

Numerical Methods

- Using R
 - Download R & R Studio
-

40% Final, 60% Assignments, quizzes

Will consider:

Solving "on Computer"

- $f(x) = 0$

$f: \mathbb{R} \rightarrow \mathbb{R}$ some fn

- $A \overset{n \times 1}{x} = \underset{n \times n}{b}$

- $\int_a^b f(x) dx$

$f: \mathbb{R} \rightarrow \mathbb{R}$ bounded
continuous

- $y: [0, T] \rightarrow \mathbb{R}$

$f: \mathbb{R}_+ \times \mathbb{R} \rightarrow \mathbb{R}$

$$\frac{dy}{dt} = f(t, y)$$

- often not able to find exact solns
- Need approximations for practical purposes

How good is approximation?

What is good enough?

Efficiency: How much work to reach soln.

Accuracy: How close to true value

Precision: Level of detail in our solution.

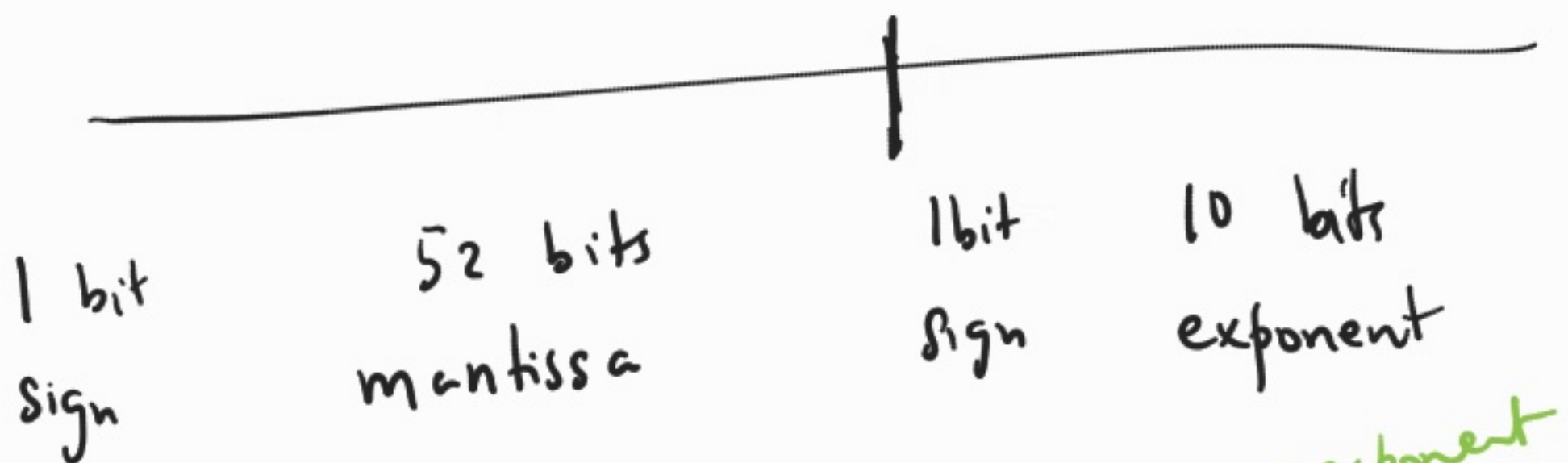
$$\pi = 3.1415926535 \dots$$

$$\frac{22}{7} = 3.14285714 \dots$$

3.1416
is more
accurate
approximation
to π

Computer softwares work with
floating point numbers

Floating point numbers in R
64 bits



$$4 =$$

$$\textcircled{1} \times 2^{\textcircled{2}} \rightarrow \text{exponent}$$

↑
mantissa

$$15 = 1111 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

sign 0 is +1 1 is -1



Binary 1.111 $\times 2^3$

Largest number that R can work with = ?

Max Exponent = $\underbrace{111 \dots 1}_{10} = 2^{10} - 1 = 1023$

Max Mantissa = $\underbrace{1.11 \dots 1}_{51} = \underbrace{1 + 1 \cdot \frac{1}{2} + 1 \cdot \frac{1}{4} + \dots + 1 \cdot \frac{1}{2^{51}}}_{\text{Decimal}}$

$$\left(1 + \frac{1}{2} + \dots + \frac{1}{2^{51}}\right) \times 2^{1023} \leftarrow \text{Max \# R can handle}$$

$$> 2 \times 2^{1023} \quad \text{NA}$$

$$> 1.5 \times 2^{1023} \quad \checkmark$$

Smallest positive number R can handle : $\frac{1}{2^{51}} \times 2^{-1023}$

$$> 20.55 - 19.2 - 1.35$$

$$1.332268 e^{-15}$$

$$> 20.55 - 1.35 - 19.2$$

0

Errors in computing

- Finite storage space
- Cannot store irrational numbers like π exactly (truncation error)

Effects

- Floating Point number line
- - largest & smallest number

• Need to be more careful. ||

Computer Storage

Bit (Binary digit) 0 or 1

Smallest unit of storage

Think of this as switching on/off.

Byte : Composed of 8 bits

Can store up to 2^8 characters
in a byte.

ASCII in Wikipedia : (Check)

Can store characters like 0, 1, 2, 3, ... 9

A, B, C, ... Z, a, b, c, ... z,

@, !, , ; ... in a byte.

Computers can work with 32 bits
or 64 bits (Machine) in one go

Storing numbers

Integers stored in R using 32 bits

↑
one bit
for sign

31 bits number
in binary

Decimal: 32 29 201 . . .

Binary : 101, 1111, 10, 110 . . .

$$32 = 3 \times 10^1 + 2 \times 10^0$$

$$201 = 2 \times 10^2 + 0 \times 10^1 + 1 \times 10^0$$

Binary 101 = $1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 5$ in decimal

↑
most significant

↑
least significant digit

Binary $1110 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
 $= 14$ in decimal.

Max number that can be stored
as an integer is

$\underbrace{1 \ 1 \ 1 \ \dots \ 1}_{31 \text{ bit}}$

$$1 \times 2^{30} + 1 \times 2^1 + 1 \times 2^0$$

$$= 2^{31} - 1$$

In R eq. to store 231 as ^{an} integer
 $> 231L$

Floating point numbers . Default in R

Decimal 21.8125

$$= 2 \times 10^1 + 1 \times 10^0 + 8 \times 10^{-1} + 1 \times 10^{-2} + 2 \times 10^{-3} + 5 \times 10^{-4}$$

Binary of 21.8125

$$\begin{aligned} 21 &= 1 \times 2^4 + 1 \times 2^2 + 1 \times 2^0 \\ &= 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \end{aligned}$$

21 in Binary 1 0 1 0 1

$$\begin{aligned} 0.8125 &= 0.5 + 0.3125 \\ &= 0.5 + 0.25 + 0.0625 \\ &= 1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4} \end{aligned}$$

Binary 0.1101

$$\begin{array}{l} 2^{-1} = 0.5 \\ 2^{-2} = 0.25 \\ 2^{-3} = 0.125 \\ 2^{-4} = 0.0625 \end{array}$$

Binary of 21.8125 is

10101.1101

Eg Decimal 0.2

$$\begin{aligned} 0.2 &= 0.125 + 0.075 \\ &= 0.125 + 0.0625 \\ &\quad + 0.0125 \\ &= 0.125 + 0.0625 \\ &\quad + 0.0078125 + \dots \end{aligned}$$

? Infinite Expansion.

Remarks.

Infinite expansion in binary cannot be stored!

$$2^{-1} = 0.5$$

$$2^{-2} = 0.25$$

$$2^{-3} = 0.125$$

$$2^{-4} = 0.0625$$

$$2^{-5} = 0.03125$$

$$2^{-6} = 0.015625$$

$$2^{-7} = 0.0078125$$

⋮

A floating point number is stored
in 64 bits

Sign - 1 bit

Mantissa - 52 bits

Exponent - 11 bits

(1 bit for sign)

64 bits

Sign Bit

0 or 1
(positive) (negative)

Exponent Range, integers in $[-1022, 1023]$

Stored in binary with a bias of 1023

So -1022 is stored as $-1022 + 1023 = 1$

1023

"

$1023 + 1023$
 $= 2046$

$$-1022 \rightarrow 1$$

In 11 Bits

0 0 0 1
10 0's

$$1023 \rightarrow 2046 = 1024 + 512 + 256 + 128 + 64 + 32 + 16 + 8 + 4 + 2$$

= 1 1 1 1 0
 10 1's

Remark. Note exponents $-1022, \dots, 0$ when stored_n in binary have 0 as 1st digit

$1, 2, \dots, 1023$ when stored with bias in binary have 1 as 1st digit

Mantissa 52 bits

$$\begin{array}{l} 231.56 \rightarrow 2.3156 \times 10^2 \\ 0.0156 \rightarrow 1.56 \times 10^{-2} \end{array}$$

Similarly write binary number with
just one 1 to the left of binary point.
(get extra bit for free!)

$$\begin{array}{c} 0.011011 \dots 1 \\ \hline 54 \end{array}$$
$$= 1.1011 \dots 1 \times 2^2$$

\uparrow
always 1

52

Eg $21.8125 \rightarrow 10101.1101$ in binary

Sign Bit

0

a) $\leftarrow 10101.1101 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4}$

b) $\leftarrow 1.0101101 = 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} + \dots$

a) $= 2^4 \times$ b)

Mantissa:

0101101 0...0
44 bits

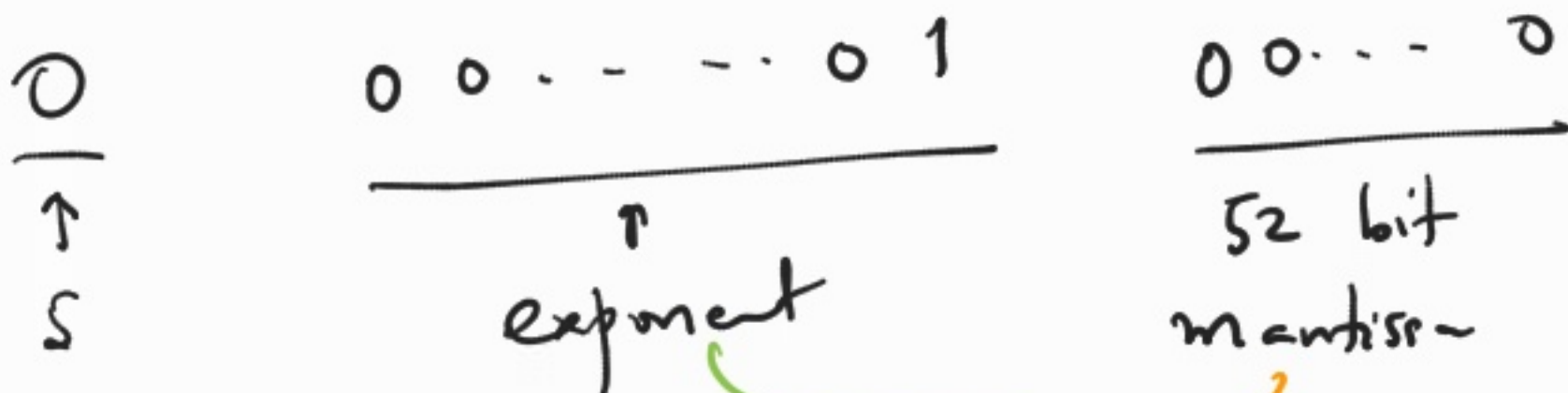
Exponent:

4 \rightarrow 1027
 $= 1024 + 2 + 1$

10...011
11 bits

~ "Smallest +ve number" "

1.00...0



$$\left(1 + \sum_{i=1}^{52} 0 \times 2^{-i} \right) \times 2^{1-1023}$$

> Machine \$ double . xmin

Remark : R can work with smaller numbers ("denormal numbers") at the cost of precision. (some digits from mantissa go to exponent)

"Largest +ve number"

$$\begin{array}{ccc} 0 & 111\dots10 & 1111\dots1 \\ \hline \text{sign} & \text{exponent} & 52 \text{ bit} \end{array}$$
$$\left(1 + \sum_{i=1}^{52} 1 \times 2^{-i} \right) \times 2^{2046-1023}$$

Remark: R cannot work with
larger numbers. Losing precision
not an option

Special numbers

0	0 0 0 0 ... 0	0 0 ... 0	+0
1	0 ... 0	0 ... 0	-0
<u>sign</u>	<u>exponent</u>	<u>mantissa</u>	
0	1 ... 1	0 ... 0	+Inf
1	1 ... 1	0 ... 0	-Inf
0	1 ... 1	1 0 ... 0	NaN
1	1 ... 1	1 0 ... 0	
		51 bits	

"NaN" not a number

Eg

Sign

0

$$1 = 1 \times 2^0 \Rightarrow \text{Binary } 1.00\dots$$

Exponent
011...

Mantissa

00...0

Exponent = 0 $\rightarrow 1023$
