$ complexity, and hence it's much more slower than the other algorithms.