CPSC 302, Fall 2018
Assignment 2, due Thursday, September 27, 23:59

Instructions: Read Chapters 2 and 3 (i.e., *Roundoff Errors* and *Nonlinear Equations in One Variable*) of the textbook as background for this assignment. Submission guidelines can be found at http://www.cs.ubc.ca/ dhavide/cpsc302/Assignments.html`Assignments.htm` on the course web site.

1. (a) The real number

$$x = \frac{653}{7} = 93.\overline{285714} = 93.28571428571429\ldots$$

has no exact representation in any decimal floating point system ($\beta = 10$) with finite precision t. Is there a finite floating-point system (i.e., some finite integer base β and precision t) in which this number does have an exact representation? If yes, then describe such a system and express x in the form $x = (d_0.d_1d_2\ldots d_t)_\beta \times \beta^e$ for a suitable number of digits t and a suitable exponent e.

Yes, it can be represented in a finite-floating point system, where the base is 7. This is because the number 653 is divided by 7.

$$x = \left(\left(\frac{1}{7^0} + \frac{6}{7^1} + \frac{2}{7^2} + \frac{2}{7^3}\right) * 7^2\right) \tag{1}$$

Where the base $\beta = 7$, precision $t = 4$ and $e = 2$. Precision 8 includes the repeating digits following the decimal point $93.\overline{285714}$. Therefore, $\frac{653}{7}$ $= (1.622)_7$.

(b) Answer the same question for the golden mean $\phi = (1 + \sqrt{5})/2$, i.e., is there a finite floating-point number system in which ϕ has an exact representation? If so, describe it and express ϕ appropriately.

No, there is no finite floating-point number system in which the golden mean has an exact representation. This is because the $\sqrt{5}$ in the $(1+\sqrt{5})/2$ is an irrational number, which has infinitely many digits with no pattern after the decimal point. Therefore, it is not possible to represent it as a combinations of ratio of two numbers.

2. In statistics, we often need to compute the *mean* $\bar{x}$ and the *variance* s_1^2 of a set $\{x_k\}_{k=1}^n$ of n numerical values (e.g., observations of a random variable). The

mean and variance are given by the respective formulas

$$\bar{x} = \frac{1}{n}\sum_{k=1}^{n} x_k, \text{ and} \tag{2a}$$

$$s_1^2 = \frac{1}{n}\sum_{k=1}^{n}(x_k - \bar{x})^2. \tag{2b}$$

In many instances, we imagine that n—the number of samples—is large, say, $n = 10,000$.

An equivalent formula for computing the variance s_1^2 is

$$s_1^2 = \frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2. \tag{2c}$$

Notice the summands in (2b) depend on $\bar{x}$; as such, it must be computed beforehand and two passes (i.e., loops) through the data are required. Computing s_1^2 using (2b) requires n subtractions, n multiplications (squares), $n-1$ additions, and one division by n. That is, the total number of flops required to compute s_1^2 using (2b) is $3n$ (not counting the computation of $\bar{x}$).

By contrast, using (2c) to compute s_1^2 requires n multiplications for squaring x_i, $n-1$ additions, one multiplication for squaring $\bar{x}$ and one subtraction. So the total number of flops here is $2n+1$. Hence, (2c) computes s_1^2 more cheaply than (2c) and requires only a single pass through the data (i.e., a single loop can be used to accumulate $\bar{x}$ and the summation in (2c)

(a) Prove that the right-hand sides of (2b) and (2c) are in fact algebraically equivalent.

$$\frac{1}{n}\sum_{k=1}^{n}(x_k - \bar{x})^2 = \frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2 \tag{3}$$

$$\frac{1}{n}\sum_{k=1}^{n}(x_k^2 - 2x_k\bar{x} + \bar{x}^2) = \frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2 \tag{4}$$

$$\frac{1}{n}\sum_{k=1}^{n} x_k^2 - \frac{2}{n}\sum_{k=1}^{n} x_k\bar{x} + \frac{1}{n}\bar{x}^2\sum_{k=1}^{n} 1 = \frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2 \tag{5}$$

$$\frac{1}{n}\sum_{k=1}^{n} x_k^2 - \frac{2}{n}\sum_{k=1}^{n} x_k\bar{x} + \frac{1}{n}\bar{x}^2 n = \frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2 \tag{6}$$

Since we know from (2a) that $\bar{x} = \frac{1}{n}\sum_{k=1}^{n} x_k$, we can substitute.

$$\frac{1}{n}\sum_{k=1}^{n} x_k^2 - 2\bar{x}^2 + \bar{x}^2 = \frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2 \tag{7}$$

Therefore,

$$\frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2 = \frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2 \tag{8}$$

(b) Which of the two methods from (2b) and (2c) do you expect to give more accurate results for computing s_1^2 in general?

(2b), the first formula, would yield more accurate results for computing the variance than (2c), the second formula. For (2c), the computation of the difference in squares can lead to cancellation errors and roundoff errors, because we are subtracting two numbers that are potentially very close to one another. This would lead to the loss of significant figures, which can yield inaccurate results. In some cases, this may result in negative values of variance. Therefore, the first formula should be used for obtaining more accurate results.

(c) Give a small example, using a decimal system with precision $t = 2$ and numbers of your choice, to validate your claims in part (b).

For example, we can calculate the variance of two samples, $x_1 = 2.8$ and $x_2 = 3.2$, with precision $t = 2$ and where the mean $\bar{x} = 3.0$

For the the first formula:

$s_1^2 = \frac{1}{n}\sum_{k=1}^{n}(x_k - \bar{x})^2$.

$s_1^2 = \frac{1}{2.0}((2.8 - 3.0)^2 + (3.2 - 3.0)^2)$.

$s_1^2 = \frac{1}{2.0}((-0.20)^2 + (0.20)^2)$.

$s_1^2 = \frac{1}{2.0}((0.040) + (0.040))$.

$s_1^2 = \frac{1}{2.0}(0.080)$.

$s_1^2 = 0.040$

For the the second formula:

$s_1^2 = \frac{1}{n}\left(\sum_{k=1}^{n} x_k^2\right) \; - \; \bar{x}^2$

$s_1^2 = \frac{1}{2.0}(2.8^2 + 3.2^2) - (3.0^2).$

$s_1^2 = \frac{1}{2.0}(7.8 + 10) - 9.0.$

$s_1^2 = \frac{1}{2.0}(18) - 9.0$

$s_1^2 = 9.0 - 9.0$

$s_1^2 = 0.0$

The variance of 2.8 and 3.2 is in fact 0.040, and therefore the first formula is more accurate.

3. Consider three algebraically equivalent polynomial functions:

$$\begin{aligned} f_1(x) &= x(x(x(x(x(x(x(x - 18) + 144) - 672) + 2016) \\ &\qquad - 4032) + 5376) - 4608) + 2304) - 512. \qquad (9a) \\ f_2(x) &= (x-2)^9, \text{ and} \qquad (9b) \\ f_3(x) &= x^9 - 18x^8 + 144x^7 - 672x^6 + 2016x^5 - 4032x^4 + 5376x^3 \\ &\qquad - 4608x^2 + 2304x - 512. \qquad (9c) \end{aligned}$$

   (a) Write a MATLAB script employing *three distinct methods* to evaluate the function:

      i. First, by applying nested evaluation (c.f. Example 1.4) for evaluating the polynomial $f_1(x)$ in the nested form (9a);

      ii. second, by calculating $f_2(x) = (x-2)^9$ from (9b) directly;

      iii. and finally, using $f_3(x)$ (the expression using monomial powers in (9c).

   In all three cases, you will sample the function at equidistant points on the closed interval $[1.94, 2.08]$ spaced 10^{-3} apart.

   Plot the resulting curves in three separate figures.

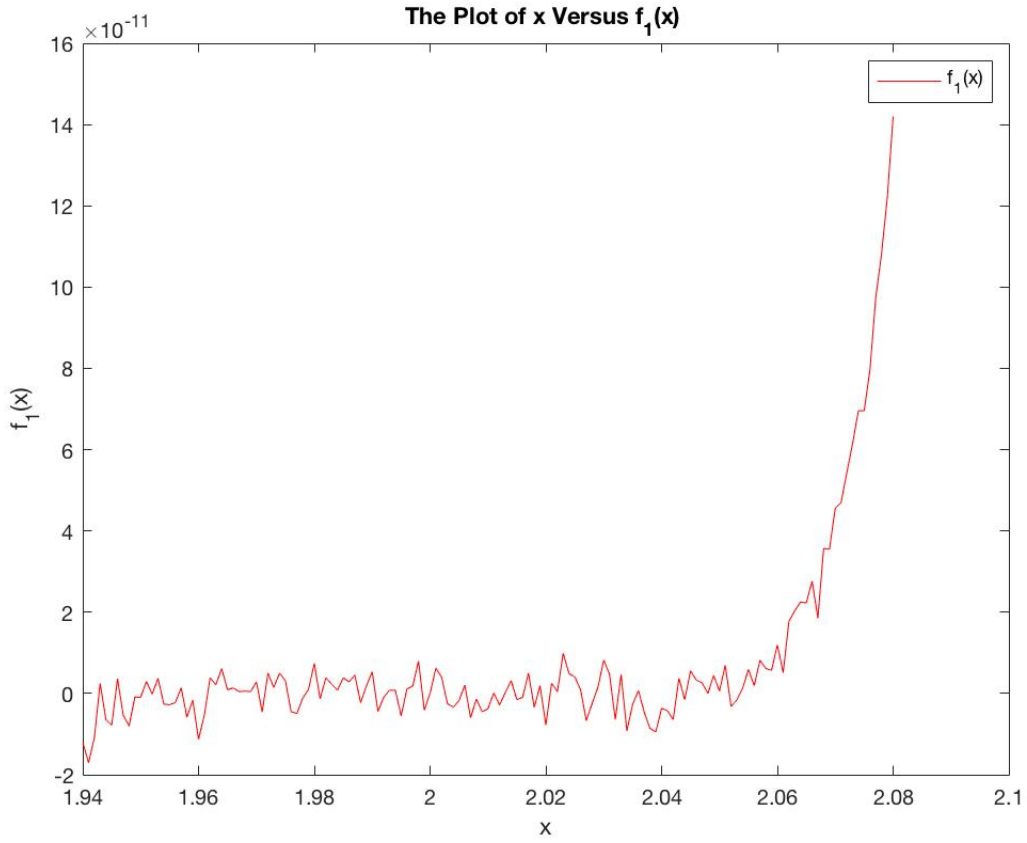

Figure 1: This is the plot of $f_1(x)$. We can see small fluctuations around y = 0.

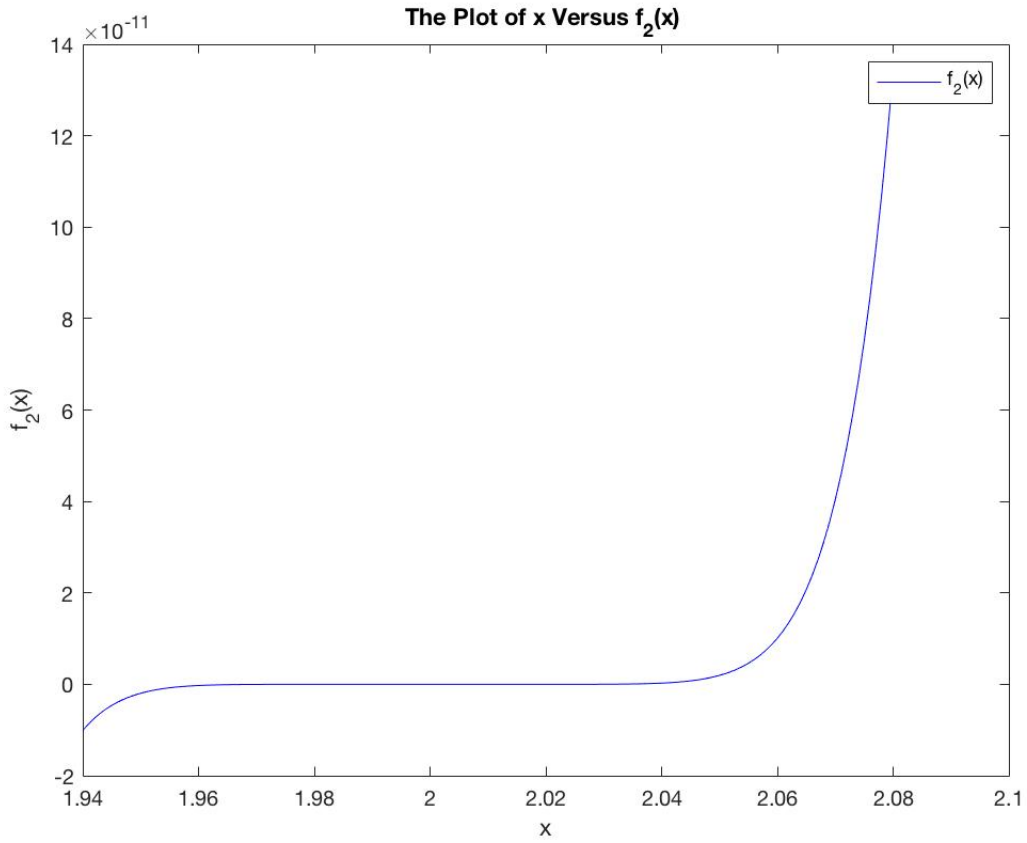

Figure 2: This is the plot of $f_2(x)$. It shows a smooth curve that crosses the y axis at exactly one point.

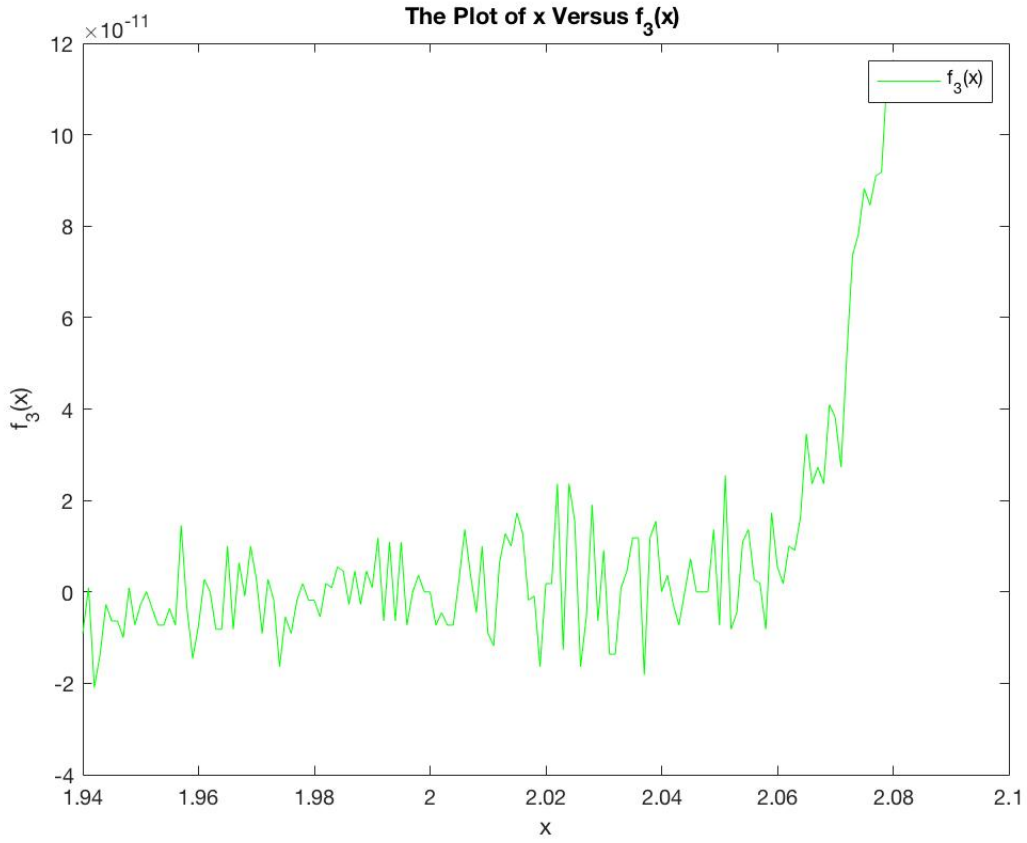

Figure 3: This is the plot of $f_3(x)$. There are large oscillations around y = 0.

(b) Explain the differences between the three graphs in *one short paragraph.*

The graphs for $f_1(x)$ and in particular $f_2(x)$, exhibits an oscillatory behavior, whereas $f_3(x)$ is a smooth curve.

The reason for this difference is due to errors. For $f_1(x)$, due to the nature of using nested loops, two values that are relatively close to each other are subtracted near the end of the evaluation multiple times. This results in a large relative error due to the loss of significant digits. The reason why the graph for $f_3(x)$ has even larger oscillations is because of the subtraction of numbers with large values, such as the one with exponents. This would lead to the loss of significant figures, and therefore would result in a very large relative error. The graph for $f_2(x)$ is smooth because the errors are minimized. Subtraction of $x - 2$ results in a small number ≤ 1. Power of numbers less than one results in even smaller numbers.

(c) Suppose you were to apply the bisection routine with an absolute tolerance 10^{-6} to find a root of the nested polynoimial function $f_1(x)$, starting from the interval $[1.96, 2.08]$. *Without computing anything*, select the correct outcome:

i. The routine will terminate with a root p satisfying $|p - 2| \leq 10^{-6}$.

ii. The routine will terminate with a root p *not* satisfying $|p - 2| \leq 10^{-6}$.

iii. The routine will not find a root.

Justify your choice in one short sentence.

The outcome will be ii. The routine will terminate with a root p *not* satisfying $|p - 2| \leq 10^{-6}$.

The graph for $f_1(x)$ shows multiple, noisy oscillations around the y-axis, and since the bisection method finds the root by looking at the difference in signs, and therefore it would most likely output a false root, which is not necessarily going to be the correct root, and thus it would not satisfy the absolute tolerance.