Floating-point lab work

Florent de Dinechin

Ecole PRECIS, May 2017

1 Floating-point under the microscope

1.1 Representable or not representable?

- Decompress `TP-code.tgz`. Compile it with `gcc main.c`, and run it. How many decimal digits seem significant in a float and in a double?
- Try a few other values, in particular integers such as 7.0. Which integers are exactly representable as `float` ?
- Look at `fp-struct.h`. If you want to reimplement it in your favorite language, go ahead (but support is not guaranteed).

1.2 Exceptional numbers

- Implement the following loop:

```
float x = 0.1;
while(x!=0) {
  print_binary32(x);
  x=x/2;
}
print_binary32(x);
```

  Observe the apparition of subnormals.

- Replace the 0.1 with 1.0.
- Implement the same loop in Python (or your favorite scripting language), with a simple print of the value. Is it using binary32 or binary64?
- Replace the division by 2 with a multiplication by 2. What happens? Replace the while loop with a for loop of the right size.
- Construct a NaN and print it.

2 Solving the quadratic equation

To solve the quadratic equation $ax^2 + bx + c$, here are the formulas I learnt in school:

$$\delta = b^2 - 4ac$$

$$\text{if } \delta \geq 0, \quad r = \frac{-b \pm \sqrt{\delta}}{2a}$$

- Implement these formulas in C using floats on one side, and double on the other side. Square root is called `sqrtf` (float) or `sqrt` (double); It requires to `#include <math.h>` and add the `-lm` flag, e.g. `gcc main.c -lm`
- Compare the results for a=0.125; b=1000; c=1;
- Compare the results for a=0.125; b=10000; c=1;
- Explains what happens. Hint: look for possible cancellations.
- Enhance the code with a test that anticipates a possible cancellation and uses a different formula in this case.

3 About sums

The `sums.c` program implements various techniques to compute the sum of N integers.

- Test `TwoSum32` on 16,000,000 + 0.125, then on other values.
- The input array is composed of inverses of i rounded to floats, for $1 < i < N = 1,000,000$. Prove that for this input array, the sum computed in double is exact.
- Run the program. Vary N.
- Replace $1/i$ by $\cos(i)$: we now have numbers of different signs. Can you think of other strategies for the summation?