AMSC 660 Homework 1
Due 09/06/23
By Marvyn Bailly

Problem 1

Starting with the declarations

```
float x, y, z, w;
const float oneThird = 1/ (float) 3;
const float oneHalf = 1/ (float) 2;
// const means these never are reassigned
```

we do lots of arithmetic on the variables `x, y, z, w`. In each case below, determine whether the two arithmetic expressions result in the same floating point number (down to the last bit) as long as no `NaN` or `inf` values or denormalized numbers are produced.

(a)
```
(x * y) + (z - w)
(z - w) + (x * y)
```

(b)
```
(x + y) + z
x + (y + z)
```

(c)
```
x * oneHalf + y * oneHalf
(x + y) * oneHalf
```

(d)
```
x * oneThird + y * oneThird
(x + y) * oneThird
```

Solution

Proof. Let $\epsilon_i > 0$ such that $|\epsilon_i| \leq \epsilon_m \ \forall i$.

(a) Consider the floating point arithmetic:

```
(x * y) + (z - w)
(z - w) + (x * y)
```

Observe

$$\begin{aligned}((x*y)(1+\epsilon_1)+(z-w)(1+\epsilon_2))(1+\epsilon_3) &= w\left(-\epsilon_2\right) - w\epsilon_2\epsilon_3 - w\epsilon_3 - w + xy\epsilon_1 \\ &\quad + xy\epsilon_3 + xy\epsilon_1\epsilon_3 + xy + z\epsilon_2 + z\epsilon_3 + z\epsilon_2\epsilon_3 + z \\ &= -w\left(\epsilon_2+\epsilon_3+1\right) + xy\left(\epsilon_1+\epsilon_3+1\right) + z\left(\epsilon_2+\epsilon_3+1\right) \\ &= -w(\eta_1+1) + xy(\eta_2+1) + z(\eta_3+1),\end{aligned}$$

where the $\epsilon_i * \epsilon_j$ terms are dropped as they are sufficiently small[1] and $\eta_1 = \epsilon_2 + \epsilon_3, \eta_2 = \epsilon_1 + \epsilon_3,$ and $\eta_3 = \epsilon_2 + \epsilon_3$. Notice that

$$\begin{aligned}((z-w)(1+\epsilon_2)+(x*y)(1+\epsilon_1))(1+\epsilon_3) &= -w\left(\epsilon_2+\epsilon_3+1\right) + xy\left(\epsilon_1+\epsilon_3+1\right) + z\left(\epsilon_2+\epsilon_3+1\right) \\ &= -w(\eta_1+1) + xy(\eta_2+1) + z(\eta_3+1),\end{aligned}$$

Thus we see under floating point arithmetic

$$\boxed{(x*y)+(z-w) = (z-w)+(x*y).}$$

(b) Consider the floating point arithmetic:

```
(x + y) + z
x + (y + z)
```

Observe

$$((x+y)(1+\epsilon_4)+z)(1+\epsilon_5) = x\left(\epsilon_4+\epsilon_5+1\right) + y\left(\epsilon_4+\epsilon_5+1\right) + z\left(\epsilon_5+1\right),$$

and notice that

$$(x+(y+z)(1+\epsilon_6))(1+\epsilon_7) = x\left(\epsilon_7+1\right) + y\left(\epsilon_6+\epsilon_7+1\right) + z\left(\epsilon_6+\epsilon_7+1\right),$$

results in the x and z terms picking up different amounts of error. Thus we have that under floating point arithmetic

$$\boxed{(x+y)+z \neq x+(y+z)}$$

(c) Consider the floating point arithmetic:

[1] We will continue to drop such terms without reference.

```
x * oneHalf + y * oneHalf
(x + y) * oneHalf
```

where `const float oneHalf = 1/ (float) 2;` is an exact floating point number. Observe that

$$((x * \text{oneHalf})(1+\epsilon_8) + (y * \text{oneHalf})(1+\epsilon_9))(1+\epsilon_{10}) = (x * \text{oneHalf})(1+\epsilon_8+\epsilon_{10}) + (y * \text{oneHalf})(1+\epsilon_9+\epsilon_{10}),$$

and

$$((x+y)(1+\epsilon_{11}) * \text{oneHalf})(1+\epsilon_{12}) = (x * \text{oneHalf})(1+\epsilon_{11}+\epsilon_{12}) + (y * \text{oneHalf})(1+\epsilon_{11}+\epsilon_{12}).$$

We see that the result of the floating point arithmetic results in a floating point number whose terms pick up the same amount of error. Thus we have that under floating point arithmetic

$$\boxed{x * \text{oneHalf} + y * \text{oneHalf} = (x+y) * \text{oneHalf}}$$

(d) Consider the floating point arithmetic

```
x * oneThird + y * oneThird
(x + y) * oneThird
```

where `const float oneThird = 1/ (float) 3;` is not an exact floating point number and thus picks up error ϵ_{13}. Observe that

$$\begin{aligned}&((x * \text{oneThird}(1+\epsilon_{13}))(1+\epsilon_{14}) + (y * \text{oneThird}(1+\epsilon_{13}))(1+\epsilon_{15}))(1+\epsilon_{16})\\ &= x * \text{oneThird}\,(1+\epsilon_{13}+\epsilon_{14}+\epsilon_{16}) + y * \text{oneThird}\,(1+\epsilon_{13}+\epsilon_{15}+\epsilon_{16}),\end{aligned}$$

and

$$\begin{aligned}&((x+y)(1+\epsilon_{17}) * \text{oneThird}(1+\epsilon_{13}))(1+\epsilon_{18})\\ &= (x * \text{oneThird})(1+\epsilon_{13}+\epsilon_{17}+\epsilon_{18}) + (y * \text{oneThird})(1+\epsilon_{13}+\epsilon_{17}+\epsilon_{18}).\end{aligned}$$

Notice that the error terms on the $(x * \text{oneThird})$ is inconsistent between the two floating point arithmetics. Thus

$$\boxed{x * \text{oneThird} + y * \text{oneThird} \neq (x+y) * \text{oneThird}}$$

□

Problem 2

The *tent map* of the interval $[0, 1]$ onto itself is defined as

$$f(x) = \begin{cases} 2x, & x \in [0, 1/2), \\ 2 - 2x, & x \in [1/2, 1]. \end{cases}$$

Consider the iteration $x_{n+1} = f(x_n)$, $n = 0, 1, 2, \ldots$.

(a) What are the fixed points of this iteration, i.e. the points x^* such that $x^* = f(x^*)$? Show that these fixed points are unstable, i.e., if you start iteration at $x^* + \delta$ for any δ small enough then the next iterate will be farther away from x^* then $x^* + \delta$.

(b) Prove that if x_0 is rational, then the sequence of iterates generated starting from x_0 is periodic.

(c) Show that for any period length p, one can find a rational number x_0 such that the sequence of iterates generated starting from x_0 is periodic of period p.

(d) Generate several long enough sequences of iterates on a computer using any suitable language (Matlab, Python, C, etc.) starting from a pseudorandom x0 uniformly distributed on $[0, 1)$ to observe a pattern. I checked in Python and C that 100 iterates are enough. Report what you observe. If possible, experiment with single precision and double precision.

(e) Explain the observed behavior of the generated sequences of iterates.

Solution

Proof. Consider the iteration $x_{n+1} = f(x_n), n = 0, 1, 2, \ldots$ where

$$f(x) = \begin{cases} 2x, & x \in [0, 1/2), \\ 2 - 2x, & x \in [1/2, 1]. \end{cases}$$

(a) We wish to find the fixed points of the system. Observe that if $x^* \in [0, 1/2)$ then

$$x^* = f(x^*) = 2x* \implies x^* = 0.$$

Similarly, if $x^* \in [1/2, 0]$ then

$$x^* = f(x^*) = 2 - 2x^* \implies x^* = 2/3.$$

Thus there are two fixed points $x^* = 0, 2/3$. To determine the stability of the fixed points, let $|\delta| > 0$ be sufficiently small. Observe for $x^* = 0$ we get

$$f(x^* + \delta) = 2(x^* + \delta) = 2\delta.$$

Since $|x^* + \delta| = |\delta| < |2\delta| = |f(x^* + \delta)|$ we see that the fixed point $x^* = 0$ is unstable. Similarly, for $x^* = 2/3$ we get

$$f(x^* + \delta) = 2 - 2(x^* + \delta) = 2/3 - 2\delta.$$

Notice that $f(x^* + \delta)$ is 2δ away from x^* and thus the fixed point $x^* = 2/3$ is unstable.

(b) Let $x_0 = p/q$ for $p, q \in \mathbb{Z}$ such that $q \neq 0$ and $0 \leq p \leq q$ so that $0 \leq x_0 \leq 1$. Notice that if $x_0 > 1/2$ then

$$x_1 = f(x_0) = 2x_0 = \frac{2p}{q},$$

otherwise

$$x_1 = f(x_0) = 2 - 2x_0 = \frac{2q - 2p}{q}.$$

Note that only the numerator is affected in this and future updates. If we continue updating the numerator, two events must occur. Either the trajectory leads to a fixed point x^* and has period 1 or $x_n = p/q = x_0$ and has period n. Notice that $x_n = x_0$ must occur, as there are a finite number of integers between 0 and q that the numerator can become. That is on each update i if x_i does not lead to a fixed point, then the numerator is mapped to an integer r such that $0 < r < q$ [2] since $f : [0, 1] \to [0, 1]$ and the sum of integers are integers. If $r \neq p$ and the update doesn't result in a fixed point, the map updates the numerator again. Thus after a certain amount updates, let's say n, the numerator must return to the original integer p giving $x_n = p/q = x_0$. Since we have the original p/q, the tent map will update x_n the same way it updated x_0 giving a periodic trajectory of period n.

(c) To show that for any period length $p = 1, 2, \ldots,$ one can find a rational number x_0 such that the sequence generated is periodic of period p, consider

$$x_0 = \frac{2}{2^p + 1} < \frac{1}{2}.$$

Notice that

$$x_i = f(x_{i-1}) = \frac{2^{i-1}}{2^p + 1} < 1/2$$

for $i \leq p - 2$. Then

$$x_{p-1} = f(x_{p-2}) = \frac{2^p}{2^p + 1} > 1/2.$$

Thus the next term will be

$$x_p = f(x_{p-1}) = 2 - 2\left(\frac{2^p}{2^p + 1}\right) = \frac{2(2^p + 1) + 2^{p+1}}{2^p + 1} = \frac{2}{2^p + 1} = x_0.$$

In the case when $p = 1$, then $x_0 = 2/3$ which is a fixed point and has period 1. Therefore $x_0 = x_p$ which shows that $\forall p = 1, 2, \ldots, \exists x_0$ rational such that the generated sequence is periodic with period p.

[2] We use strict inequalities as $0/q$ and q/q lead to fixed points.

(d) I created an algorithm for the tent map in Visual Basic that prints the sequence to an Excel spreadsheet. While this method appeared to work for both single and double precision, the sequence would always converge to zero after around 25 iterations. So I also created an algorithm in Python where I observed that for double precision, the sequence would hit zero on average after around 50 iterations while for single precision it took around 25 iterations on average. In both cases, the sequence would always end by hitting $1/2 \to 1 \to 0$. Another pattern I observed is that the sequence usually jumps between $[0, 1/2)$ and $[1/2, 1]$ in a few of the following manners: switching back and forth at each iteration, spending a few iterations in $[0, 1/2)$ and then switching, or spending a few iterations near $2/3$ before switching. Code can be found at `https://github.com/MarvynBailly/AMSC660/tree/main/homework1`

(e) The pattern observed above is due to the round-off error of floating point numbers and the instability of floating point arithmetic. We don't achieve any periodic trajectory for two reasons, it is unlikely to have x_0 be a rational number from its construction and if x_n becomes a rational number, it will be perturbed. Also, notice that if we rewrite x_0 in binary, $2x$ will swap 0 bits into 1 and vice versa while $2 - 2x$ deletes the first entry of the mantissa and swap 0 and 1s. Thus every time the number $x_i \geq 1/2$, the floating-point number loses a value in the mantissa. Since in double precision, there are 52 mantissa bits and in single precision, there are 23, it makes sense that after that around 50 or 20 iterations, the floating-point number is rounded to a sequence that converges to 0 as it has lost all the significant entries in the mantissa.

□