Numerical Analysis – An Overview

Bellenot

January 3, 2014

Contents

Introduction

One of the things that most scares me and other Mathematicans, both pure and applied, is how people, even smart people, over trust Calculators and Computers.

There are famous examples of broken computers and programs. Intel sold thousands of computers that couldn't multiply correctly. NASA has lost Mars probes. Bridges have fallen. Student grades have been miscalculated.

1 MUNG

MUNG is an acronym. It is a recursive acronym, one defined in terms of itself. So MUNG stands for "MUNG Until No Good" which in turn stands for "MUNG Until No Good Until No Good" and of course it keeps on going:
MUNG Until No Good Until No Good Until No Good Until No Good ...
or there is nothing that can screw stuff up faster than a misprogrammed computer.

Your program produced answer X to question Y. How do you convince your boss that X is a reasonable answer?

2 Error Analysis

The key to numerical approximations is Error Analysis. Saying π is 3 is a valid approximation, but not very accurate. Well it is accurate to one significant digit, but it is not very precise. Saying $\pi = 3.142857$ (roughly 22/7) is more precise but not as accurate as it is precise. A better approximation is $\pi = 3.141592654$.

Good algorithms have useful error estimates. Often these estimates require some theory from calculus and the estimates depend on extremal values of some (often higher) derivative of some function. Taylor's formula is the classic example:

$$f(x+h) = f(x) + f'(x)h + f''(x)\frac{h^2}{2} + f'''(x)\frac{h^3}{3!} + f^{(4)}(x)\frac{h^4}{4!} + \cdots + f^{(n)}(\xi)\frac{h^n}{n!}$$

The error term requires maximizing the n-th derivative of $f(x)$ on $[x, x+h]$

3 Loss of Significance

This is a negative feature of floating point arithmetic. If one subtracts 3.12 from 3.13, one gets 0.01. The notation on 3.12 and 3.13 implies three digits of significance. We have 3.13 trapped between 3.125 and 3.135 and 3.12 trapped between 3.115 and 3.125. So 0.01 is trapped between 0 and 0.02, we only have one digit of significance. Using scientific notation, 0.01 is 1E-2 and not 1.00E-2. There is no way around this problem.

But Numerical Analysis can find ways to avoid this problem by rearranging the calculation that arrived at these numbers. If b is large compared to ac in the quadratic formula, then $\sqrt{b^2-4ac}$ is nearly b so one of the roots is near zero.

$$\begin{aligned}
&\frac{-b+\sqrt{b^2-4ac}}{2a} \\
&= \frac{(-b+\sqrt{b^2-4ac})(-b-\sqrt{b^2-4ac})}{2a(-b-\sqrt{b^2-4ac})} \\
&= \frac{(b^2-b^2+4ac)}{2a(-b-\sqrt{b^2-4ac})} \\
&= \frac{2c}{-b-\sqrt{b^2-4ac}}
\end{aligned}$$

In this case, the first formula does have a loss of significance, but the last formula does not.

4 Hardware Addition is not Associative

Using three digits $1.23 + 0.0045$ still rounds to 1.23, but $1.23 + 0.0090$ rounds to 1.24. So using three digit arithmetic:

$$(1.23 + 0.0045) + 0.0045 = 1.23 \neq 1.24 = 1.23 + (0.0045 + 0.0045)$$

A rule of thumb is to add terms in order of increasing absolute value. In the case above, 1.24 is a better estimate than 1.23. Hardware multiplication is also not associative.

5 Other issues

Algorithms can have other problems. They might not work in all cases. Worst they might work, but give wrong answers in some cases. They might work all the time, but take too long run at times. Algorithms can even be so complex that no one has tried to write the code for them.

There are a large number of numerical libraries of varying quality available. How do you tell the good ones from the bad ones? It is often easy to code an algorithm fast that runs slow. But it is also a waste of time to code something that is very fast, when a slower code would do the job.

These are engineering decisions. Hence one needs practice to gain experience.