LECTURE 4

Floating Point Error Analysis

1. Absolute and Relative Errors

2. Relative Error Analysis

We now state two theorems regarding the propagation of round off errors for sums and products.

THEOREM 4.1. *Let $x_0, x_1, \ldots, x_n$ be positive machine numbers in a computer whose unit roundoff error is ε. Then the relative roundoff error of the sum*

$$\sum_{k=0}^{n} x_k$$

is at most $(1+\varepsilon)^n - 1 \approx n\varepsilon$.

2.1. Subtraction of Nearly Equal Quanities. As we saw in the preceding lecture roundoff errors are an inevitable consequence of the way computers carry out calculations. Another, but often avoidable, means of introducing large relative errors is by computing the difference between two nearly equal floating point numbers.

For example, let

$$\begin{aligned} x &= 0.3721478693 \\ y &= 0.3720230572 \\ x - y &= 0.0001248121 \end{aligned}$$

and suppose that the difference $x - y$ is computed on a decimal computer allowing a 5-digit mantissa (i.e., 5 significant figures)

$$\begin{aligned} fl(x) &= 0.37215 \\ fl(y) &= 0.37202 \end{aligned}$$

two numbers, each with 5 significant digits. When the machine computes the difference between $fl(x)$ and $fl(y)$ it obtains

$$fl(x) - fl(y) = 0.00013 = 1.3 \times 10^4$$

a number with only two significant digits. The relative error is thus fairly large

$$\left| \frac{x - y - [fl(x) - fl(y)]}{x = y} \right| = \left| \frac{0.000124812 - 0.00013}{0.000124812} \right| \approx 4\%$$

The following theorem gives bounds on the relative error that can be introduced by such subtraction errors.

THEOREM 4.2. ***(Loss of Precision Theorem.)*** *If x and y are positive normalized binary machine numbers such that $x > y$, and*

$$2^{-q} \leq 1 - \frac{y}{x} \leq 2^{-p}$$

then at most q and at least p significant binary digits will be lost in the subtraction $x - y$.

Proof. Let

$$\begin{aligned} x &= r \times 2^n \quad , \quad 1 \leq r < 2 \\ y &= s \times 2^m \quad , \quad 1 \leq s < 2 \end{aligned}$$

be the normalized binary floating point forms for x and y. Since x is larger than y, $m \leq n$. In order to carry out the subtraction, a computer will first have to shift the floating point of y so that the machine representations of x and y have the same number of decimal places; effectively writing y as

$$y = \left(s \times 2^{m-n}\right) \times 2^n$$

We then have

$$\times - y = \left(r - s \times 2^{m-n}\right) \times 2^n$$

The mantissa of this expression satisfies

$$r - s \times 2^{m-n} = r\left(1 - \frac{s \times 2^m}{r \times 2^n}\right) = r\left(1 - \frac{y}{x}\right) < 2^{-p}$$

To normalize this expression, expression then a shift of at least p digits is required. This means that at least p bits of precision have been lost. On the other hand, since the new mantissa also satisfies

$$r - s \times 2^{m-n} = r\left(1 - \frac{y}{x}\right) > 2^{-q}$$

then a shift of no more than q digits will be necessary to put the mantissa in standard form. Thus, at most q bits of precision have been lost.