1
Thurs
1/10

chapter 1 : finite precision arithmetic

number systems

decimal : $(2013)_{10} = 2 \cdot 10^3 + 0 \cdot 10^2 + 1 \cdot 10^1 + 3 \cdot 10^0$

binary : $(101.01)_2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = (5.25)_{10}$

question : how are numbers stored in a computer?

answer : floating point representation

$x = \pm(0.d_1 d_2 \cdots d_n)_\beta \cdot \beta^e$, n significant digits , $d_1 \neq 0$: normalization

β : base , d_i : digits (or bits if $\beta = 2$) , $0 \leq d_i \leq \beta - 1$

$(0.d_1 d_2 \cdots d_n)$: mantissa , e : exponent , $-M \leq e \leq M$

ex : consider a computer with $\beta = 2$, $n = 4$, $M = 3$

$x_{\max} = (0.1111)_2 \cdot 2^3 = (\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16}) \cdot 8 = 4 + 2 + 1 + \frac{1}{2} = 7.5$

$x_{\min} = (0.1000)_2 \cdot 2^{-3} = \frac{1}{2} \cdot \frac{1}{8} = 0.0625$

question : how many different numbers can be represented in this system?

answer : hw

note

1. In IEEE double precision format, each number is stored as a string of 64 bits.

±	±	mantissa = 52 bits	exponent = 10 bits

The first 2 bits are for the sign of the mantissa and exponent. Hence we have $\beta = 2, n = 52, M = (1111111111)_2 = 1 + 2 + 4 + 8 + \cdots + 2^9 = 2^{10} - 1 = 1023$.

2. If x is a number and $\text{fl}(x)$ is its floating point representation, then $x - \text{fl}(x)$ is the roundoff error.

ex 1

$\pi = 3.14159\ 26535\ 89793\ 23846\ldots$: exact value to 20 decimal digits

$\text{fl}(\pi) = 3.14159\ 26535\ 89793$: Matlab gives 15 decimal digits

$\Rightarrow$ roundoff error $= 0.23846\ldots \cdot 10^{-15} \approx 2^{-52}$

ex 2

$0.1234 - 0.1233 = 0.0001 = (0.1000)_{10} \cdot 10^{-3}$

$\Rightarrow$ the result has only 1 significant digit

Hence if 2 floating point numbers with n significant digits are subtracted, the result may have less than n significant digits; this is called loss of significance due to cancellation of digits.

ex 3 : quadratic formula

$$ax^2 + bx + c = 0 \;\Rightarrow\; x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad , \quad \text{pf} : \ldots$$

$0.2x^2 - 47.91x + 6 = 0 \;\Rightarrow\; x = 239.4247, 0.1253 :$ Matlab

Now use 4-digit arithmetic (i.e. each step is rounded to 4 digits, as if we're using a computer with $n = 4$).

$$x = \frac{47.91 \pm \sqrt{47.91^2 - 4(0.2)6}}{2(0.2)} = \frac{47.91 \pm \sqrt{2295 - 4.8}}{0.4} = \frac{47.91 \pm \sqrt{2290}}{0.4}$$

$$= \frac{47.91 \pm 47.85}{0.4} = \begin{cases} \dfrac{47.91 + 47.85}{0.4} = \dfrac{95.76}{0.4} = 239.4 \;:\; \text{all 4 digits are correct} \\ \dfrac{47.91 - 47.85}{0.4} = \dfrac{0.06}{0.4} = 0.15 \;:\; \text{only 1 digit is correct} \end{cases}$$

question : what caused the problem?

answer : loss of significance in the subtraction $47.91 - 47.85$

remedy 1 : Matlab (higher precision arithmetic)

remedy 2 : reformulate the arithmetic

$$x = \frac{-b - \sqrt{b^2 - 4ac}}{2a} \cdot \frac{-b + \sqrt{b^2 - 4ac}}{-b + \sqrt{b^2 - 4ac}} = \frac{b^2 - (b^2 - 4ac)}{2a(-b + \sqrt{b^2 - 4ac})} = \frac{2c}{-b + \sqrt{b^2 - 4ac}}$$

$$= \frac{2 \cdot 6}{47.91 + 47.85} = \frac{12}{95.76} = 0.1253 \;:\; \text{now all 4 digits are correct}$$

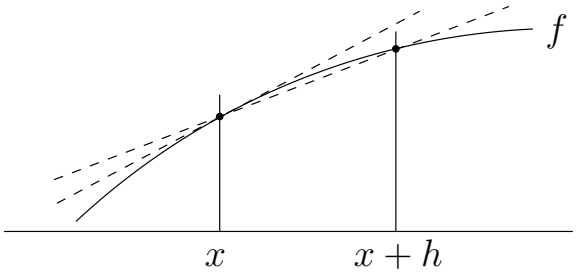
remedy 3 : solve for x using an iterative method (more later)

2
Tues
1/15

ex 4 : finite-difference approximation of a derivative

forward difference

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} = D_+ f(x)$$



question : how large is the error?

Taylor series : $f(x) = f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 + \cdots$

equivalent form :

$$\left.\begin{array}{l} x \to x+h \\ a \to x \end{array}\right\} \Rightarrow \quad f(x+h) = f(x) + f'(x)h + \tfrac{1}{2}f''(x)h^2 + \cdots$$

$$\Rightarrow \quad \frac{f(x+h) - f(x)}{h} = f'(x) + \underbrace{\tfrac{1}{2}f''(x)h + \cdots}$$

$$\uparrow \qquad \uparrow \qquad \uparrow$$

approximation — exact value — truncation error

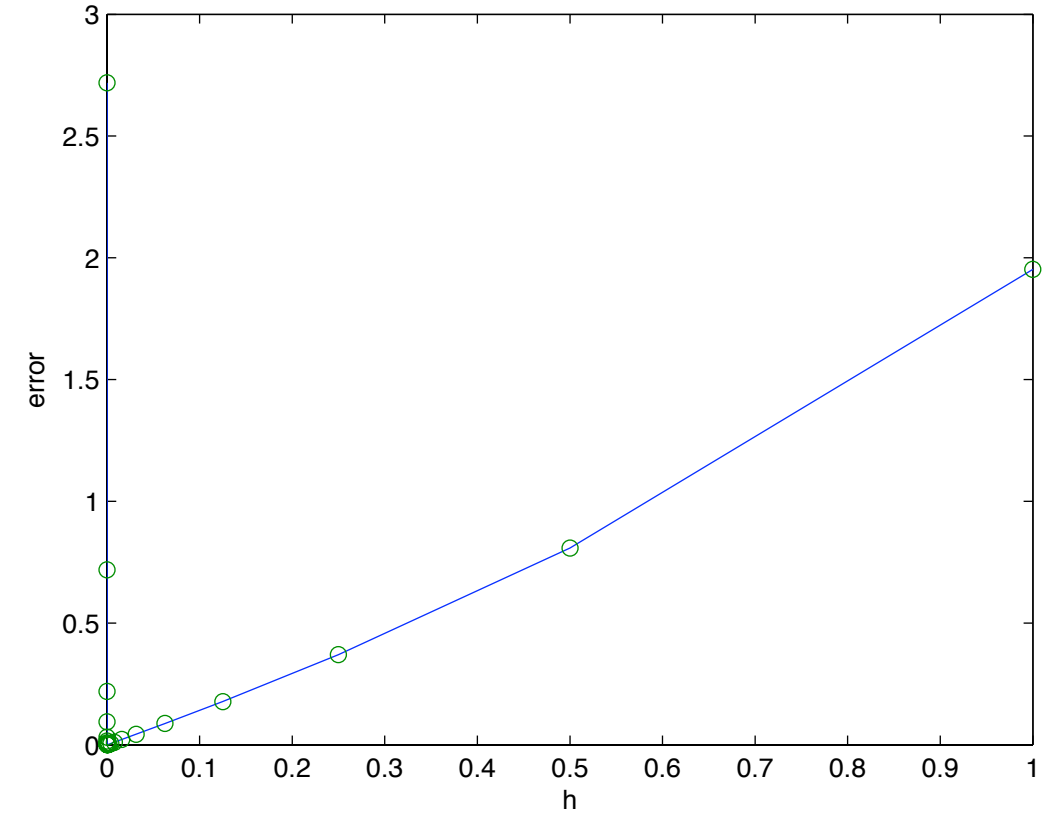
Hence the error is proportional to h; we write this as $D_+ f(x) = f'(x) + O(h)$, where the symbol $O(h)$ means "order h".

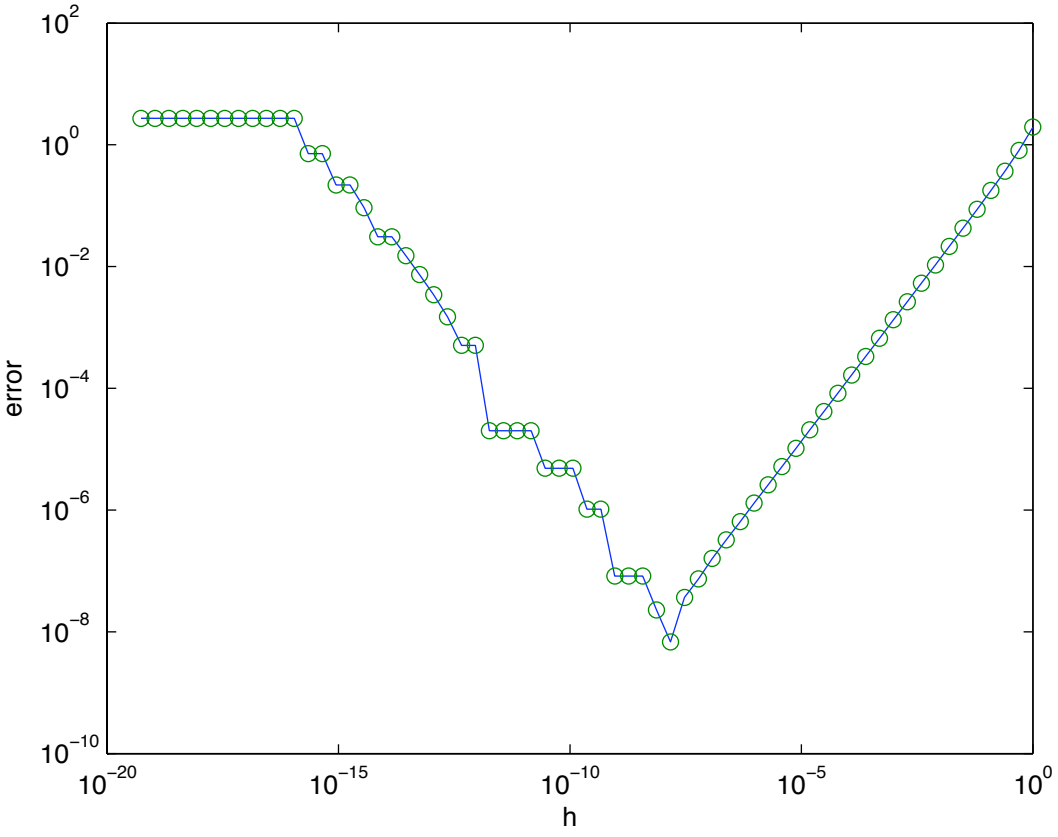
For example, if $f(x) = e^x, x = 1$, then $f'(1) = e = 2.71828\ldots$ is the exact value.

h	$D_+ f(1)$	$\lvert D_+ f(1) - f'(1)\rvert$	$\lvert D_+ f(1) - f'(1)\rvert / h$
0.1	2.8588	0.1406	1.4056
0.05	2.7874	0.0691	1.3821
0.025	2.7525	0.0343	1.3705
$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
0	e	0	$\frac{e}{2} = \frac{1}{2}f''(1)$

```
% Matlab
  exact_value = exp(1);
  for j=1:65
    h(j) = 1/2^(j-1);
    computed_value = (exp(1+h(j)) - exp(1))/h(j);
    error(j) = abs(computed_value - exact_value);
  end
  plot(h,error,h,error,'o'); xlabel('h'); ylabel('error')
  loglog(h,error,h,error,'o'); ...
```

finite-difference approximation of a derivative





note

If error $\approx ch^p$, then p is called the order of accuracy of the approximation. In this case we have $\log(\text{error}) \approx \log(ch^p) = \log c + p \log h$; this implies that the slope of the data on the log-log plot gives the value of p. We see that $p = 1$ for large h (expected) and $p = -1$ for small h (unexpected).

question : why does the error increase for small h?

1. The computed value has two sources of error: truncation error is due to replacing the exact derivative $f'(x)$ by the finite-difference approximation $D_+ f(x)$, and roundoff error is due to using finite precision arithmetic.

2. The truncation error is $O(h)$ and the roundoff error is $O(\epsilon/h)$, where $\epsilon \approx 10^{-15}$ in Matlab.

3. The total error is $O(h) + O(\epsilon/h)$. Hence, for large h the truncation error dominates the roundoff error, but for small h the roundoff error dominates the truncation error.

note

backward difference : $D_- f(x) = \dfrac{f(x) - f(x-h)}{h}$

centered difference : $D_0 f(x) = \dfrac{f(x+h) - f(x-h)}{2h}$ ... hw