16. IEEE floating point numbers

- floating point numbers with base 10
- floating point numbers with base 2
- IEEE floating point standard
- machine precision
- rounding error

Floating point numbers with base 10

notation:

$$x = \pm\,(.d_1 d_2 \ldots d_n)_{10} \cdot 10^e$$

- $.d_1 d_2 \ldots d_n$ is the *mantissa* (d_i integer, $0 \leq d_i \leq 9$, $d_1 \neq 0$ if $x \neq 0$)
- n is the *mantissa length* (or *precision*)
- e is the *exponent* ($e_{\min} \leq e \leq e_{\max}$)

interpretation: $x = \pm(d_1 10^{-1} + d_2 10^{-2} + \cdots + d_n 10^{-n}) \cdot 10^e$

example (with $n = 7$):

$$\begin{aligned} 12.625 &= +\,(.1262500)_{10} \cdot 10^2 \\ &= +\,(1 \cdot 10^{-1} + 2 \cdot 10^{-2} + 6 \cdot 10^{-3} + 2 \cdot 10^{-4} + 5 \cdot 10^{-5} \\ &\quad + 0 \cdot 10^{-6} + 0 \cdot 10^{-7}) \cdot 10^2 \end{aligned}$$

used in pocket calculators

properties

- a finite set of numbers
- unequally spaced: distance between floating point numbers varies
  - the smallest number greater than 1 is $1 + 10^{-n+1}$
  - the smallest number greater than 10 is $10 + 10^{-n+2}$, . . .
- largest positive number:

$$+(.999\cdots 9)_{10} \cdot 10^{e_{\max}} = (1 - 10^{-n})10^{e_{\max}}$$

- smallest positive number:

$$x_{\min} = +(.100\cdots 0)_{10} \cdot 10^{e_{\min}} = 10^{e_{\min}-1}$$

Floating point numbers with base 2

notation

$$x = \pm(.d_1 d_2 \ldots d_n)_2 \cdot 2^e$$

- $.d_1 d_2 \ldots d_n$ is the *mantissa* ($d_i \in \{0, 1\}$, $d_1 = 1$ if $x \neq 0$)
- n is the *mantissa length* (or *precision*)
- e is the *exponent* ($e_{\min} \leq e \leq e_{\max}$)

interpretation: $x = \pm(d_1 2^{-1} + d_2 2^{-2} + \cdots + d_n 2^{-n}) \cdot 2^e$

example (with $n = 8$):

$$\begin{aligned} 12.625 &= +(.11001010)_2 \cdot 2^4 \\ &= +(1 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} + 0 \cdot 2^{-4} + 1 \cdot 2^{-5} \\ &\quad + 0 \cdot 2^{-6} + 1 \cdot 2^{-7} + 0 \cdot 2^{-8}) \cdot 2^4 \end{aligned}$$

used in almost all computers

a finite set of unequally spaced numbers

- for $n = 3$, $e_{\min} = -1$, $e_{\max} = 2$

- largest positive number:

$$x_{\max} = +(.111\cdots1)_2 \cdot 2^{e_{\max}} = (1 - 2^{-n})2^{e_{\max}}$$

- smallest positive number:

$$x_{\min} = +(.100\cdots0)_2 \cdot 2^{e_{\min}} = 2^{e_{\min}-1}$$

- in practice, the number system includes '*subnormal numbers*': unnormalized small numbers ($d_1 = 0$, $e = e_{\min}$), and the number 0

IEEE standard for binary arithmetic

specifies two binary floating point number formats

IEEE standard single precision:

$$n = 24, \qquad e_{\min} = -125, \qquad e_{\max} = 128$$

requires 32 bits: 1 sign bit, 23 bits for mantissa, 8 bits for exponent

IEEE standard double precision:

$$n = 53, \qquad e_{\min} = -1021, \qquad e_{\max} = 1024$$

requires 64 bits: 1 sign bit, 52 bits for mantissa, 11 bits for exponent

used in almost all modern computers

Machine precision

definition: the *machine precision* of a binary floating point number system with mantissa length n is defined as

$$\epsilon_M = 2^{-n}$$

example: IEEE standard double precision ($n = 53$):

$$\epsilon_M = 2^{-53} \approx 1.1102 \cdot 10^{-16}$$

interpretation:

$1 + 2\epsilon_M$ is the smallest floating point number greater than 1:

$$(.10\cdots01)_2 \cdot 2^1 = 1 + 2^{1-n} = 1 + 2\epsilon_M$$

Rounding

a floating-point number system is a finite set of numbers; all other numbers must be rounded

notation: $\mathrm{fl}(x)$ is the floating-point representation of x

rounding rules used in practice:

- numbers are rounded to the nearest floating-point number
- in case of a ties: round to the number with least significant bit 0 ('round to nearest even')

example: numbers $x \in (1, 1 + 2\epsilon_M)$ are rounded to 1 or $1 + 2\epsilon_M$:

- $\mathrm{fl}(x) = 1$ if $1 < x \leq 1 + \epsilon_M$
- $\mathrm{fl}(x) = 1 + 2\epsilon_M$ if $1 + \epsilon_M < x < 1 + 2\epsilon_M$

gives another interpretation of ϵ_M: numbers between 1 and $1 + \epsilon_M$ are indistinguishable from 1

Rounding error and machine precision

general result:

$$\frac{|\,\mathrm{fl}(x) - x|}{|x|} \leq \epsilon_M$$

(no proof)

- machine precision gives a bound on the relative error due to rounding
- number of correct (decimal) digits in $\mathrm{fl}(x)$ is roughly

  $$-\log_{10} \epsilon_M$$

  $i.e.$, about 15 or 16 in IEEE double precision
- fundamental limit on accuracy of numerical computations

Exercises

- explain the following Matlab results (Matlab uses IEEE double precision)

```
>> (1 + 1e-16) - 1
ans =
     0
>> (1 + 2e-16) - 1
ans =
    2.2204e-16
>> (1 - 1e-16) - 1
ans =
   -1.1102e-16
>> 1 + (1e-16 - 1)
ans =
     1.1102e-16
```

- run the following code in Matlab and explain the result

```
x = 2;
for i=1:54
   x = sqrt(x);
end;
for i=1:54
   x = x^2;
end
```

- explain the following results ($\log(1+x)/x \approx 1$ for small x)

```
>>  log(1+3e-16)/3e-16
ans =
    0.7401

>>  log(1+3e-16)/((1+3e-16)-1)
ans =
    1.0000
```