CS206P Assignment #2 - squares and square roots

Those who submit their write-up written in LaTeX will receive a 10% bonus over those who hand-write their solutions.

Please submit both the write-up (PDF or scan of hand-written) and your code (including LaTeX source if you used LaTeX) on openlab using the submit command as previously described.

1.a) Analyze the error propagation when computing the **square** of a number. That is, assume t is a real number represented as $fl(t) = t \cdot (1 + \delta_0) = x_0$ in a floating point system. We would like to compute the square $f(t) = t^2$ in the floating-point system. In practice, we compute $f(x_0)$. Find an upper bound for the absolute value of the relative error in $f(x_0)$. Note that the relative error in $f(x_0)$ does not include any error in the representation of $f(x_0)$.

1.b) Let δ_k denote the relative error in squaring t, k times in a floating-point system, ie $x_k = t^{(2^k)}(1+\delta_k)$, where $x_k = fl(f(x_{k-1}))$. Show that $|\delta_k| \leq 2|\delta_{k-1}| + E$ where E is the machine epsilon. Then show by induction that $|\delta_k| \leq 2^k|\delta_0| + (2^k - 1)E$. Note that the error δ_k for $k > 0$ is due to both the computation and the representation of the result.

2.a) Analyze the error propagation when computing the **square root** of a number. That is, do the same as in 1.a) but with $f(t) = \sqrt{t}$.

2.b) Let δ_k denote the relative error in taking the square root of t, k times in a floating-point system. That is, $x_k = t^{1/2^k}(1 + \delta_k)$ where $x_k = fl(f(x_{k-1}))$. Show that $|\delta_k| \leq \frac{1}{2}|\delta_{k-1}| + E$ where E is the machine epsilon. Then show by induction that

$$|\delta_k| \leq \frac{1}{2^k}|\delta_0| + (2 - \frac{1}{2^{k-1}})E.$$

3. Given some value of x_0 and some value of y_0 we may for some positive integer N define the finite sequences

$$(i) \quad x_k = x_{k-1}^2, \quad k = 1, \ldots, N \tag{1}$$
$$(ii) \quad y_k = \sqrt{y_{k-1}}, \quad k = 1, \ldots, N \tag{2}$$
$$\tag{3}$$

Consider the following two experiments, where α is assumed to be an arbitrary small positive (real) number that is larger than the machine epsilon.

Experiment 1: Set $x_0 = t = 1 + \alpha$ and compute x_N by applying (i) N times. Then set $y_0 = x_N$ and compute y_N by applying (ii) N times.

Experiment 2: Reverse the order of (i) and (ii): Set $y_0 = t = 1 + \alpha$ and compute y_N by applying (ii) N times; then set $x_0 = y_N$ and compute x_N by applying (i) N times.

The mathematical result is expected to be the same in both cases: you should end up back at $t = 1 + \alpha$, but the computed results are likely to be different due to machine precision issues.

3.a) Set $\alpha = 2.37 \times 10^{-7}$ and use double-precision in your favourite language. Try both the above experiments for N ranging all values 1 through 30. For the final values of y_N and x_N in experiments 1 and 2, respectively and for every value of N, compute the error $y_N - t$ and $x_N - t$. Tabulate the results so that each line in the table corresponds to a value of N. Comment on how the error behaves during each experiment as N increases.

3.b) Let $\delta_k^{(i)}$ denote the relative error in x_k, and let $\delta_k^{(ii)}$ denote the relative error in y_k, and δ_0 denote the relative error in the representation x_0 of $t = 1 + \alpha$. Use the results of question **1.** above to obtain bounds for the errors $y_N - t$ and $x_N - t$ in the two experiments. Comment on whether these bounds agree with what was observed in 3.a.

Note: in 3.b), x_k and y_k are floating point numbers. Errors $\delta_k^{(i)}$ and $\delta_k^{(ii)}$ for $k > 0$ are due to both the respective computations and their representations.