Numerical Methods :: 8/1/2020 - 10:00 am

- Solving problems where theory does not go/apply.
- Seek solutions to practical problems (good enough)

Efficiency Accuracy Precision [level of detail]

- Different from other branches of mathematics. One needs to execute a computation & find an answer.

Examples :- (That we have seen)

- Roots of polynomials / function.

  Solve :- $f(x) = 0$
  for $x \in \mathbb{R}$.

- Solutions to linear system of equations.

$$Ax = b$$

$$x \in \mathbb{R}^n, \quad A_{n\times n}, \quad b_{n\times 1}$$

- $\{(x_i, y_i) : 1 \le i \le n\}$ - Data

Relationship f : $y_i = f(x_i)$

$\mathbb{R}^m \ni y_i ; x_i \in \mathbb{R}^d$

f - linear, Polynomial

- $\int_a^b f(x)\,dx$ $\quad f : \mathbb{R} \to \mathbb{R}$

bounded

continuous

- $y : [0, T] \to \mathbb{R}$ $\quad f : \mathbb{R}_+ \times \mathbb{R} \to \mathbb{R}$

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

For each of the above, we :-

- Explore where theory begins / ends.
- Reach for cases where manual computation is not possible.
- Find the answer.

Cannot, then get as close to it as possible.

Question :- What are the limitations. ?

Machines can only store / work with numbers that can be represented by finitely many digits. [number of bits allocated]

$\Rightarrow$ There is a largest number
& a smallest number

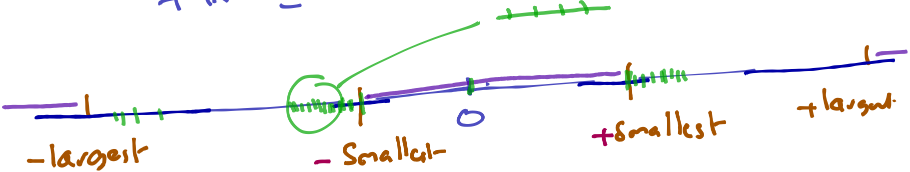
We are used to real line $\mathbb{R}$.

$a, b \in \mathbb{R}$ — $\#\{x \in \mathbb{R} \mid a \leq x \leq b\}$ is infinite & we have spent classes understanding infinity.

For a machine the real line does not exist. It works with floating point line. - $f\mathbb{R}$

- there $\hat{a}, \hat{b}$ are floating point numbers
$= \# \{ x \in f\mathbb{R} \mid \hat{a} \leq x \leq \hat{b} \}$
is finite (could be zero).

$f\mathbb{R} \subseteq \mathbb{R}$



· Range is limited.

$\Rightarrow$ Values created in any computation (add, subtracting, multiplying, dividing) will also have limited range.

Further this will introduce errors

E.g. $(\mathbb{R}, +, *, \div)$ is well understood object.

$\mathbb{FR}$ is not "closed" under arithmetic operations. In the sense addition of floating point values cannot be represented always as another floating pt. value.

Introduce:
[fl R arithmetic]

- Round off error
- Catastrophic cancellation
- Truncation error

- Keep track of each computation

&

also

Number of computations.

Size of Example is n

theory A — results in computations a_n

B b_n

we must know how to compare a_n & b_n.

Stability issues of a Theory :-

Say we know a method to solve

$Ax = b$ $\quad A_{n\times n}$ $\quad b_{n\times 1}$

$x \in \mathbb{R}^n$

Suppose change A

$\tilde{a}_{ij} = a_{ij} + \varepsilon \leftarrow$ small #

$\tilde{x}$ solves $\tilde{A}\tilde{x} = b$

Is $\tilde{x}$ close to x?

Some Computations : Jan 10 - 2pm

We wish to evaluate the polynomial

$$f(x) = a_0 + a_1 x + \dots + a_n x^n$$

To evaluate $f(c)$

Algorithm 1

```
Coeff = c(a0, a1, .. an)
    x = c ;
    Eval = 0
for i = 1, ..., n+1
    Eval = Eval + Coeff[i] * x^(i-1)
end
```

In each loop the algorithm executes one addition and (i^{th} loop) 'i' multiplications.

$\Rightarrow$ Total number of multiplication & addition, $= n + \sum_{i=0}^{n} i = n + \frac{n^2+n}{2}$

Algorithm 2 :·

Coeff = c(a_0, a_1, .. a_n)

$x = c$

Eval = 0

C[x] = 1

for $i = 1, \dots, n$ |

Eval = Eval + Coeff [i] . C[x]

C[x] = C[x] $\times$ x.

end
return (Eval)

The above algorithm reduces multiplications to $2n$.

Horner's Method :- Rewrite

$$f(x) = a_0 + a_1 x + \cdots + a_n x^n$$
$$= a_0 + a_1 x + \cdots + x^{n-1}(a_{n-1} + x a_n)$$
$$\vdots$$
$$= a_0 + x(a_1 + \cdots + x(a_{n-1} + x a_n))$$

Algorithm Horner

```
Coeff = c(a0, a1, .. an)
    x = c ;
    Eval = 0
  for i = 1, ..., n+1
      Eval = Eval * x + Coeff[i]
  end
  return(Eval)
```

In this method in each loop there is one addition and one multiplication.

- One must try to minimize the number of computations for any given situation.