Lab Session 2

MA581 : Numerical Computations Lab S. Bora August 10, 2021

Switch to `format long e` for experiments 2-4.

1. The purpose of this exercise is to illustrate anomaly in automatic computation. On my computer, MATLAB produces

$$\begin{aligned}(\frac{4}{3}-1)*3-1 &= -2.2204\times 10^{-16}\\ 5\times\frac{(1+\exp(-50))-1}{(1+\exp(-50))-1} &= \textbf{NaN}\\ \frac{\log(\exp(750))}{100} &= \textbf{Inf}\end{aligned}$$

   Try on your machine. Can you explain the reason behind these anomalies?

2. Suppose that you are trying to approximate the values of each of the functions

   (i) $\dfrac{\tan x - x}{x^3}$ (ii) $\dfrac{e^x+\cos x-\sin x-2}{x^3}$ (iii) $\dfrac{e^x-1}{x}$

   at $x=0$ by evaluating them at $x=10^{-p}$, $p=1,\ldots,16$.

   a. Find the smallest value of p among $p=1,\ldots,16$ for which the expression calculated in double precision arithmetic at $x=10^{-p}$ has no correct significant digits. (Hint: First find the exact answer by taking the limit of each function as $x\to 0$. Then find the number of correct significant digits by taking the log of the absolute error in base 10. Use `log10` for this.)
   b. Rewrite each function so that the number of correct significant digits improves as the value of p increases.
   c. In a single graph, make a plot that shows the absolute error in log scale for the given function and the one for the rewritten version of the function on the y-axis against the values of $\log10(x)$ on the x-xis.

      Use `semilogy` and `hold on` commands to produce the graph. Also use the `legend` command to distinguish between the plots.

3. The following exercise illustrates that addition is not necessarily associative in finite precision environments.

   Given $n\in\mathbb{N}$, the built in MATLAB function `round` may be used to round $1/n$ to k significant digits and to simulate the summing of a finite number of terms of the sequence $\{\frac{1}{n}\}$, say the first m, in k-digit arithmetic. (Type `help round` for details)

   Use the `round` function to write a function program `[s, scf, scb] = sumreciprocal(m, k)` to return the sum of the first m terms of the sequence $\frac{1}{n}, n\in\mathbb{N}$, as `s`, the sum of the first m terms in 'k' digit arithmetic as `scf` and finally the same sum in reverse order, that is, from $\frac{1}{m}$ to 1 in 'k' digit arithmetic as `scb`.

   Now calculate the following:

   (a) Sum up $1/n$ for $n=1,2,\ldots,10^3$.

(b) Round each number $1/n$ to 5 digits and simulate the summing of the resulting sequence for $n = 1, 2, \ldots, 10^3$ in 5-digit arithmetic.

(c) Sum up the same chopped (or rounded) numbers in (b) again in 5-digit arithmetic but in reverse order, that is, for $n = 10^3, \ldots, 2, 1$.

Compare the three computed results. Which among (b) and (c) is closer to (a)?

4. *The purpose of the following exercises is to illustrate the immense difficulties in handling polynomials in finite precision computation.*

   a. Write a MATLAB function program $\mathtt{y} = \mathtt{Horner(p, x)}$ which uses Horner's rule (see `Note` below) to evaluate a polynomial $p(z) = p_1 z^n + p_2 z^{n-1} + \cdots + p_n z + p_{n+1}$ at $z = x$ by taking $x \in \mathbb{R}$ and a vector p with $p(i) = p_i, i = 1 : n + 1$ as input. Design your program in such a way that if the user passes x as a vector (say in $\mathbb{R}^m$), then the output is a vector $y \in \mathbb{R}^m$ satisfying $y(i) = p(x(i)), i = 1 : m$.

   b. Write a MATLAB program $\mathtt{x} = \mathtt{bisect[p, x_0, x_1, tol]}$ which finds a root of a polynomial $p(x) = \Sigma_{k=0}^{m} a_k x^k$ (provided in terms of a vector p with $p(i) == a_i$) in an interval $[x_0, x_1]$ (where $p(x_1)p(x_2) < 0$) up to a given tolerance `tol` using the Bisection method. You should use the function `Horner` to evaluate the polynomial $p(x)$ at a point x.
   Now apply your algorithm to find a root of the polynomial

   $$p(x) = (x-2)^9 = x^9 - 18x^8 + 144x^7 - 672x^6 + 2016x^5 - 4032x^4 + 5376x^3 - 4608x^2 + 2304x - 512.$$

   in different intervals $[x_0, x_1]$ that lie within $[1.95, 2.05]$. What do you observe?

   c. This exercise should help you to understand the observations made in the previous exercise.
   Evaluate the polynomial p introduced in the previous exercise at 151 equidistant points (use `linspace` command) in the interval [1.93, 2.08] using the program $\mathtt{y} = \mathtt{Horner(p, x)}$. Then evaluate p for the same points by directly using the formula $p(x) = (x - 2)^9$.
   [If $\mathtt{x} = \mathtt{linspace(1.93, 2.08, 151)}$, then a vector $\mathtt{z}$ for which $z(i) = p(x_i)$, $i = 1 : 151$ is obtained via the command $\mathtt{z = (x - 2).\hat{}9}$.]
   Plot the graphs for both the procedures in the same figure. (Use the `hold on` command and different colors to distinguish between the plots).
   Do the plots differ from one another? If yes, can you think of possible reasons? Explain the results obtained in part (b) in the light of the difference in the plots.

Write a report of your experiments that covers all questions raised above as a pdf file. Include all tables and figures generated in your experiments in the report. Submit the report and all the required function programs in a single folder named as Rollnumber-MA581-Lab2.

Note 1 *(***Horner's method***)* `Given` $p(z) = \Sigma_{k=0}^{n} p_{n-k+1} z^k$, `the Horner's method uses the fact that`

$$p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1} = p_{n+1} + x(p_n + \cdots + x(p_3 + x(p_2 + p_1 x)))$$

`to evaluate` $p(z)$ `at` $z = x$.

*** End ***