NC Lecture Notebook

Contents

1 Introduction

Examples:

1. Compute the partial sums of the harmonic series
$$\sum_{k=1}^{n} \frac{1}{k}$$
   - $S(n) = 1 + \frac{1}{2} + \cdots + \frac{1}{n-1} + \frac{1}{n}$
   - $s(n) = \frac{1}{n} + \frac{1}{n-1} + \cdots + \frac{1}{2} + 1$

   Mathematically, $S(n) = s(n)$; Computationally, they're not.

   The difference growth with n

2. Let $f(x) = \sqrt{x}$ for $x > 0$, and we know $f'(x) = \dfrac{1}{2\sqrt{x}}$

   Define a function
$$\begin{aligned} y(k) &= \frac{f(16+k) - f(16)}{k} \\ &= \frac{\sqrt{16+k} - 4}{k} \end{aligned}$$
   then, $\lim_{k \to 0} y(k) = f'(16) = \frac{1}{8}$

   As k decrease to $k = 10^{-12}$, $y(k)$ is a good approximation for $f'(16) = \dfrac{1}{8}$, when k further decrease, y(k) starts to oscillates and the errors are visible; after k drops below 10^{-14}, the computed $y(k)$ is around 0.
$$\frac{\sqrt{16+k} - 4}{k} = \frac{1}{\sqrt{16+k} + 4}$$
   $\dfrac{\sqrt{16+k} - 4}{k}$ goes to 0.

3. Consider the function $y(x) = (x-1)^8$ and it's expanded form
$$y(x) = x^8 - 8x^7 + 28x^6 - 56x^5 + 70x^4 - 56x^3 + 28x^2 - 8x + 1$$
   Again, two mathematically identical function are nor the same computationally.

   The expected form even leads to negative values.

1.1 Computational Complexity : Polynomial Evaluation

$$p(x) = 2x^4 + 3x^3 - 3x^2 + 5x - 1$$

1. Straight Forward:

$$\begin{aligned} 2 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} & \qquad 4(x) \\ 3 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} & \qquad 3(x) \\ (-3) \cdot \frac{1}{2} \cdot \frac{1}{2} & \qquad 2(x) \\ 5 \cdot \frac{1}{2} & \qquad 1(x) \end{aligned}$$

$$N_x = 10, N_+ = 4$$

2. Storage

$$\left(\frac{1}{2}\right)^2 = \frac{1}{4} \qquad 1(x)$$
$$\left(\frac{1}{2}\right)^3 = \left(\frac{1}{2}\right)^2 \cdot \frac{1}{2} = \frac{1}{8} \qquad 1(x)$$
$$\left(\frac{1}{2}\right)^4 = \left(\frac{1}{2}\right)^3 \cdot \frac{1}{2} = \frac{1}{16} \qquad 1(x)$$

multiply by coefficient $4(x)$

$$N_x = 7, N_+ = 4$$

3. Horner's method:

$$\begin{aligned} P(x) &= x(2x^3 + 3x^2 - 3x + 5) - 1 \\ &= x(x(2x^2 + 3x - 3) + 5) - 1 \\ &= x(x(x(2x + 3) - 3) + 5) - 1 \end{aligned}$$

$$N_x = 4, N_+ = 4$$

$$\begin{aligned} N_+ &= 4 \\ N_x &= \begin{cases} \sum_{k=1}^{d} k & = \dfrac{d(1+d)}{2} \\ 2d - 1 \\ d \end{cases} \end{aligned}$$

1.2 Floating Point Arithmetic

Consider -321.416: (Decimal Representation)

$$-321.416 = -(3 \cdot 10^2 + 2 \cdot 10 + 1 \cdot 0 + 4 \cdot 10^{-1} + 1 \cdot 10^{-2} + 6 \cdot 10^{-3})$$

A similar representation is used in computer: floating - point arithmetic:

$$-.321416 \times 10^3$$

sign, fraction, base, exponent
In general,

$$\pm f \times \beta^e$$

where $\beta = 2$: binary number
$\beta = 10$: decimal number
$\beta = 16$: hexadecimal number f: fraction, digits from $0, 1, \cdots \beta - 1$
e: exponent, digits from $0, 1, \cdots \beta - 1$
Binary Number:

$$b_m \cdots b_2 b_1 b_0 . a_1 a_2 \cdots a_n$$

(all integer)

Each digit b_i, a_j takes 0 or 1.
This number in base 10, is

$$b_m \cdot 2^m + b_{m-1} \cdot 2^{m-1} + \cdots + b_1 \cdot 2^1 + b_0 \cdot 2^0 + a_1 \cdot 2^{-1} + a_2 \cdot 2^{-2} + \cdots + a_n 2^{-n}$$

Note:

$$\begin{aligned}(0.1101)_2 &= (1.101)_2 \cdot 2^{-1} \\ &= (0.001101)_2 \cdot 2^3\end{aligned}$$

To convert between binary ($\beta = 2$) and decimal ($\beta = 10$)
Example:

1. $x = (1.1011)_2$ convert x to a decimal number:

$$\begin{aligned}x &= 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 1 \cdot 2^{-4} \\ &= 1 + \frac{1}{2} + 0 + \frac{1}{8} + \frac{1}{16} \\ &= \frac{27}{16}\end{aligned}$$

2.

$$\begin{aligned}x &= (1.1010 \cdots 10)_2 \\ &= (1.1\bar{0})_2 \\ &= 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-3} + 1 \cdot 2^{-5} + \cdots\end{aligned}$$

Recall geometric series:

$$1 + r + r^2 + \cdots = \frac{1}{1-r}$$

if $|r| < 1$

$$\begin{aligned}x &= 1 + \frac{1}{2} + \left(\frac{1}{2}\right)^3 + \left(\frac{1}{2}\right)^5 + \cdots \\ &= 1 + \frac{1}{2}\left(1 + \frac{1}{4} + \left(\frac{1}{4}\right)^2 + \cdots\right) \\ &= 1 + \frac{1}{2} \cdot \frac{1}{1 - \frac{1}{4}} \\ &= 1 + \frac{2}{3} \\ &= \frac{5}{3}\end{aligned}$$

Alternatively,

$$\begin{aligned}x &= (1.1\bar{0})_2 \\ &= 1 \cdot 2^0 + (0.1\bar{0})_2 \\ &= 1 + (10.1\bar{0})_2 \cdot 2^{-2}\end{aligned}$$

$$
\begin{aligned}
y &= (0.\bar{1}\bar{0})_2 \\
&= (10.\bar{1}\bar{0})_2 \cdot 2^{-2} \\
&= \{(10)_2 \cdot 2^{-2} + (0.\bar{1}\bar{0})_2 \cdot 2^{-2}\} \\
y &= (2+y) \cdot 2^{-2} \\
4y &= 2 + y \\
y &= \frac{2}{3} \\
x &= 1 + y \\
&= \frac{5}{3}
\end{aligned}
$$

3. Convert 14.8125 to a binary number:

   We are looking for

$$14.8125 = (b_m b_{m-1} \cdots b_1 b_0 . a_1 a_2 \cdots a_n)_2$$

   Fractional part:

$$
\begin{aligned}
0.8125 &= (.a_1 a_2 \cdots a_n)_2 \\
&= a_1 \cdot 2^{-1} + a_2 \cdot 2^{-2} + \cdots + a_n \cdot 2^{-n}
\end{aligned}
$$

   - $*2$

$$
\begin{aligned}
1.6250 &= a_1 + a_2 \cdot 2^{-1} + \cdots a_n \cdot 2^{-(n-1)} \\
a_1 &= 1 \\
0.6250 &= a_2 \cdot 2^{-1} + \cdots a_n \cdot 2^{-(n-1)}
\end{aligned}
$$

   - $*2$

$$
\begin{aligned}
1.2500 &= a_2 + a_3 \cdot 2^{-1} \cdots a_n \cdot 2^{-(n-2)} \\
a_2 &= 1
\end{aligned}
$$

   - $*2$

$$
\begin{aligned}
0.25 \cdot 2 &= 0.50 && a_3 = 0 \\
0.50 \cdot 2 &= 1 && a_4 = 1
\end{aligned}
$$

$$\therefore 0.8125 = (.1101)_2$$

   Collect Integer part ordered from radix point:

   Integer part:

$$
\begin{aligned}
14 &= (b_m \cdots b_2 b_1 b_0)_2 \\
&= b_m \cdot 2^m + b_{m-1} \cdot 2^{m-1} + \cdots b_1 \cdot 2^1 + b_0
\end{aligned}
$$

Divided by 2:

$$\begin{aligned}
\frac{14}{2} &= 7R0 \\
&= (b_m \cdot 2^{m-1} + \cdots b_1)Rb_0 \\
\frac{7}{2} &= 3R1 && b_1 \\
\frac{3}{2} &= 1R1 && b_2 \\
\frac{1}{2} &= 0R1 && b_3 \\
\therefore 14 &= (1110)_2
\end{aligned}$$

$$\therefore 14.8125 = (1110.1101)_2$$

1.2.1 Floating point number

$$\pm f \times \beta^e$$

f(fraction) : the number of digits in f determines the precision.
e(exponent):the number of digits in e determines the range of representable numbers.
We follow IEEE 754 floating point standard:

1. Normalized form: $f = 1.b_m b_{m-1} \cdots b_1 b_0$, $(0.0101010 \cdots)$
2. Advantage: leading 1 needs not be stored
   - 32-bit single precision:
     sign : 1 -bit
     exponent : 8-bits
     fraction: 23 -bits
   - 64 - bit double precision:
     sign : 1 -bit
     exponent : 11-bits
     fraction: 52-bits
3. The represented number is
   $$(-1)^s \cdot (1+f) \cdot 2^{e-e_0}$$
   - e: unsigned, e^0: exponent bias
   - $e - e^0$: can be either positive or negative (negative represent small number)

   Let's focus on "e" or equivalently 2^{e-e_0}
   - Single Precision:

$$\begin{aligned}
e &\in [e_{min}, e_{max}] \\
e_{min} &= (0 \cdots 01)(8 \text{ bit}) = 1 \\
e_{max} &= (11 \cdots 10) \\
&= 1 \cdot (2 + 2^2 + \cdots + 2^7) \\
&= 2 \cdot \left(\frac{1-2^7}{1-2}\right) \\
&= 254 \\
\implies 2^{e-e_0} &\in [2^{-126}, 2^{127}] && e_0 = 127 \\
&\approx [10^{-38}, 10^{38}]
\end{aligned}$$

- $$\begin{aligned} e &\in [e_{min}, e_{max}] \\ e_{min} &= 1 \\ e_{max} &= 2^1 + 2^2 + \cdots + 2^{10} \\ &= 2046 \\ e_0 &= 1023 \\ 2^{e-e_0} &\in [2^{-1022}, 2^{1023}] \\ &\approx [10^{-308}, 10^{308}] \end{aligned}$$

1.2.2 fraction f and precision

Using double-precision on an example:

- How to store a number
- How to do calculation

Consider

$$\begin{aligned} x_1 &= \frac{27}{16} = (1.1011)_2 \\ x_2 &= \frac{5}{3} = (1.\bar{1}\bar{0})_2 \\ x_3 &= \frac{2}{3} = (.\bar{1}\bar{0})_2 = (1.\bar{0}\bar{1})_2 \cdot 2^{-1} \\ x_4 &= 1 \qquad\qquad = (1.0)_2 \\ x_5 &= 1 \times 2^{-52} \\ x_6 &= 1 \times 2^{-53} \end{aligned}$$

$$\begin{aligned} x_1 &: 1.\boxed{101100\cdots 0}(52bits) \\ x_4 &: 1.\boxed{00\cdots 0}(52bits) \\ x_5 &: 1.\boxed{00\cdots 0}(52bits) \times 2^{-52} \\ x_6 &: 1.\boxed{00\cdots 0}(52bits) \times 2^{-53} \end{aligned}$$

Now x_2, x_3:

$$\begin{aligned} x_2 &: 1.\boxed{101010\cdots 10}10\cdots \\ x_3 &: 1\boxed{0101\cdots 01}0101\cdots \times 2^{-1} \end{aligned}$$

We follow: IEEE rounding to the nearest rule: $x \to fl(x)$
General relative rounding error:

$$\frac{|fl(x) - x|}{|x|} \leqslant \frac{1}{2} \cdot 2^{-52}$$

Example:

$$
\begin{aligned}
x_1 &= \frac{27}{16} = (1.1011)_2 = fl(x_1) \\
x_2 &= \frac{5}{3} = 1\frac{2}{3} = (1.\bar{1}\bar{0})_2 = 1.\boxed{10\cdots 10}1010\cdots \\
fl(x_2) &= 1.\boxed{10\cdots 101011} \\
x_3 &= \frac{2}{3} = (0.\bar{1}\bar{0})_2 = (0.1010\cdots 10)_2 = (1.0101\cdots 01\cdots)_2 \cdot 2^{-1} \\
fl(x_3) &= 1.\boxed{0101\cdots 01} \cdot 2^{-1} \\
x_4 &= 1 = fl(x_4) \\
x_5 &= 2^{-52} = 1.00\cdots 0 \cdot 2^{-52} = fl(x_5) \\
x_6 &= 2^{-53} = fl(x_6)
\end{aligned}
$$

And

$$
\begin{aligned}
fl(x_2) &= 1.\boxed{1010\cdots 1011} \\
&= x_2 - 0.\boxed{0\cdots 0}1010\cdots + 0.\boxed{00\cdots 01} \\
&= x_2 - (0.\bar{1}\bar{0})_2 \times 2^{-52} + 2^{-52} \\
&= x_2 + \frac{1}{3} \cdot 2^{-52} \\
\frac{|fl(x_2) - x_2|}{|x_2|} &= \frac{\frac{1}{3} \cdot 2^{-52}}{\frac{5}{3}} \\
&= \frac{1}{5} \cdot 2^{-52}
\end{aligned}
$$

Machine epsilon:

$$
\epsilon_{mach} = \begin{cases} 2^{-52} & \text{Double} \\ 2^{-23} & \text{Single} \end{cases}
$$

MATLAB:

- eps
- eps('Single')

Remark:

- ϵ_{mach}: relative rounding "precision", "resolution"
-

$$
\begin{aligned}
1.\boxed{00\cdots 0} &= 1 \\
1.\boxed{00\cdots 1} &= 1 + \epsilon_{mach}
\end{aligned}
$$

1.2.3 Computation

Next, we want to compute, with double precision:

Rule for calculation:

$$a \pm b \ominus fl(fl(a) \pm fl(b)$$

$$\begin{aligned} fl(x_4) + fl(x_5) &= 1 + 2^{-52} \\ &= (1.\boxed{000 \cdots 01})_2 \\ &\ominus 1 + 2^{-52} \end{aligned}$$

$$\begin{aligned} fl(x_4) + fl(x_6) &= 1 + 2^{-53} \\ &= (1.\boxed{00 \cdots 0}1)_2 \\ &\ominus 1 \end{aligned}$$

$$\begin{aligned} fl(x_5) - fl(x_6) &= 2^{-52} - 2^{-53} \\ &\ominus 2^{-53} \end{aligned}$$

$$\begin{aligned} &(x_2 - x_3) - x_4 \\ fl(x_2) - fl(x_3) &= (1.\boxed{10 \cdots 1011})_2 - (1.\boxed{0101 \cdots 01})_2 \cdot 2^{-1} \\ &= (1.\boxed{10 \cdots 1011}0)_2 - (1.\boxed{0101 \cdots 01})_2 \cdot 2^{-1} \\ &= (1.\boxed{00 \cdots 00}1)_2 \\ &\ominus 1 \\ fl(1 - x_4) &= 0 \end{aligned}$$

$$\begin{aligned} &(x_2 - x_4) - x_3 \\ fl(x_2) - fl(x_4) &= (1.\boxed{10 \cdots 1011})_2 - 1 \\ &\ominus (1.\boxed{01 \cdots 10110})_2 \times 2^{-1} fl(x_2 - x_4) - fl(x_3) \quad = (1.\boxed{0101 \cdots 0110})_2 \times 2^{-1} - (1.\boxed{0101 \cdots 0101})_2 \times \\ &= (0.\boxed{00 \cdots 01})_2 \times 2^{-1} \\ &= 2^{-53} \end{aligned}$$

1.2.4 Loss of significant digits

This occurs when we subtract two nearly equal number:
Example:

1. Use 7-digit base 10 floating point arithmetic to compute $x - y$:

$$\begin{aligned} x &= 1.234567 \times 10^5 \\ y &= 1.234566 \times 10^5 \\ x - y &= 0.000001 \times 10^5 \\ &= 10^{-1} \end{aligned}$$

2. Use 3 digit, base 10 floating point arithmetic to compute $\sqrt{9.01}-3$:

$$\begin{aligned}\sqrt{9.01} &= 3.00166\cdots\\ &\circeq 3.00\\ \sqrt{9.01}-3 &\circeq 3.00-3=0\end{aligned}$$

Using double precision:

$$\sqrt{9.01}-3 \approx 1.6662\times 10^{-3}$$

Remedy: reformulate:

$$\begin{aligned}\sqrt{9.01}-3 &= \frac{(\sqrt{9.01}-3)(\sqrt{9.01}+3)}{\sqrt{9.01}+3}\\ &= \frac{0.01}{\sqrt{9.01}+3}\\ &\circeq \frac{0.01}{6}\\ &= 1.67\times 10^{-3}\end{aligned}$$

3.

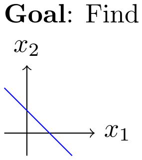
$$\begin{aligned}f(x) &= \frac{1-\cos x}{\sin^2 x}\\ \lim_{x\to 0} f(x) &= \lim_{x\to 0}\frac{1-\cos x}{\sin^2 x}\\ &= \lim_{x\to 0}\frac{\sin x}{2\sin x\cos x}\\ &= \frac{1}{2}\end{aligned}$$

Reformulate:

$$\begin{aligned}f(x) &= f(x)\cdot 1\\ &= \frac{1-\cos^2 x}{\sin^2 x(1+\cos x)}\\ &= \frac{1}{1+\cos x}\\ &= \frac{1}{2}\end{aligned}$$

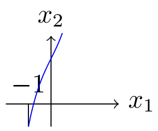
2 Solving an algebraic equation

Goal: Find a solution (or solutions of) $f(x)=0$



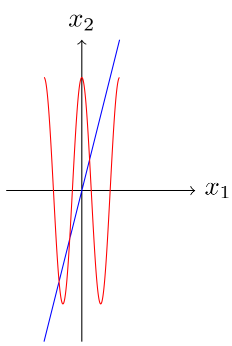
Simple (Trivial) $f(x)=ax+b$
$f(x)=ax+b$, a,b constants, $a\neq 0$
$f(x)=0 \implies x=-\frac{b}{a}$
Example:

1. $f(x) = x^3 + 2x + 2 = 0$



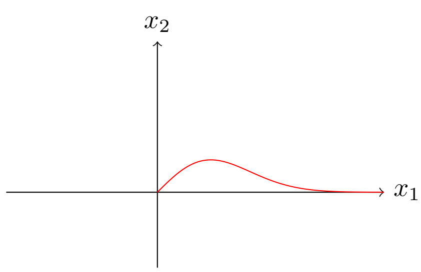
   One root in $(-1, 0)$, $f'(x) = 3x^2 + 2 \geqslant 2$

2. $f(x) = 4x - 3\cos 2\pi x = 0$
   $f_1(x) = 4x$, $f_2(x) = 3\cos 2\pi x$



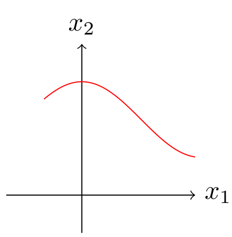
   Therefore, at least three roots.

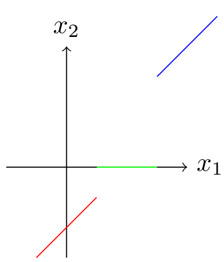
3. Find where $F(x) = x \cdot e^{-x^2}$ attains its maximum $[0, \infty)$



   Convert to $f'(x^*) = 0$

4. $f(x) = 0$ no root!





5. Infinetely many solution (roots)

2.1 Bisection Method

To solve $f(x) = 0$ on $[a, b]$, where $f(x)$ is continuous $[a, b]$ and $f(a)f(b) < 0$

- Set $[a_0, b_0] = [a, b]$
- Let $c_0 = \dfrac{a_0 + b_0}{2}$ be the mid point.
- If $f(c_0) = 0$, then $x = c_0$ is a root, STOP.
- If not, set

$$[a_1, b_1] = \begin{cases} [a_0, c_0] & \text{if } f(a_0)f(c_0) < 0 \\ [c_0, b_0] & \text{if } f(c_0)f(b_0) < 0 \end{cases}$$

  Note that $b_1 - a_1 = \frac{1}{2}(b_0 - a_0)$
  Repeat the process:
  Suppose we have $[a_j, b_j], j \geqslant 1, f(a_j)f(b_j) < 0$, and $b_j - a_j = \frac{1}{2^j}(b_0 - a_0)$
  then, let $c_j = \dfrac{1}{2}(b_j + a_j)$

  1. If $f(c_j) = 0$, then $x = c_j$ is a root, STOP.
  2. If not, set

$$[a_{j+1}, b_{j+1} = \begin{cases} [a_j, c_j] & \text{if } f(a_j)f(c_j) < 0 \\ [c_j, b_j] & \text{otherwise} \end{cases}$$

  We know

$$\begin{aligned} b_{j+1} - a_{j+1} &= \frac{1}{2}(b_j - a_j) \\ &= \frac{1}{2^{j+1}}(b_0 - a_0) \end{aligned}$$

  If the process stops at finitely many steps, we find a solution. Otherwise, the sequence $c_0, c_1, c_2 \cdots c_j \cdots$ with $c_j \in [a_j, b_j]$ approach a solution, denotes as $\bar{x}$, as $j \to \infty$

Error:

$$\begin{aligned} e_j &= |c_j - \bar{x}| \\ &\leqslant \frac{1}{2}|b_j - a_j| \\ &\leqslant \frac{1}{2^{j+1}}|b_0 - a_0| \end{aligned}$$

Computing Cost: To get c_j, $j + 2$ functions on evaluation.

Theorem 2.1. If f is continuous on [a,b], and $f(a)f(b) < 0$, then the midpoint $c_0, c_1 c_2 \cdots c_j \cdots$ computed using the bisection method converges to a solution $\bar{x}$ of $f(x) = 0$, and the error satisfies

$$|c_j - \bar{x}| \leqslant \frac{1}{2^{j+1}}(b - a)$$

Actual error con be smaller.

Remark:

- Bisection method always works as long as
  1. f is continuous
  2. $f(a)f(b) < 0$
- It has an "explicit" formula for the error bound, one can estimate the number of iteration to ensure the error is below a given level δ.

$$
\begin{aligned}
&\frac{1}{2^{j+1}}(b-a) \leqslant \delta \\
\iff\ & 2^{j+1} \geqslant \frac{b-a}{\delta} \\
\iff\ & j+1 \geqslant \frac{\ln((b-a)/\delta)}{\ln(2)}
\end{aligned}
$$

For example: $b-a=2, \delta=10^{-p}$

$$
\begin{aligned}
j+1 &\geqslant \frac{\ln(b-a)-\ln(\delta)}{\ln(2)} \\
&= \frac{\ln(2)-\ln(10^{-p})}{\ln 2} \\
&= 1+p\frac{\ln 10}{\ln 2} \\
\implies j &\geqslant p\frac{\ln 10}{\ln 2} \\
&\geqslant 3.32p
\end{aligned}
$$

If $p=1$, take $j=4$; If $p=2$, take $j=7$.
Discussion:

- Pick a more random $\xi \in [a_j, b_j]$
- Tri-section theory?
- Complex roots

Notation:
Let $I=(a,b),[a,b],(a,b]$ then $f \in C^n(I)$: means $f, f^1 \cdots f^{(n)}$ exists, and are continuous on I.
n : non-negative integer.
when $n=0$, $C^0(I)=C(I)$ (Set of continuous functions).
$f(x), f(t)$, x,t dummy variable.

Theorem 2.2 (Taylor Theorem)**.** Given a function $f \in C^{n+1}([a,b])$, then for any $x, x_0 \in (a,b)$,

$$f(x)=f(x_0)+f'(x_0)(x-x_0)+\frac{f''(x_0)}{2!}(x-x_0)^2+\frac{f^{(3)}(x_0)}{3!}(x-x_0)^3\cdots+\frac{f^{(n)}(x_0)}{n!}(x-x_0)^n+R_{n+1}$$

(Taylor's polynomial of degree n)+Reminder

$$R_{n+1}=\frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)^{n+1}$$

ξ : Some number between x and x_0

2.2 Newton Method

A curve locally can be approximated by a straight line.
To solve $f(x) = 0$, we start with an initial guess x_0, and consider the tangent line of $f(x)$ passing through $(x_0, f(x_0))$

$$f(x_0) + f'(x_0)(x - x_0)$$

Instead of solving $f(x) = 0$, we solve

$$f(x_0) + f'(x_0)(x - x_0)$$

and the solution is denoted as x_1,

$$x_1 = x_0 - \frac{f(x_j)}{f'(x_0)}$$

Repeat the process, and we get

$$x_{j+1} = x_j - \frac{f(x_j)}{f'(x_j)}$$

$j = 0, 1, 2 \cdots$
Example:

$$f(x) = x^3 + 2x + 2$$

we need to solve $f(\bar{x}) = 0$ for $\bar{x}$:
Apply Newton's Method:
By sketching, we know $\bar{x} \in (-1, 0)$, start with $x_0 = -\frac{1}{2}$

$$\begin{aligned} x_{j+1} &= x_j - \frac{f(x_j)}{f'(x_j)} \\ &= x_j - \frac{x_j^3 + 2x_j + 2}{3x_j^2 + 2} \\ &= \frac{2x_j^3 - 2}{3x_j^2 + 2} \\ x_1 &= \frac{2 \cdot (-\frac{1}{2})^3 - 2}{3 \cdot (-\frac{1}{2})^2 + 2} = -\frac{9}{11} \\ &\approx -0.818181 \end{aligned}$$

Apply Newton's method to

$$f(x) = x^3 + 2x + 2 = 0$$

Observation:

- $e_{j+1} \approx c e_j^r$, $r = 2$
- $j = 5, 6$: Due to finite precision

$$x_{j+1} = x_j - \frac{f(x_j)}{f'(x_j)}$$

Theorem 2.3. Given $f \in C^2([a, b])$ with $f(\bar{x}) = 0$ for some $\bar{x} \in (a, b)$, and $f'(\bar{x}) \neq 0$. Start with $x_0 \in [a, b)$ that is sufficiently close to $\bar{x}$, then the Newton's method converges to $\bar{x}$, namely

$$\lim_{j \to \infty} x_j = \bar{x}$$

And the error $e_j = |x_j - \bar{x}|$ satisfies

$$e_{j+1} = c_j e_j^r$$

where $r = 2$ and $\lim_{j\to\infty} c_j = \left|\frac{f''(\bar{x})}{2f'(\bar{x})}\right|$

Convergence of Newton's Method: Quadratic
Example:

1. A bad initial x_0:

   Apply newton method to

   $$f(x) = \frac{x}{1+x^2} = 0$$

   we know $\bar{x} = 0$, initial $x_0 = 2$

   $$\begin{aligned} x_{i+1} &= x_i - \frac{f(x_i)}{f'(x_i)} \\ &= \frac{2x_i^3}{1-x_i^2} \\ x_1 &= \frac{16}{3} \\ x_2 &= \frac{8192}{741} \end{aligned}$$

   Newton method diverges with $x_0 = 2$, instead, we can take $x_0 = \frac{1}{2}$, Newton's method will converge.

2. $$f(x) = x^2 = 0$$

   Root: $\bar{x} = 0$, $f'(\bar{x}) = 2\bar{x} = 0$, Theorem no longer holds.

   Newton's:

   $$\begin{aligned} x_{j+1} &= x_j - \frac{f(x_j)}{f'(x_j)} \\ &= x_j - \frac{x_j^2}{2x_j} \\ &= x_j - \frac{x_j}{2} \\ &= \frac{x_j}{2} \end{aligned}$$

   $$\therefore x_{j+1} = \frac{1}{2}x_j$$

   (not sensitive to initial)

   $$\begin{aligned} e_j &= |x_j - \bar{x}| \\ &= |x_j| \\ e_{j+1} &= \frac{1}{2}e_j \end{aligned}$$

   (Linear convergence)

3. Apply Newton's Method to $f(x) = 4x^4 - 6x^2 - \frac{11}{4}$ with $x_0 = \frac{1}{2}$,

$$x_0 = \frac{1}{2}$$
$$x_1 = -\frac{1}{2}$$
$$x_2 = \frac{1}{2}$$
$$x_3 = -\frac{1}{2}$$

Divergent case : x_j, $j = 0, 1, 2\cdots$ will oscillate between two values

Note: The method can work with a different initial.

Stopping Criteria

- $|x_{j+1} - x_j| < TOL$ (absolute) or $\frac{|x_{j+1} - x_j|}{\max(|x_j|, \epsilon)} < TOL$(relative)
  To deal with the case of $\bar{x} = 0$, here ϵ is some small number of user's choice.
- $j < I_{max}$ (Maximum number of iteration)
- $|x_j| < x_{max}$ (A bounds)

Algorithm (Newton's Method):

Initialization: Pick an initial guess x_0 error tolerance $tol > 0$, maximum iteration number $I_{max} > 0$, (possibly an upper bound of approximated root, $x_{max} > 0$)
Let $err = 10 \times tol$, $j = 0$
while $err > tol$ **do**
  $z = \frac{f(x_j)}{f'(x_j)}$
  $err = |z| = abs(z)$
  $x_{j+1} = x_j - z$
  $j = j + 1$
  if $j > I_{max}$ **then**
    STOP
  end
  if $|x_{j+1}| > x_{max}$ **then**
    STOP
  end
end

Example: (Application)

Given $a \neq 0$, we want to calculate $x = \frac{1}{a}$, by using only $+, -*$, one try:

$$f(x) = ax - 1 = 0$$

Newton:

$$\begin{aligned} x_{j+1} &= x_j - \frac{f(x_j)}{f'(x_j)} \\ &= x_j - \frac{ax_j - 1}{a} \\ &= \frac{1}{a} \end{aligned}$$

(not good)
Another try: $f(x) = a - \frac{1}{x}$, $f'(x) = \frac{1}{x^2}$
Newton's:

$$\begin{aligned} x_{j+1} &= x_j - \frac{f(x_j)}{f'(x_j)} \\ &= x_j - (a - \frac{1}{x_j})x_j^2 \\ &= x_j(2 - ax_j) \end{aligned}$$

(2* 1-) a=3:

$$\begin{aligned} x_0 &= 0.3 \\ x_1 &= 0.3(2 - 3 \times 0.3) \\ &= 0.3(2 - 0.9) \\ &= 0.3 \times 1.1 \\ &= 0.33 \\ x_2 &= 0.3(2 - 3 \times 0.33) \\ &= 0.33(2 - 0.99) \\ &= 0.33 \times 1.01 \\ &= 0.3333 \\ x_3 &= 0.3333(2 - 3 \times 0.3333) \\ &= 0.33333333 \end{aligned}$$

How to compute $e_j = |x_j - \bar{x}|$?

- For general examples, $\bar{x}$ is not available, e_j cannot be calculated.
- To verify your code, one can use examples than have exact solution, namely, $\bar{x}$ is known.
- e_j can be approximated:

$$\begin{aligned} e_j &\approx |x_j - \bar{x}_{ref}| \\ \bar{x}_{ref} &= x_J & J >> j \end{aligned}$$

2.3 Secant method

To solve $f(x) = 0$,
Newton: x_j

$$x_{j+1} = x_j - \frac{f(x_j)}{f'(x_j)}$$

There're application where f' is expensive or non-trivial to compute. Instead, we use $f'(x_j) \approx \frac{f(x_j) - f(x_{j-1})}{x_j - x_{j-1}}$
Secant Method:

$$x_{j+1} = x_j - \frac{f(x_j)(x_j - x_{j-1})}{f(x_j) - f(x_{j-1})}$$

$j \geqslant 1$, with x_0, x_1 given.

To verify:

$$\frac{f(x_j) - f(x_{j-1})}{x_j - x_{j-1}} = \frac{f(x_j) - 0}{x_j - x_{j+1}}$$

With x, w is to take into account storage,

Algorithm (Secant Method):

Initialization: Pick $x_0, x, tol > 0, I_{max} > 0$(possibly $x_{max} > 0$)
Let $err = 10 \times tol$, $j = 0$
while $err > tol$ **do**

$$z = \frac{f(x_j)(x_j - x_{j-1})}{f(x_j) - f(x_j - 1)} = \frac{f(x)(x - w)}{f(x) - f(w)}$$

$err = |z| = abs(z)$
$x_{j+1} = x_j - z$ $(w - x, x = x - z)$
$j = j + 1$
if $j > I_{max}$ **then**
STOP
end
if $|x_{j+1}| > x_{max}$ **then**
STOP
end
end

Theorem 2.4. Given $f \in C^2([a,b])$ with $f(\bar{x}) = 0$ for some $\bar{x} \in (a,b)$, and $f'(\bar{x}) \neq 0$. Start with x_0, x that is sufficiently close to $\bar{x}$, then secant method converges to $\bar{x}$, namely

$$\lim_{j\to\infty} x_j = \bar{x}$$

And the error $e_j = |x_j - \bar{x}|$ satisfies

$$e_{j+1} = D_j e_j^r$$

where $r = \dfrac{\sqrt{5}+1}{2} \approx 1.618$ and $\lim_{j\to\infty} D_j = \left|\dfrac{f''(\bar{x})}{2f'(\bar{x})}\right|^{r-1}$

Convergence of Newton's Method: Super Linear

2.4 Sensitivity of root-finding problems

What is effectively solved on computer (or by MATLAB) is a perturbed problem:

$$\hat{f}(x) = f(x) + h(x) = 0$$

(original: $f(x) = 0$, $h(x)$: due to finite-precision arithmetic)
Consider solving $f(x) = 0$, the solution is $\bar{x}$.
Perturbed problem:

$$\hat{f}(x) = f(x) + \epsilon h(x)$$
$$0 < \epsilon << 1$$

It's root is

$$\hat{x} = \bar{x} + \Delta x$$
$$|\Delta x| << 1$$

"Try to establish the relation of ϵ and Δx"

$$\begin{aligned}
\hat{f}(\hat{x}) &= 0 \\
f(\bar{x}+\Delta x) + \epsilon h(\bar{x}+\Delta x) &= 0 \\
f(\bar{x}) + \Delta x f'(\bar{x}) + \mathcal{O}(\Delta x^2) + \epsilon(h(\bar{x}) + \Delta x h'(\bar{x}) + \mathcal{O}(\Delta x^2)) &= 0 \\
\Delta x(f(\bar{x}) + \epsilon h'(\bar{x})) + \epsilon h(\bar{x}) &\approx 0 \\
\Delta x &\approx -\frac{\epsilon h(\bar{x})}{f'(\bar{x}) + \epsilon h'(\bar{x})} \\
&\approx -\epsilon \frac{h(\bar{x})}{f'(\bar{x})}
\end{aligned}$$

This tells the sensitivity of $\bar{x}$, with the perturbation of $\epsilon h(x)$.
Example:

$$\begin{aligned}
f(x) &= (x-1)\cdots(x-7) \\
&= \Pi_{n=1}^{7}(x-n) \\
h(x) &= x^7 \\
\hat{f}(x) &= f(x) + \epsilon h(x)
\end{aligned}$$

To capture the root $\bar{x} = 6$ of $f(x) = 0$, the root of $\hat{f}$ is $\hat{x} = \bar{x} + \Delta x$:
From analysis:

$$\begin{aligned}
\Delta x &\approx -\epsilon \frac{h(\bar{x})}{f'(x)} \\
h(\bar{x}) &= 6^7 \\
f'(x) &= \sum_{m=1}^{7} \Pi_{n=1, n\neq m}^{7}(x-n) \\
f'(x) &= (x-1)(x-2)(x-3)(x-4)(x-5)(x-7)|_{x=\bar{x}=6} \\
&= 5!(-1) \\
&= -5! \\
\frac{\Delta x}{\epsilon} &\approx \frac{-h(\bar{x})}{f'(\bar{x})} \\
&= \frac{6^7}{5!} \\
&= 2332.8
\end{aligned}$$

In demo:

$$\begin{aligned}
\epsilon = 10^{-6} \frac{\Delta x}{\epsilon} &\approx 2.3322 \times 10^3 \\
\epsilon = 10^{-10} \frac{\Delta x}{\epsilon} &\approx 2.3328 \times 10^3
\end{aligned}$$

Less Sensitive: Well-Conditioned
More Sensitive: Ill- Conditioned

2.5 Fixed Point Iteration

To solve $f(x) = 0$:
Recall Newton's method:
x_0: Initial
$x_{j+1} = x_j - \frac{f(x_j)}{f'(x_j)}$, $j = 0, 1, 2 \cdots$
If $x_0 \approx \bar{x}$, $x_j \to \bar{x}$, $f'(\bar{x} \neq 0)$ as $j \to \infty$
Define $g(x) = x - \frac{f(x)}{f'(x)}$
The newton's method is also

$$x_{j+1} = g(x_j)$$

On the other hand, $\bar{x}$ also satisfies

$$x = g(x) = x - \frac{f(x)}{f'(x)}$$

$(x = g(x))$

Definition 2.1. z is a fixed point of the function G if

$$z = G(z)$$

Fixed Point Iteration (FPI):
Let x_0 be an initial guess:

$$\begin{aligned} x_{j+1} &= G(x_j) && j = 1, 2 \cdots \\ x_1 &= G(x_0) \\ x_2 &= G(x_1) \cdots \end{aligned}$$

How to fixed point problem /iteration related?
If the FPI converges:

$$x_j \to z \text{ as } j \to \infty$$

If G is continuous,

$$G(x_j) \to G(z) \text{ as } j \to \infty$$

That is, the limit z of x_j is a fixed point of G.
Quick Summary:

- The solution $\bar{x}$ of $f(x) = 0$ is also a fixed point of

$$g(x) = x - \frac{f(x)}{f'(x)}$$

- Newton's Method is a fixed point iteration to find a fixed point of g.

Remarks:

- Fixed point problem are general class of problem.
- FPI is to try to capture a fixed point. The method may converge (hence work), or may diverge (hence fail).

- A root finding problem can be converted to a fixed point problem:

$$
\begin{aligned}
f(x) &= 0 \\
\text{Newton's } x = g(x) &= x - \frac{f(x)}{f'(x)} \\
x &= x - 100f(x) \\
x &= x + f(x)
\end{aligned}
$$

$$
\begin{aligned}
\cos(x) &= 2\sin(x) \\
& x - 2\sin(x) - \cos(x) + x
\end{aligned}
$$

Examples

1. To solve $x^3 + x - 1 = 0$ (two messages:1) root finding $\implies$ many fixed point problems, 2) some FPI works, some don't)

   Convert to fixed point problem:

$$
\begin{aligned}
x &= g_1(x) = 1 - x^3 && \times \\
x &= g_2(x) = \sqrt[3]{1-x} && \surd \\
x &= g_3(x) = \frac{1+2x^3}{1+3x^2} && \surd
\end{aligned}
$$

   g(x) = sx+T, s, T two constants

   Fixed point problem:

$$
\begin{aligned}
x &= g(x) = sX + T \\
x &= \frac{T}{1-S} && (s \neq 1)
\end{aligned}
$$

   Indeed:
   When $|s| > 1$, the scheme diverges
   When $|s| < 1$, the scheme converges

Some analysis:

$$x = g(x) = sx + /t$$

FPI:

$$
\begin{aligned}
x_j &= s(x_{j-1}) + T \\
\bar{x} &= s\bar{x} + T
\end{aligned}
$$

($\bar{x}$ is a fixed point of g)
Linear convergence:

$$
\begin{aligned}
(x_j - \bar{x}) &= s(x_{j-1} - \bar{x}) \\
&= s^2(x_{j-2} - \bar{x}) \\
&\cdots \\
&= s^j(x_0 - \bar{x})
\end{aligned}
$$

For $x_j \to \bar{x}$ as $j \to \infty \iff |s| < 1$

Theorem 2.5 (Fixed Point Iteration)**.** Assume g, g' are continuous. Let $\bar{x}$ be a fixed point of g and $s = |g'(\bar{x})| < 1$. Then the fixed point iteration $x_{j+1} = g(x_j)$ converges linearly, when x_0 is sufficiently close to $\bar{x}$.
That is

$$e_j = c_j e_{j-1}$$

and $c_j \to s$ as $j \to \infty$, here $e_j = |x_j - \bar{x}|$

Finally, we go back to Newton's method, which is a FPI, with $g(x) = x - \dfrac{f(x)}{f'(x)}$
Assume $f(\bar{x}) = 0$

$$\begin{aligned} g'(x) &= 1 - \left(\frac{f'(x)f'(x) - f(x)f''(x)}{(f'(x))^2}\right) \\ &= \frac{f(x)f''(x)}{(f'(x))^2} \\ s = |g'(x)| &= \left|\frac{f(\bar{x})f''(\bar{x})}{(f'(x))^2}\right| = 0 \end{aligned}$$

Therefore, Newton's method is a locally convergent FPI.
KEY: (for FPI works)
Contractive property: There exists a constant r: $0 < r < 1$, such that $|g(x_1) - g(x_2)| \leqslant r|x_1 - x_2|$ for any relevant x_1, x_2 (say in a neighborhood of a fixed point $\bar{x}$).

3 Solving systems of equations

To solve more than one equation together. We start with systems of linear equation.
In matrix- vector form:
To solve

$$Ax = b$$

for $x \in \mathbb{R}^n$
where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^n$ are given, and A is invertible, therefore, $x = A^{-1}b$

Definition 3.1. A is invertible, if there exists $B \in \mathbb{R}^{n\times n}$ such that

$$AB = BA = \mathbb{R}^{n\times n}$$

3.1 Iterative methods

$$Ax = b \; x^k \to x^{k+1}$$

Example

$$\begin{cases} 3u + v &= 5 \text{ First equation} \\ u + 2v &= 5 \text{ Second equation} \end{cases}$$

Matrix-Vector form:

$$A = \begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix} \; b = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

$Ax = b$, x unknown, start with an initial

$$x^0 = \begin{bmatrix} u^{(0)} \\ v^{(0)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

3.1.1 Jacobi Method

Idea: Solve the ith unknown from the ith equation:

$$u = \frac{5-v}{3} \; v = \frac{5-u}{2}$$

$$\begin{cases} u^{(1)} &= \frac{5-v^{(0)}}{3} = \frac{5}{3} \\ v^{(1)} &= \frac{5-u^{(0)}}{2} = \frac{5}{2} \end{cases}$$

$$x^{(1)} = \begin{bmatrix} \frac{5}{3} \\ \frac{5}{2} \end{bmatrix}$$

$$\begin{cases} u^{(2)} &= \frac{5-v^{(1)}}{3} = \frac{5}{6} \\ v^{(2)} &= \frac{5-u^{(1)}}{2} = \frac{5}{3} \end{cases}$$

$$x^{(2)} = \begin{bmatrix} \frac{5}{6} \\ \frac{5}{3} \end{bmatrix}$$

$$x^{(k)} \to x = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \text{ as } k \to \infty$$

If:

$$Ax = b \; x^k \to x^{k+1}$$

Example

$$\begin{cases} u + 2v &= 5 \text{ First equation} \\ 3u + v &= 5 \text{ Second equation} \end{cases}$$

Apply Jacobi method with

$$x^{(0)} = \begin{bmatrix} u^{(0)} \\ v^{(0)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

So

$$u = 5 - 2v \; v = 5 - 3u$$

Therefore,

$$x^{(1)} = \begin{bmatrix} u^{(1)} \\ v^{(1)} \end{bmatrix} = \begin{bmatrix} 5-2v^{(0)} \\ 5-3u^{(0)} \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \end{bmatrix} x^{(2)} \qquad = \begin{bmatrix} u^{(2)} \\ v^{(2)} \end{bmatrix} = \begin{bmatrix} 5-2v^{(1)} \\ 5-3u^{(1)} \end{bmatrix} = \begin{bmatrix} -5 \\ -10 \end{bmatrix}$$

$x^{(k)}$ diverges as $k \to \infty$

Definition 3.2. $A = (a_{ij}) \in \mathbb{R}^{n\times n}$ is strictly diagonally dominant if for each $1 \leqslant i \leqslant n$,

$$|a_{ij}| > \sum_{j\neq i} |a_{ij}|$$

That is, each diagonal entry dominates its row in the sense that it is greater in magnitude than the sum of magnitude of the remainder of entries in its row.

Jacobi method converges if A is strictly dominant. (Sufficient Condition)
Express Jacobi method in matrix-vector form:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \; b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \; x = \begin{bmatrix} u \\ v \end{bmatrix}$$

Scheme:

$$\begin{aligned} &\begin{cases} a_{11}u^k + a_{12}v^{k-1} &= b_1 \\ a_{21}u^{k-1} + a_{22}v^k &= b_2 \end{cases} \\ \iff &\begin{bmatrix} a_{11} & 0 \\ 0 & a_{22} \end{bmatrix} \begin{bmatrix} u^{(k)} \\ v^{(k)} \end{bmatrix} = -\begin{bmatrix} 0 & a_{12} \\ a_{21} & 0 \end{bmatrix} \begin{bmatrix} u^{k-1} \\ v^{k-1} \end{bmatrix} + b \\ \iff &Dx^k = b - (L+U)x^{k-1} \end{aligned}$$

Since

$$\begin{aligned} A &= \begin{bmatrix} a_{11} & 0 \\ 0 & a_{22} \end{bmatrix} + \begin{bmatrix} 0 & a_{12} \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ a_{21} & 0 \end{bmatrix} \\ &= D + U + L \\ A\bar{x} &= b \\ \iff (D+U+L)\bar{x} &= b \\ \iff D\bar{x} &= b - (U+L)\bar{x} \end{aligned}$$

Assume D is invertible,

$$x = D^{-1}(b - (U+L)x) = G(x)$$

(Fixed point problem)
Fixed point iteration:

$$x^{k+1} = G(x^{k+1})$$

Specially,

$$x^{k+1} = D^{-1}(b - (U+L)x^k)$$

Idea: To get the k^{th} iterate x^k, solve the i^{th} unknown from the i^{th} equation, based on x^{k-1}

3.1.2 Gauss-Seidel Method

Idea: Solve i^{th} unknown from the i^{th} equation, using the most updated values of the unknowns.
Revisit example:

$$Ax = b \; x^k \to x^{k+1}$$

Example

$$\begin{cases} 3u + v &= 5 \text{ First equation} \\ u + 2v &= 5 \text{ Second equation} \end{cases}$$

$$x^0 = \begin{bmatrix} u^{(0)} \\ v^{(0)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$u = \frac{5-v}{3} \; v = \frac{5-u}{2}$$

$$\begin{cases} u^{(1)} &= \dfrac{5-v^{(0)}}{3} = \dfrac{5}{3} \\ v^{(1)} &= \dfrac{5-u^{(1)}}{2} = \dfrac{5}{2} \end{cases}$$
$$\begin{cases} u^{(2)} &= \dfrac{5-v^{(1)}}{3} = \dfrac{10}{9} \\ v^{(2)} &= \dfrac{5-u^{(2)}}{2} = \dfrac{35}{18} \end{cases}$$

Matrix -Vector form of GS:

$$\begin{aligned} &\begin{cases} a_{11}u^k + a_{12}v^{k-1} &= b_1 \\ a_{21}u^k + a_{22}v^k &= b_2 \end{cases} \\ \iff &\begin{bmatrix} a_{11} & 0 \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} u^{(k)} \\ v^{(k)} \end{bmatrix} = - \begin{bmatrix} 0 & a_{12} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u^{k-1} \\ v^{k-1} \end{bmatrix} + b \\ \iff &(D+L)x^k = b - Ux^{k-1} \end{aligned}$$

Fixed point problem:

$$\begin{aligned} (D+L)\bar{x} = b - U\bar{x} \\ \bar{x} = (D+L)^{-1}(b - U\bar{x}) \\ = G(\bar{x}) \end{aligned}$$

GS method:
To solve $A\bar{x} = b$, for $\bar{x} \in \mathbb{R}^n$, where $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$, and $A = D + L + U$
Gauss - Seidel method is

$$(D+L)x^{(k)} = b - Ux^{(k-1)}$$

where D is invertible,

$$x^{(k)} = D^{-1}(b - Ux^{(k-1)} - Lx^{(k)})$$

Idea: To solve the i^{th} unknown from the i^{th} equation, using the most recently updated unknowns.
To solve

$$Ax = b$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^n$ are given.
M is invertible, a splitting of A:

$$\begin{aligned} A &= M - N \\ A\bar{x} &= b \\ \iff M\bar{x} &= Nx + b \\ \iff \bar{x} &= M^{-1}(Nx + b) \\ &= G(\bar{x}) \end{aligned}$$

For Jacobi method:

$$M = D \; N = -(L+U)$$

Gauss-Seidel mdethod:

$$M = D + L \; N = -U$$

(fixed point iteration for the related $\bar{x} = G(\bar{x})$)

General form of these two method:
Problem :

$$\bar{x} = B\bar{x} + d \tag{1}$$

Numerical method

$$x^k = Bx^{k-1} + d \tag{2}$$

Question on convergence:
(2) - (1)

$$\begin{aligned} x^k - x &= B(x^{k-1} - x) \\ z^k &= Bz^{k-1} \\ &= B^2 z^{k-2} \\ &= B^k z^0 \end{aligned}$$

$z_i^{(k)} \to 0$ as $k \to \infty$ iff $|\lambda_i| < 1$

That is , when $B = \begin{bmatrix} \lambda_1 & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \lambda_n \end{bmatrix}$, then $x^k = Bx^{k-1} + d$

It converges to $\bar{x}$, (the solution of $\bar{x} = B\bar{x} + d$), if all eigenvalues of B are bounded by 1 in their absolute values.
Spectral radius of B:

$$\begin{aligned} &= \max_{1 \leqslant i \leqslant n} |\lambda_i| \\ &= \rho(B) \end{aligned}$$

λ_i is an eigrnvalue of B
Converge $\iff$ $\rho(B) < 1$

Theorem 3.1. Let $B \in \mathbb{R}^{n\times n}$, the $x^{(k+1)} = Bx^k + d$ converges to a solution of $\bar{x} = B\bar{x} + d$ for any given d and initial x^0 iff the spectral radius $\rho(B) < 1$

Theorem 3.2. Given $A \in \mathbb{R}^{n\times n}$, If A is strictly diagonally dominant, then

- A is invertible
- Jacobi and Gauss-Seidel methods applied to $A\bar{x} = b$ converges to the unique solution for any $b \in \mathbb{R}^n$, and with any initial x^0

  For the related :

$$\begin{aligned} &x^{k+1} = Bx^k + d \\ &\rho(B) < 1 \end{aligned}$$

3.2 Direct Method

Gaussian Elimination (GE) and LU factorization:
Given $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$. A is invertible, we want to solve $Ax = b$ for $x \in \mathbb{R}^n$
They're methods that involve finitely many operations.
A special case: when A is upper triangular, that is $a_{ij} = 0$ for any $i > j$.
The algorithm process is back substitution: To solve upper triangular system:

In general, $x = \begin{bmatrix} x_1 \\ x_2 \cdots x_n \end{bmatrix}$,

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\cdots \\ a_{n-1n-1}x_{n-1} + a_{n-1n}x_n &= b_{n-1} \\ a_{nn}x_n &= b_n \end{aligned}$$

To solve using back substitution:

$$\begin{aligned} x_n &= \frac{b_n}{a_{nn}} && \text{Cost } = 1 \\ x_{n-1} &= \frac{b_{n-1} - a_{n-1n}x_n}{a_{n-1n-1}} && \text{Cost } = 3 \\ &\cdots \\ x_2 &= \frac{b_2 - a_{23}x_3 - \cdots - a_{2n}x_n}{a_{22}} && \text{Cost } = 2(n-1) - 1 \\ x_1 &= \frac{b_1 - a_{12}x_2 - \cdots - a_{1n}x_n}{a_{11}} && \text{Cost } = 2n - 1 \end{aligned}$$

Computational Complexity of B.S:

$$\begin{aligned} & 1 + 3 + 5 + \cdots + (2n-1) \\ =& \sum_{j=1}^{n} 2j - 1 \\ =& 2\sum_{j=1}^{n} j - n \\ =& n(n+1) - n \\ =& n^2 \end{aligned}$$

Back substitution:
$A = (a_{ij} \in \mathbb{R}^{n\times n})$: Upper triangular $(a_{ij} = 0 \forall i > j)$

$$\bar{b} = \begin{bmatrix} b_1 \\ b_2 \\ \cdots b_n \end{bmatrix} \in \mathbb{R}^n$$

To solve $A\bar{x} = b$ for $x \in \mathbb{R}^n$ Algorithm:

for $i = n : -1 : 1$ **do**
  for $j = i + 1 : n$ **do**
    $b(i) = b(i) - a(i, j) * x(j)$
  end
  $x(i) = \dfrac{b(i)}{a(ij)}$
end

Another special case: A is lower triangular, that is, $a_{ij} = 0$ for any $i < j$.
Forward substitution: Cost n^2.

General form:

$$[A|b]$$

Tableau form (augmented matrix):

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & | & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & | & b_2 \\ \cdots & \cdots & \cdots & \cdots & | & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & | & b_n \end{bmatrix}$$

We want to zero out the entries below the main diagonal:
Start with first column:

$$Row_i - \frac{a_{i1}}{a_{11}} Row_1 \to Row_i$$

$$\iff a_{i1} - \frac{a_{i1}}{a_{11}} a_{11} = 0 \; a_{i2} - \frac{a_{i1}}{a_{11}} a_{12}$$

(all zeros, no need to compute)
Cost:

$$\begin{bmatrix} 0 & & & & & \\ 2n+1 & 0 & & & & \\ 2n+1 & 2(n-1)+1 & 0 & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ & & & 2\cdot 3+1 & 0 & \\ 2n+1 & 2(n-1)+1 & \cdots & 2\cdot 3+1 & 2\cdot 2+1 & 0 \end{bmatrix}$$

Total cost will be

$$\begin{aligned} \# &= \sum_{j=1}^{n-1} (2(j+1)+1)\cdot j \\ &= \sum_{j=1}^{n-1} (2j+3)\cdot j \\ &= 2\sum_{j=1}^{n-1} j^2 + 3\sum_{j=1}^{n-1} j \\ &= \frac{2}{3}n^3 + \frac{1}{2}n^2 - \frac{7}{6}n \\ &\approx \frac{2}{3}n^3 \\ &= \mathcal{O}(n^3) \end{aligned}$$

This leads to the algorithm **Gaussian Elimination (GE)** that converts a full system to an upper-triangular system. We then call back substitution, this altogether solves $Ax = b$.
Cost

$$\begin{aligned} GE &\approx \frac{2}{3}n^3 \\ back\ substitution &= n^2 \\ add\ up &\approx \frac{2}{3}n^3 \end{aligned}$$

Gaussian Elimination

$A = (a_{ij}) \in \mathbb{R}^{n\times n}$ $b \in \mathbb{R}^n$ **for** $i = 1 : n-1$ **do**
  if $|a(j,j)| < eps$ **then**
    Error
  end
  for $i = j+1 : n$ **do**
    $z = \dfrac{a(i,j)}{a(j,j)}$**for** $k = j+1 : n$ **do**
      $a(i,k) = a(i,k) - z * a(j.k)$
    end
    $b(i) = b(i) - z * b(j)$
  end
end

To reduce A to an upper triangular U, namely, $A \to U$, essentially is to find the LU factorization of A, $A = LU$
U : Upper-triangular L : Lower-triangular, where the main diagonal entries are 1

Revisit matrix multiplication: Given

$$A = (a_{ij}) \in \mathbb{R}^{n\times n}$$
$$B = (b_{ij}) \in \mathbb{R}^{n\times n}$$

We know $C = AB = (c_{ij}) \in \mathbb{R}^{n\times n}$, where $c_{ij} = \sum_{k=1}^n a_{ik}b_{kj}$
Let β_i be the ith row of B:
Take a look at ith row of C:

$$\begin{aligned}\begin{bmatrix} c_{i1} & c_{i2} & \cdots & c_{in}\end{bmatrix} &= \begin{bmatrix} \sum_{k=1}^n a_{ik}b_{k1} & \sum_{k=1}^n a_{ik}b_{k2} & \cdots \sum_{k=1}^n a_{ik}b_{kn}\end{bmatrix} \\ &= a_{i1}\beta_1^T + a_{i2}\beta_2^T + \cdots + a_{in}\beta_n^T\end{aligned}$$

ith row of C is a linear combination of rows of B, with the coefficients from ith row of A.
To understand GE, go back to example $Ax = b$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 4 & 1 \\ 5 & -1 & -1 \end{bmatrix} \; b = \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix} \; x = \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

Tableau form:

$$\begin{bmatrix} 1 & 1 & 1 & | & 1 \\ 2 & 4 & 1 & | & 0 \\ 5 & -1 & -1 & | & 2 \end{bmatrix}$$
$$\xrightarrow{R_2-2R_1} \begin{bmatrix} 1 & 1 & 1 & | & 1 \\ 0 & 2 & -1 & | & -2 \\ 5 & -1 & -1 & | & 2 \end{bmatrix}$$
$$\xrightarrow{R_3-5R_1} \begin{bmatrix} 1 & 1 & 1 & | & 1 \\ 0 & 2 & -1 & | & -2 \\ 0 & -6 & -6 & | & -3 \end{bmatrix}$$
$$\xrightarrow{R_3-(-3)R_2} \begin{bmatrix} 1 & 1 & 1 & | & 1 \\ 0 & 2 & -1 & | & -2 \\ 0 & -6 & -9 & | & -9 \end{bmatrix}$$

Express every step using matrix notation:

$$\begin{bmatrix}1&0&0\\0&1&0\\0&3&1\end{bmatrix}\begin{bmatrix}1&0&0\\0&1&0\\-5&0&1\end{bmatrix}\begin{bmatrix}1&0&0\\-2&1&0\\0&0&1\end{bmatrix}A=U=\begin{bmatrix}1&1&1\\0&2&-1\\0&0&-9\end{bmatrix}$$

$$\begin{aligned}\therefore A&=\begin{bmatrix}1&0&0\\-2&1&0\\0&0&1\end{bmatrix}^{-1}\begin{bmatrix}1&0&0\\0&1&0\\-5&0&1\end{bmatrix}^{-1}\begin{bmatrix}1&0&0\\0&1&0\\0&3&1\end{bmatrix}^{-1}U\\&=\begin{bmatrix}1&0&0\\2&1&0\\0&0&1\end{bmatrix}\begin{bmatrix}1&0&0\\0&1&0\\5&0&1\end{bmatrix}\begin{bmatrix}1&0&0\\0&1&0\\0&-3&1\end{bmatrix}U\\&=\begin{bmatrix}1&0&0\\2&1&0\\5&-3&1\end{bmatrix}U\end{aligned}$$

Once we have LU factorization of A, namely $A=LU$, then $Ax=b$

$$\begin{aligned}&\iff LUx=b \text{ Cost } =\frac{2}{3}n^3\\&\iff\begin{cases}Ly &= b \text{ For } y \text{ Cost } = n^2\\Ux &= y \text{ For } x \text{ Cost } = n^2\end{cases}\end{aligned}$$

GE method does not always work, typical step

$$Row_i-\frac{a_{ij}}{a_{jj}}Row_j\to Row_i$$

$a_{jj}=0\to$ fail.
Remedy: Pivoting: is preferred to have the highest magnitude, and this is to have a less sensitive problem to rounding error.
Two purpose:

- GE cam continue
- The scheme will be less sensitive to rounding error

3.3 Sensitivity of $Ax=b$ and condition number of A

As some preparation: How to measure the size of a vector and how to measure the difference?

$$x=\begin{bmatrix}x_1\\x_2\\\cdots\\x_n\end{bmatrix}\in\mathbb{R}^n y=\begin{bmatrix}y_1\\y_2\\\cdots\\y_n\end{bmatrix}\in\mathbb{R}^n$$

$$\begin{aligned}||x||_2&=\left(\sum_{j=1}^n|x_j|^2\right)^{\frac{1}{2}}\\||x||_\infty&=\max_{1\leqslant j\leqslant n}|x_j|\end{aligned}$$

Difference:

$$||x-y||_2 = \left(\sum_{j=1}^{n} |x_j - y_j|^2\right)^{\frac{1}{2}}$$
$$||x-y||_\infty = \max_{1\leqslant j\leqslant n} |x_j - y_j|$$

A norm $||\cdot||$ of $\mathbb{R}^n$ needs 3 properties:

- $||x|| \geqslant 0$ and $||x|| = 0 \iff x = 0$
- $||ax|| = |a|||x||$, $a \in \mathbb{R}$, $x \in \mathbb{R}^n$
- $||x+y|| \leqslant ||x|| + ||y||$, $\forall x, y \in \mathbb{R}^n$
  $\iff ||x-z|| \leqslant ||x-y|| + ||y-z||$

How to measure a matrix? Similar three properties required.
One example:

$$||A||_\infty = \max_{1\leqslant i\leqslant n} \sum_{j=1}^{n} |a_{ij}|$$

sum of absolute value of ith row.
Sensitivity of $Ax = b$ is determined by the condition number of A. $cond(A) = ||A||||A^{-1}||$ with $\infty-$norm: $cond(A,\infty) = ||A||_\infty ||A^{-1}||_\infty$
Overall: The larger the condition number of A is, the more sensitive solving $Ax = b$ is with respect to rounding error. (Regardless of $||\cdot||$)
Generally: To solve $Ax = b$ on a computer, and get x_c, then $\dfrac{||x-x_c||}{||x||} \approx \epsilon_{mach} cond(A)$
Double precision : $\epsilon \approx 10^{-16}$

Demo: Example:

$$A = \begin{bmatrix} 1 & 1 \\ 1+\epsilon & 1 \end{bmatrix} \; 0 < ep < 1$$

$$cond(A,\infty) = ||A||_\infty ||A^{-1}||_\infty$$
$$||A||_\infty = 2+\epsilon$$
$$A^{-1} = -\frac{1}{\epsilon}\begin{bmatrix} 1 & -1 \\ -1-\epsilon & 1 \end{bmatrix}$$
$$||A^{-1}||_\infty = \frac{2+\epsilon}{\epsilon}$$
$$cond(A,\infty) = \frac{(2+\epsilon)^2}{\epsilon} \approx \frac{4}{\epsilon}$$

Example: Hilber matrix:

$$a_{ij} = \frac{1}{i+j-1}$$
$$H = hilb(n)$$

$$\begin{aligned} Ax &= b \\ BAx &= Bb \\ cond(BA) &<<< cons(A) \end{aligned}$$

B:invertible, it is the "preconditioner", best: $B = A^{-1}$ is the best.

3.4 Symmetric positive definite matrix and Cholasky factorization

:

Definition 3.3. $A \in \mathbb{R}^{n\times n}$, A is symmetric if $A = A^T$. A is positive definite if $x^T Ax > 0$ for any nonzero $x \in \mathbb{R}^n$.

Theorem 3.3. Let $A \in \mathbb{R}^{n\times n}$ be symmetric, A is positive definite if and only if all eigenvalues of A are positive.

Proof: Idea: Take x to be an eigenvector of an eigenvalue λ:
Then

$$\begin{aligned} x^T Ax &= x^T \lambda x \\ &= \lambda x^T x \\ &= \lambda ||x||_2^2 > 0 \\ \implies \lambda &> 0 \end{aligned}$$

Theorem 3.4 (Negative results)**.** Assume $A \in \mathbb{R}^{n\times n}$ is symmetric:

- A is not positive definite if some diagonal entry is negative or zero.
- A is not positive definite if the largest entry of A, in absolute value, is off the diagonal.
- A is not positive definite if $det(A) \leqslant 0$

Proof:

- Take $x = e_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$, $e_i^T Ae_i = a_{ii}$
- Take

$$\begin{aligned} x &= e_i + e_j \\ x^T Ax &= a_{ii} + 2a_{ij} + a_{jj} > 0 \\ x &= e_i - e_j \\ x^T Ax &= a_{ii} - 2a_{ij} + a_{jj} > 0 \\ -2a_{ij} &> -(a_{ii} + a_{jj}) \\ 2a_{ij} &< a_{ii} + a_{jj} \\ |a_{ij}| &< \frac{a_{ii} + a_{jj}}{2} \end{aligned}$$

- $$\det(A) = \lambda_1\lambda_2\cdots\lambda_n$$

  Due to $\det(\lambda I - A) = 0$

  $$(\lambda-\lambda_1)(\lambda-\lambda_2)\cdots(\lambda-\lambda_n) = 0$$

  Take

  $$\begin{aligned}\lambda &= 0\\ \det(-A) &= \Pi_{i=1}^n \lambda_i(-1)^n\\ \det(A) &= \Pi_{i=1}^n \lambda_i\end{aligned}$$

Theorem 3.5. Let $A \in \mathbb{R}^{n\times n}$ be SPD, then it always has cholesky factorization:

$$A = R^T R$$

R is upper-triangular, and $r_{ii} > 0$, $i = 1, 2\cdots n$

Recall: Cost of GE/LU factorization: $\frac{2}{3}n^3$
Cost of Cholesky factorization: $\frac{1}{3}n^3$ (half of LU/GE) due to symmetry.
Given $A \in \mathbb{R}^{n\times n}$, it's SPD, Given $b \in \mathbb{R}^n$, to solve $Ax = b$

- Step 1 : Find R such that

  $$A = R^T R \qquad\qquad \frac{1}{3}n^3$$

- Step 2: Solve

  $$R^T y = b \qquad\qquad n^2$$

- Step 3: Solve x from

  $$Rx = y \qquad\qquad n^2$$

How to find Cholesky factorization:
Example:

$$A = \begin{bmatrix} 2 & 2\\ 2 & 5\end{bmatrix}$$

$$\begin{aligned}A &= \begin{bmatrix} 2 & 2\\ 2 & 5\end{bmatrix}\\ &= \begin{bmatrix} r_{11} & 0\\ r_{12} & r_{22}\end{bmatrix}\begin{bmatrix} r_{11} & r_{12}\\ 0 & r_{22}\end{bmatrix}\\ &= \begin{bmatrix} r_{11}^2 & r_{11}r_{12}\\ r_{12}r_{11} & r_{11}^2 + r_{22}^2\end{bmatrix}\end{aligned}$$

Compare entry by entry (1st column, the 2nd column, use symmetry, so not all entries need to be examined).

$$\begin{aligned}
2 &= r_{11}^2 && (r_{11} > 0)\\
\implies r_{11} &= \sqrt{2}\\
2 &= r_{12}r_{11}\\
r_{12} &= \sqrt{2}\\
5 &= r_{11}^2 + r_{22}^2\\
r_{22} &= \sqrt{3}\\
R &= \begin{bmatrix} \sqrt{2} & \sqrt{2} \\ 0 & \sqrt{3} \end{bmatrix}
\end{aligned}$$

Remark: The procedure in the example can be extended to a SPD matrix $A \in \mathbb{R}^{n\times n}$ (any n)

3.5 Nonlinear systems of equations

We want to solve

$$\begin{aligned}
f_1(x_1, x_2 \cdots x_n) &= 0\\
f_2(x_1, x_2 \cdots x_n) &= 0\\
&\cdots\\
f_n(x_1, x_2 \cdots x_n) &= 0
\end{aligned}$$

and solve for $x_1, x_2 \cdots x_n$
Recall: TO solve $f(x) = 0$, start from x_0

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \cdots f(x_0) + f'(x_0)(x - x_0) \qquad = \hat{f}(x)$$

(Linear polynomial of degree 1)
We instead solve $\hat{f}(x) = 0 \implies$ this gives x, newton's method
Newton: Approximate $f(x)$ by linear polynomial.
Solve

$$\begin{cases} \hat{f}(x,y) &= 0\\ \hat{g}(x,y) &= 0 \end{cases}$$

Let

$$\begin{aligned}
F(x,y) &= \begin{bmatrix} f(x,y) \\ g(x,y) \end{bmatrix}\\
DF(x,y) = J_F(x,y) &= \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \end{bmatrix}
\end{aligned}$$

Approximation Step:

$$F(x,y) \approx F(x_0,y_0) + J_F(x_0,y_0)\begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix}$$

Algorithm (Newton's Method):

Start with an initial $\begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$, for $k = 0, 1, 2 \cdots$ Solve $S \in \mathbb{R}^2$ from
$J_F(x_k, y_k)s = -F(x_k, y_k)$
$\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \end{bmatrix} + s$
Stop if $||s|| < tol$, or $k > k_{max}$

Core:Linearization

4 Interpolation and data fitting

$f(x) \approx \hat{f}(x)$ based on several points on the graph:
$\hat{f}(x)$: Simpler form, an approximation of $f(x)$:

- Extract information of data
- Approximate functions

4.1 Interpolation : Global

In general, given $(x_1, y_1), (x_2, y_2), \cdots (x_{n+1}, y_{n+1})$

Definition 4.1. A function $y = p(x)$ interpolates the points given above if $y_j = P_{x_j}$, $j = 1, 2, \cdots n+1$

For now, we assume $P(x)$ is a polynomial of degree n.
Problem:

$$\begin{cases} \text{Given } (x_j, y_j) & j = 1, 2, \cdots n+1 \\ \text{find } P_n(x) \end{cases}$$

such that $P_n(x_j) = y_j$, $j = 1, 2, \cdots n+1$

4.1.1 Direct method

Look for

$$P_n(x) = a_0 + a_1 x + \cdots + a_n x^n$$

such that $P_n(x_j) = y_j$, $j = 1, 2, \cdots n+1$

$$\begin{aligned} a_0 + a_1 x_1 + a_2 x_1^2 + \cdots + a_n x_1^n &= y_1 \\ a_0 + a_1 x_2 + a_2 x_2^2 + \cdots + a_n x_2^n &= y_2 \\ &\cdots \\ a_0 + a_1 x_{n+1} + a_2 x_{n+1}^2 + \cdots + a_n x_{n+1}^n &= y_{n+1} \end{aligned}$$

In matrix- vector form:

$$a = \begin{bmatrix} a_0 \\ a_1 \\ \cdots \\ a_n \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \cdots y_{n+1} \end{bmatrix}$$

we have

$$A = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ & \cdots & & \cdots & \\ 1 & x_{n+1} & x_{n+1}^2 & \cdots & x_{n+1}^n \end{bmatrix}$$

(Vandermonde Matrix)
$Aa = y$, A is square.
Unique solvability $\iff$ A is invertible $\iff$ $\det(A) \neq 0$

Lemma 4.1.

$$\det(A) = \Pi_{1 \leqslant i < j \leqslant n+1}(x_j - x_i)$$

Theorem 4.2. Given $\{(x_j, y_j)\}_{j=1}^{n+1}$ the interpolating polynomial $P_n(x)$ exists uniquely $\iff$ $\{x_j\}_{j=1}^{n+1}$ are distinct.

Discussion:

- Given $\{(x_i, y_i)\}_{i=1}^{n+1}$, $\{x_i\}_{i=1}^{n+1}$ are distinct, $P_n(x)$ is the unique polynomial interpolant of degree n.

$$\begin{aligned} \implies P_{n+1}(x) &= P_n(x) + c(x - x_1)(x - x_2) \cdots (x - x_{n+1}) \\ P_{n+1}(x) &= P_n(x) + c()|x = x_i \end{aligned}$$

  c: any constant, interpolants of polynomial of higher degree.
  $P_{n+1}(x)$: Infinitely many

- Given 3 points:

  Polynomial of degree 1 ¡- Special polynomials of degree 2.

$$a_0 + a_1 x + a_2 x^2$$

  a_1, a_2 can be zero.

- In practice, Vandermonde matrix can be ill-conditioned.

  Example:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_{n+1} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{3} \\ \cdots \\ \frac{1}{n} \end{bmatrix}$$

  $A = Vander(x)$

$$\begin{aligned} cond(A) &\approx 753 & n &= 4 \\ &\approx 2,4 \times 10^5 & n &= 6 \\ &\approx 1.52 \times 10^8 & n &= 8 \\ &\approx 1.59 \times 10^{11} & n &= 10 \end{aligned}$$

4.1.2 Lagrange Approach

Problem:

$$\{(x_i, y_i\}_{i=1}^{n+1} \text{ look for } P_n(x)$$
$$\{(x_i\}_{i=1}^{n+1}\text{distinct}$$
$$P_n(x_i) = y_i$$

$n = 1$: Two points:

$$\begin{aligned}
P_1(x) &= y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) \\
&= y_1 \left(1 - \frac{x - x_1}{x_2 - x_1}\right) + y_2 \left(\frac{x - x_1}{x_2 - x_1}\right) \\
&= y_1 \left(\frac{x - x_2}{x_1 - x_2}\right) + y_2 \left(\frac{x - x_1}{x_2 - x_1}\right) \\
&= y_1 l^{(1)}(x) + y_2 l^{(2)}(x)
\end{aligned}$$

Features of $l^{(1)}(x)$, $l^{(2)}(x)$

- Polynomial of degree 1
- $$\begin{aligned}
l^{(1)}(x_1) &= 1 \\
l^{(1)}(x_2) &= 0 \\
l^{(2)}(x_1) &= 0 \\
l^{(2)}(x_2) &= 1
\end{aligned}$$

$$\iff l^{(i)}(x_i) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

For general n,

$$\begin{aligned}
P_n(x) &= y_1 l^{(1)}(x) + y_2 l^{(2)}(x) + \cdots + y_{n+1} l^{(n+1)}(x) \\
&= \sum_{j=1}^{n+1} y_j l^{(j)}(x)
\end{aligned}$$

- $l^{(j)}(x)$: Polynomial of degree n
- $l^{(j)}(x_i) = \delta_{ij} \implies$ uniquely exists, as it interpolate $(x_1, 0) \cdots (x_{i-1}, 0), (x_{i+1}, 0) \cdots (x_{n+1}, 0)$

$$l^{(j)}(x) = \frac{(x - x_1) \cdots (x - x_{j-1})(x - x_{j+1}) \cdots (x - x_{n+1})}{(x_j - x_1) \cdots (x_j - x_{j-1})(x_j - x_{j+1}) \cdots (x_j - x_{n+1})}$$

"Lagrange Polynomial"

Remark:

- In approach 1:
$$P_n(x) = a_0 + a_1 x + \cdots + a_n x^n$$
$1, x, x^2 \cdots x^n$: Monomial basis of polynomial degree up to n.
- In approach 2: A different basis: "Lagrange basis" $l^{(1)}(x), l^{(2)}(x), \cdots l^{(n+1)}(x)$
- Also if
$$P_n(x) = \sum_{j=1}^{n+1} y_j l^{(j)}(x)$$
then
$$P_n(x) = \sum_{j=1}^{n+1} y_j \delta_{ij} = y_j$$
$\implies y_i = P_n(x_i)$
Coefficients of lagrange representation give function values at x_j, $j = 1, 2, \cdots$

4.1.3 Newton's divided difference

Recall: Given 1 point (x_1, y_1), then $P_0(x_1) = y_1$
Given two points $(x_1, y_1), (x_2, y_2)$,

$$\begin{aligned} P_1(x) &= y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) \\ &= P_0(x) + a_1(x - x_1) \end{aligned}$$

$(a_1(x - x_1)$ is correction)
To verify:

$$\begin{aligned} P_1(x_1) &= P_0(x_1) + a_1(x_1 - x_1) = P_0(x_1) \\ P_1(x_2) &= P_0(x_2) + a_1(x_2 - x_1) \\ &= y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x_2 - x_1) = y_2 \end{aligned}$$

Now we add one more point (x_3, y_3)

$$P_2(x) = P_1(x) + a_2(x - x_1)(x - x_2)$$

we can see

$$\begin{aligned} P_2(x_1) &= P_1(x_1) + 0 = y_1 \\ P_2(x_2) &= P_1(x_2) + 0 = y_2 \\ P_2(x_3) &= P_1(x_3) + a_2(x_3 - x_1)(x_3 - x_2) = y_3 \\ \implies a_2 &= \frac{y_3 - P_1(x_3)}{(x_3 - x_1)(x_3 - x_2)} \\ &= \frac{\frac{y_3 - y_2}{x_3 - x_2} - \frac{y_2 - y_1}{x_2 - x_1}}{x_3 - x_1} \end{aligned}$$

(Related to "Divided Difference")
Given $\{(x_j, y_j)\}_{j=1}^n$, n points, $\{x_j\}_{j=1}^n$ distinct
$P_{n-1}(x)$: Unique interpolating polynomial of degree $n - 1$

Now, suppose there is an extra point (x_{n+1}, y_{n+1}), $x_{n+1} \neq x_j,\ j = 1, 2, \cdots n$:

$$P_n(x) = P_{n-1}(x) + a_n(x - x_1) \cdots (x - x_n)$$

One can check:

$$\begin{aligned}
P_n(x_i) &= P_{n-1}(x_i) + 0 \\
&= y_i && i = 1 \cdots n \\
P_n(x_{n+1}) &= P_{n-1}(x_{n+1}) + a_n(x_{n+1} - x_1) \cdots (x_{n+1} - x_n) \\
&= y_{n+1} \\
\implies a_n &= \frac{y_{n+1} - P_{n-1}(x_{n+1})}{(x_{n+1} - x_1) \cdots (x_{n+1-x_n})}
\end{aligned}$$

Introduce some notation:
Given $\{(x_i, y_i)\}_{i=1}^{n+1}$
Define divided difference, recursively:

$$\begin{aligned}
g[x_j] &= y_j && \forall j \\
g[x_j, x_{j+1}] &= \frac{g(x_{j+1}) - g(x_j)}{x_{j+1} - x_j} && \forall j \\
g[x_j, x_{j+1}, x_{j+2}] &= \frac{g[x_{j+1}, x_{j+2}] - g[x_j, x_{j+1}]}{x_{j+2} - x_j} && \forall j \\
&\cdots \\
g[x_j, x_{j+1}, x_{j+2}, \cdots x_{j+k}] &= \frac{g[x_{j+1}, \cdots x_{j+k}] - g[x_j \cdots x_{j+k-1}]}{x_{j+k} - x_j} && \forall j
\end{aligned}$$

What we got so far $P_0(x), P_1(x), P_2(x) \cdots$ can be written in terms of divided difference

$$\begin{aligned}
P_0(x) &= y_1 = g[x_1] \\
P_1(x) &= P_0(x) + g[x_1, x_2](x - x_1) \\
&= g[x_1] + g[x_1, x_2](x - x_1) \\
P_2(x) &= P_1(x) + g[x_1, x_2, x_3](x - x_1)(x - x_2)
\end{aligned}$$

In general

$$P_n(x) = \sum_{j=1}^{n+1} g[x_1, x_2 \cdots x_j](x - x_1)(x - x_2) \cdots (x - x_{j-1})$$

Divided difference can be organized and computed via a table:
Given $(x_j, y_j), i = 1, 2, 3$

x_1	$g[x_1]$			
x_2	$g[x_2]$	$g[x_1, x_2]$		
x_3	$g[x_3]$	$g[x_2, x_3]$	$g[x_1, x_2, x_3]$	
x_4	$g[x_4]$	$g[x_3, x_4]$	$g[x_2, x_3, x_4]$	$g[x_1, x_2, x_3, x_4]$

To add one new point $[x_4, y_4]$, we just need to add one more row in the table.
The diagonal values contribute to the coefficients of the polynomial.

4.2 Interpolating error

Suppose $\{x_j, y_j\}_{j=1}^{n+1}$ comes from sampling $y = f(x)$. $\{(x_j)\}_{j=1}^{n+1}$ are distinct, $y_j = f(x_j)$

$P_n(x)$: Interpolating polynomial of degree n.
Interpolating error:

$$|f(x) - P_n(x)|$$

Theorem 4.3.

$$f(x) - P_n(x) = \frac{(x - x_1)(x - x_2)\cdots(x - x_{n+1})}{(n+1)!} f^{(n+1)}(c)$$

c is depend on x and is some number from $[\min(x, x_1, \cdots, x_{n+1}), \max(x, x_1, \cdots, x_{n+1})]$

Runge's Phenomenon

1. Related to global interpolation based on equi-distanced points
2. Non-equal distanced sample points, "Chebyshev nodes"
   Local Interpolation

4.3 Interpolation: Local

Given $\{x_j, y_j\}_{j=1}^{n+1}$ with $x_1 < x_2 \cdots < x_n < x_{n+1}$
Recall: An interpolant $y = P(x)$ of the data, $P(x_i) = y_i$, $i = 1, 2, \cdots n+1$
So far: $P(x)$ is polynomial of degree n,(global)
We have consider local interpolation

4.3.1 Piecewise linear interpolation

Look for $g(x)$ with $g(x) = g_j(x)$ on $[x_j, x_{j+1}]$, such that $g_j(x)$ is linear, (a polynomial of degree 1)., and $\begin{cases} g_j(x_j) &= y_j \\ g_j(x_{j+1}) &= y_{j+1} \end{cases}$
To find $g(x)$, by construction, $g_j(x) = y_j + \dfrac{y_{j+1} - y_j}{x_{j+1} - x_j}(x - x_j)$
$\implies$ existence, uniqueness $\surd$
In practive, lagrange-type basis is used to represent $g(x)$.
$\Phi_i(x)$: piecewise linear (hat function)

$$\Phi_i(x) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} = \delta_{ij}$$

$$g(x) = \sum_{j=1}^{n+1} y_j \Phi_j(x)$$

Discussion:

1. $g(x)$ is continuous, it's not differentiable at x_j
2. Same x_j case : local interpolant will work, global one does not.

4.3.2 Cubic Spline

Goal: To get better smoothness at x_j
Again, given $\{x_j, y_j\}_{j=1}^{n+1}$, $x_1 < x_2 < x_3 \cdots < x_{n+1}$
We look for $g(x)$ is a cubic polynomial

- Property1:
$$\begin{cases} g_j(x_j) & = y_j \\ g_j(x_{j+1}) & = y_{j+1} \end{cases}$$

- Property2:
$$g'_j(x_j) = g'_{j-1}(x_j)$$
$j = 2, \cdots n$

- Property3:
$$g''_j(x_j) = g''_{j-1}(x_j)$$
$j = 2, \cdots n$

$g(x), g'(x), g''(x)$ are continuous.
Unique existence:

$$g_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$$

n intervals: 4n unknowns,
Conditions: $2n + n - 1 + n - 1$
2 condition short: No uniqueness.
Two more conditions can be imposed at both end of the data interval:

1. Choice 1: Natural Splines:
$$g''_1(x_1) = 0,\ g''_n(x_{n+1}) = 0$$

2. Choice 2: Clamped Splines:
$$g'_1(x_1) = \alpha,\ g'_n(x_{n+1}) = \beta$$
, α, β are given

3. Choice 3: Not a knot Splines:
$$g'''_1(x_2) = g'''_2(x_2),\ g'''_{n-1}(x_n) = g'''_n(x_n)$$

With any of the choice above, the cubic spline interpolation can be uniquely determined.
Remark: In general, with more data, all the unknowns can be expressed in terms of $\{c_j\}_{j=1}^{n}$ and $c = \begin{bmatrix} c_1 \\ \vdots \\ c_{n+1} \end{bmatrix}$ satisfies a tri-diagonal system, Ac= given. ($\mathcal{O}(n)$)

4.3.3 Interpolation error

Use piecewise linear interpolation as an example.
Assume $\{(x_j, y_j\}_{j=1}^{n+1}$ forms a given function, namely $y_j = f(x_j),$. We want to bound $|f(x) - g(x)|$, here $g(x)$ is a piecewise linear interpolation.
Recall

$$g(x) = g_j(x) \text{ on } [x_j, x_{j+1}]$$
$$g_j(x) \text{ is a polynomial of degree 2}$$
$$g_j(x_j) = y_j$$
$$g_j(x_{j+1}) = y_{j+1}$$

Consider $[x_j, x_{j+1}]$, $g_j(x)$ is a global interpolatant for $f(x)$ based on $(x_j, y_j), (x_{j+1}, y_{j+1})$. Based on the error of global interpolant, we know for $x \in [x_j, x_{j+1}]$, $f(x) - g_j(x) = \dfrac{(x - x_j)(x - x_{j+1})}{2} f''(c)$, c is some number from (x_j, x_{j+1})

$$\iff |f(x) - g_j(x)| \leqslant \max_{x_j \leqslant x \leqslant x_{j+1}} |P(x)||f''(s)|$$
$$s \in (x_j, x_{j+1})$$

Since $P(x) = \dfrac{(x - x_j)(x - x_{j+1})}{2}$, $x_{max} = \dfrac{x_j + x_{j+1}}{2}$, and $P(x)_{max} = \dfrac{(x_{j+1} - x_j)^2}{8}$. For $x \in [x_j, x_{j+1}]$,

$$|f(x) - g(x)| \leqslant \frac{h_j^2}{8} \max |f''(s)|$$

$s \in (x_j, x_{j+1})$

Theorem 4.4. Given f on (a, b). f,f',f" are continuous, consider $a = x_1 < x_2 < \cdots < x_{n+1} = b$, and the piecewise linear interpolation $g(x)$ of $f(x)$, $h = \max_{1 \leqslant j \leqslant n} |x_{j+1} - x_j|$
Then

$$|f(x) - g(x)| \leqslant \frac{1}{8} h^2 \max_{s \in [a,b)} |f''(s)|$$

Therefore, second order accuracy.
Discussion: For what $f(x)$, the error in g is zero?
Answer: When f(x) is linear polynomial of degree 1.

4.3.4 Least square solution, and the data fitting

Motivation:

1. $A \in \mathbb{R}^{m \times n}$, $m > n$, $b \in \mathbb{R}^m$ to solve $Ax = b$ for $x \in \mathbb{R}^n$
2. To overcome the possible issue of global interpolation (Runge phenomenon) $\{(x_j, y_j)\}_{j=1}^{n+1}$ approximate data by $P_m(x), m < n$.
3. Represent or analyze a more scattered data set.

4.4 Data Fitting

Review: Matrix- Vector Multiplication:
Given

$$A = (a_{ij}) \in \mathbb{R}^{m\times n}$$
$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$$

Define

$$Ax = b = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \in \mathbb{R}_m$$
$$\begin{aligned} b = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \\ = \begin{bmatrix} \sum_{j=1}^n a_{1j}x_j \\ \vdots \\ \sum_{j=1}^n a_{mj}x_j \end{bmatrix} \\ = \sum_{j=1}^n x_j \begin{bmatrix} a_{1j} \\ \vdots \\ a_{mj} \end{bmatrix} \end{aligned}$$

Let $a_j = \begin{bmatrix} a_{1j} \\ \vdots \\ a_{mj} \end{bmatrix}$ be the jth column of A. Then

$$\begin{aligned} b = Ax &= \sum_{j=1}^n x_j a_j \\ &= x_1a_1 + x_2a_2 + \cdots + x_na_n \end{aligned}$$

That is Ax is a linear combination of column vector of A, with the coefficients being the entries of x.
Consider

$$A \in \mathbb{R}^{m\times n}$$
$$b \in \mathbb{R}^m$$

(m ¿ n)
Projection of b into range(A), Find $\hat{x}$ such that $A\hat{x} = \hat{b}$
$\hat{x}$ will be the least squares solution of $Ax = b$:

- $b \in range(A)$
- $b - \hat{b} \perp Ax$, $\forall x \in \mathbb{R}^2$
- $b - \hat{b} \perp range(A)$

$$\iff ||b-\hat{b}||_2 = \min ||b-y||_2 \ \ \forall y \in range(A)$$

If $\hat{x}$ exists, how to find it?

$$\begin{aligned}
b-\hat{b} = b - A\hat{x} &\perp range(A) \\
\iff (b-A\hat{x}) &\perp A\hat{x} && \forall x \in \mathbb{R}^n \\
\iff (Ax)^T(b-A\hat{x}) &= 0 && \forall x \in \mathbb{R}^n \\
\iff x_T A^T(b-A\hat{x}) &= 0 && \forall x \in \mathbb{R}^n \\
\iff x^T w &= 0 && \forall x \in \mathbb{R}^n \\
\text{Take } x = w, w^T w &= 0 \\
w &= 0 \\
\iff A^T b &= A^T Ax \\
\text{Or } A^T Ax &= A^T b
\end{aligned}$$

Existence and uniqueness of $\hat{x}$
$\iff$ A^TA is invertible
$\iff$ Columns of A are linearly independent

Proof: Suppose A^TA is invertible, let $x \in \mathbb{R}^n$, satisfying $Ax = 0 \iff (x_1a_1 + x_2a_2 + \cdots x_na_n = 0)$
$\iff$ $A^A x = A^T 0 = 0$
A^TA being invertible.
$\hat{x} = 0$
$\iff$ columns of A are linearly independent.
Suppose columns of A are linear independent, we want to show A^TA is invertible.
By conreadiction, otherwise $\exists y \in \mathbb{R}^n, y \neq 0$

$$\begin{aligned}
A^TAy &= 0 \\
y^TA^TAy &= 0 \\
||Ay||_2^2 &= 0 \\
Ay &= 0
\end{aligned}$$

Columns of A are linearly independent: $y = 0$, contradiction
Therefore A^TA is invertible.
The existence and uniqueness of $\hat{x}$:

- $\iff$ A^TA is invertible
- Columns of A are linearly independent.

How to understand $\hat{x}$ is the best? in what sense?

Lemma 4.5. $\hat{b} \in range(A)$, satisfying $b - \hat{b} \perp range(A)$

$$\begin{aligned}
\implies ||b-\hat{b}||_2 &= \min_{y \in range(A)} ||b-y||_2 \\
\iff ||b-Ax||_2 &= \min_{x \in \mathbb{R}^n} ||b-Ax||_2
\end{aligned}$$

'Next best': The residual $r = b - Ax$ is minimized in $||\cdot||$ sense.

Proof of Lemma:
Let $S = range(A)$
Let $\hat{b} \in S$, and it satisfies $b - \hat{b} \perp S$, we want to show

$$||b - \hat{b}||_2 = \min_{\forall y \in S} ||b - y||_2$$

Consider any $y \in S$:

$$\begin{aligned}
||b - y||_2 &= ||b - \hat{b} + \hat{b} - y||_2^2 \\
&= (b - \hat{b} + \hat{b} - y)^T(b - \hat{b} + \hat{b} - y) \\
&= (b - \hat{b})^T(b - \hat{b}) + 2(b - \hat{b})^T(b - y) + (b - y)^T(b - y) \\
&= ||b - \hat{b}||_2^2 + 2(b - \hat{b})^T(b - y) + ||b - y||_2^2
\end{aligned}$$

Note $\hat{b} - y \in S$, hence $2(b - \hat{b})^T(b - y) = 0$, $\forall y \in S$

$$\begin{aligned}
\iff ||b - y||_2^2 &= ||b - \hat{b}||_2^2 + ||\hat{b} - y||_2^2 \\
&\geqslant ||b - \hat{b}||_2^2 \\
||b - y||_2 &\geqslant ||b - \hat{b}||_2 \\
||b - \hat{b}||_2 &= \min_{\forall y \in S} ||b - y||_2
\end{aligned}$$

Quadratic least square and linear least square, which is better?

- Intuitively: Quadratic is no worse than linear.
- To measure $r = b - Aa = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{bmatrix}$

  Squared Error(SE):
  $$SE = r_1^2 + r_2^2 + \cdots + r_m^2 = ||r||_2^2$$
  Root mean squared error(RMSE):
  $$RMSE = \sqrt{\frac{SE}{m}}$$

Discussion: $A \in \mathbb{R}^{m \cdot n}$, $b \in \mathbb{R}^m$
Consider $Ax = b$

1. Least square solution $(m > n)$

   $$||b - Ax||_2 = \min_{x \in \mathbb{R}^n} ||b - Ax||_2$$

   The normal equation:
   $$A^T Ax = A^T b$$
   Other norm can be used $||\cdot||_*$ instead of 2-norm.
2. When $m >> n$, computationally solving the normal equation is not the robust way to find LS solution.
   More robust algorithms are available:

- QR factorization
- SVD decomposition

3. What about $Ax = b$, $A \in \mathbb{R}^{m\times n}$, $m < n$

   Additional constraint are needed

   - $\|\|_*$ are needed
   - Fewest nonzero entries (Sparsity)

5 Numerical Differentiation and integration

$f(x)$: To approximate $f'(x)$, or to approximate $\int_a^b f(x)dx$

5.1 Numerical Differentiation

Given a function$f(x)$, we sample $\{x_j, y_j\}_{j=1}^{n+1}$, $y_j = f(x_j)$, we assume $x_{j+1} - x_j = h = constant$ (not essential).

n =1 (2 points): $P_1(x)$ is the linear interpolation:

$$P_1(x) = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$
$$P_1'(x) = \frac{y_2 - y_1}{x_2 - x_1} = \frac{f(x_2) - f(x_1)}{x_2 - x_1}$$

At x_1, $f'(x_1) \approx P'(x_1) = \dfrac{f(x_1 + h) - f(x_1)}{h}$

At x_2, $f'(x_2) \approx P'(x_2) = \dfrac{f(x_2 + h) - f(x_2)}{h}$

$$f'(x) \approx \begin{cases} \dfrac{f(x+h) - f(x)}{h} & \text{forward difference} \\ \dfrac{f(x) - f(x-h)}{h} & \text{backward difference} \end{cases}$$

These approximation can also be derive based on Taylor's expansion:

$$\begin{aligned} & f(x+h) - f(x) \\ =& hf'(x) + \frac{h}{2}f''(c) \end{aligned}$$

c is some number between x and $x + h$

$$\frac{f(x+h) - f(x)}{h} = f'(x) + \frac{h}{2}f''(c)$$

Forward different approximation:

$$f'(x) \sim \frac{f(x+h) - f(x)}{h}$$

The error $\dfrac{h}{2}f''(c)$ is first-order in h, or written as $\mathcal{O}(h)$

$\mathcal{O}(h^n)$: For a quantity $Q(h)$, if $\exists H > 0$, such that $|Q(h)| \leqslant Hh^n$, $\forall h > 0$, then $Q(h) = \mathcal{O}(h^n)$, it's said to be nth order of h.

Similarly,

$$\begin{aligned}
f(x-h) = f(x) &= -hf'(x) + \frac{h}{2}f''(\tilde{c}) && \tilde{c} \in [x-h, x] \\
\frac{f(x-h)-f(x)}{-h} &= \frac{f(x)-f(x-h)}{h} \\
&= f'(x) + \frac{h}{2}f''(\tilde{c})(\mathcal{O}(h)) \\
\implies f'(x) &\approx \frac{f(x)-f(x-h)}{h}
\end{aligned}$$

Backward difference approximation with a first order in h error.

n = 2

$P_2(x)$ is quadratic interpolation, $x_{j+1} - x_j = h$.

$$P_2(x) = y_1 + \frac{y_2 - y_1}{h}(x - x_1) + \frac{y_3 - 2y_2 + y_1}{2h^2}(x - x_1)(x - x_2)$$

Take derivative:

$$\begin{aligned}
P_2'(x) &= \frac{y_2 - y_1}{h} + \frac{y_3 - 2y_3 + y_1}{2h^2}(2x - x_1 - x_2) \\
P_2''(x) &= \frac{y_3 - 2y_2 + y_1}{h^2}
\end{aligned}$$

At $x = x_1$

$$\begin{aligned}
P_2'(x)|_{x=x_1} &= \frac{y_2 - y_1}{h} + \frac{y_3 - 2y_2 + y_1}{2h^2}(2x_1 - x_1 - x_2) \\
&= \frac{3y_1 + 4y_2 - y_3}{2h} \\
&= \frac{-3f(x_1) + 4f(x_1 + h) - f(x_1 + 2h)}{2h}
\end{aligned}$$

At $x = x_2$

$$\begin{aligned}
P_2'(x)|_{x=x_2} &= \frac{y_2 - y_1}{h} + \frac{y_3 - 2y_2 + y_1}{2h^2}(h) \\
&= \frac{y_3 - y_1}{2h} \\
&= \frac{f(x_2 + h) - f(x_2 - h)}{2h}
\end{aligned}$$

At $x = x_3$

$$P_2'(x)|_{x=x_3} = \frac{f(x_3 - 2h) - 4f(x_3 - h) + 3f(x_3)}{2h}$$

This gives us two one-sides approximation and one central approximation.

Similarly:

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

Central approximation

Example: Given $f(x)$ $(h > 0)$

- Approximate $f'(x)$ using $f(x), f(x+h), f(x+2h)$ (linear combination)
  - Up to 2nd order accurate (error $= \mathcal{O}(h^2)$)
  - What about approximation, of third order? what about $1st$ order?
- Approximate $f'(x)$ using $f(x), f(x+h), f(x+2h)$, up to $1st$ order, possibly $2nd$ order accuracy.

Using Taylor Series Expansion:

$$\begin{aligned} f(x+2h) &= f(x) + 2hf'(x) + \frac{(2h)^2}{2!}f''(x) + \frac{(2h)^3}{3!}f'''(x) + \mathcal{O}(h^4) \\ f(x+h) &= f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{3!}f'''(x) \\ f(x) &= f(x) \end{aligned}$$

Based on these:

$$\begin{aligned} &\alpha f(x+2h) + \beta f(x+h) + \gamma f(x) \\ =&(\alpha+\beta+\gamma)f(x) + (2\alpha+\beta)hf'(x) + (2\alpha+\beta)hf'(x) + \frac{4\alpha+\beta}{2} + \frac{4\alpha+\beta}{2}h^2f''(x) + \frac{6\alpha+\beta}{6}h^3f'''(x) + \mathcal{O}(h^4) \end{aligned}$$

To approximate $f(x)'$, we first require $\alpha+\beta+\gamma = 0$ (consistency). Also require $2\alpha+\beta \neq 0$.

$$\implies \frac{\alpha f(x+2h) + \beta f(x+h) + \gamma f(x)}{(2\alpha+\beta)h} = f'(x) + \frac{4\alpha+\beta}{2(2\alpha+\beta)}hf''(x) + \frac{8\alpha+\beta}{6(2\alpha+\beta)}h^2f'''(x) + \mathcal{O}(h^3)$$

To get 2^{nd} order approximation for $f'(x)$, we require $4\alpha+\beta = 0$
So far we have

$$\begin{aligned} \alpha+\beta+\gamma &= 0 \\ 4\alpha+\beta &= 0 \end{aligned}$$

So,

$$\begin{aligned} &\alpha \\ &\beta = -4\alpha \\ &\gamma = 3\alpha \end{aligned}$$

$$\begin{aligned} LHS &= \frac{\alpha f(x+2h) - 4\alpha f(x+h) + 3\alpha f(x)}{-2\alpha h} \\ &= \frac{f(x+2h) + 4f(x+h) - 3f(x)}{2h} \\ RHS &= f'(x) + \frac{8\alpha+\beta}{6(2\alpha+\beta)}h^2f'''(x) + \mathcal{O}(h^3) \\ &= f'(x) - \frac{1}{3}h^2f'''(x) + \mathcal{O}(h^3) \end{aligned}$$

Remark:

- It's impossible to get a 3rd order approximation for $f'(x)$, simply based on $f(x), f(x+h), f(x+2h)$

- What about first order?

$$\begin{aligned} \alpha + \beta + \gamma &= 0 \\ 4\alpha + \beta &\neq 0 \\ 2\alpha + \beta &\neq 0 \end{aligned}$$

  One example: we require $3\alpha + \beta = 0$

- If instead we want to approximate $f''(x)$ using $f(x), f(x+h), f(x+2h)$, we require

$$\begin{aligned} \alpha + \beta + \gamma &= 0 \\ 2\alpha + \beta &= 0 \\ 4\alpha + \beta &\neq 0 \end{aligned}$$

  and this leads to

$$\begin{aligned} \beta &= -2\alpha \\ \gamma &= -\alpha \end{aligned}$$

$$\begin{aligned} \implies \frac{\alpha f(x+2h) + \beta f(x+h) + \gamma f(x)}{\frac{(4\alpha+\beta)h^2}{2}} &= f''(x) + \frac{(8\alpha+\beta)/6}{\dfrac{4\alpha+\beta}{2}} h f'''(x) + \mathcal{O}(h^2) \\ \implies \frac{f(x+2h) - 2f(x+h) + f(x)}{h^2} &= f'(x) + hf''(x) + \mathcal{O}(h^2) \\ \implies f''(x) \sim \frac{f(x+2h) - 2f(x+h) + f(x)}{h^2} & \end{aligned}$$

  (First order, second order is impossible)

  But for central:

$$f''(x) \sim \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

  (2nd order approximation, central)

5.1.1 Richardson Extrapolation

(To enhance the accuracy)
Recall:

$$f'(x) = \frac{f(x+h) - f(x)}{h} - \frac{h}{2} f''(x) + \mathcal{O}(h^2)$$

General setting: To approximate a quantity Q, with $F_n(h)$, where $F_n(h)$ is nth order accurate.

$$Q = F_n(h) + kh^n + \mathcal{O}(h^{n+1})$$

Now, we half the stepsize:

$$\begin{aligned}
Q &= F_n(h/2) + k(h/2)^n + \mathcal{O}((h/2)^{n+1}) \\
2^nQ &= 2^nF_n\left(\frac{h}{2}\right) + kh^n + \frac{2^n}{2^{n+1}}\mathcal{O}(h^{n+1}) \\
(2^n-1)Q &= 2^nF_n\left(\frac{h}{2}\right) - F_n\left(\frac{h}{2}\right) + \mathcal{O}(h^{n+1}) \\
Q &= \frac{2^nF_n\left(\frac{h}{2}\right) - F_n(h)}{2^n-1} + \mathcal{O}(h^{n+1})
\end{aligned}$$

This gives a new approximation with $(n+1)$th order accurate for Q.
Example:

$$\begin{aligned}
Q &= f'(x) \\
F_1(h) &= \frac{f(x+h)-f(x)}{h} \\
\frac{2^nF_n(h/2)-F_n(h)}{2^n-1} &= \frac{2\left(\frac{f(x+h/2)-f(x)}{h/2}\right) - \frac{f(x+h)-f(x)}{h}}{1} \\
&= \frac{4(f(x+h/2)-3f(x)-f(x+h))}{h}
\end{aligned}$$

This gives a 2nd order approximation for $f'(x)$
Let $\hat{h} = \frac{h}{2}$

$$f'(x) \approx \frac{4f(x+\hat{h}) - 3f(x) - f(x+2\hat{h})}{2\hat{h}}$$

(Related to some approximation we've derived)

5.2 Numerical Integration

$$\int_a^b f(x)dx \approx \int_a^b P(x)dx$$

Newton- Cotes: When $P(x)$ is an interpolation based on equally spaced points
Linear Interpolation

$$\begin{aligned}
\int_{x_0}^{x_1} f(x)dxy_i & \qquad = f(x_i)i = 0, 1\cdots \\
P_1(x) &= y_0 + \frac{y_1-y_0}{x_1-x_0}(x-x_0) \\
f(x) &= P_1(x) + E(x) \\
E(x) &= \frac{(x-x_0)(x-x_1)}{2}f''(c) \qquad c(x)
\end{aligned}$$

$$\begin{aligned}\int_{x_0}^{x_1} f(x)dx &= \int_{x_0}^{x_1} P_1(x)dx + \int_{x_0}^{x_1} E(x)dx \\ &\approx \int_{x_0}^{x_1} P_1(x)dx \\ &= \int_{x_0}^{x_1} y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0)dx \\ &= \frac{y_0 + y_1}{2} \cdot h & h := x_1 - x_0\end{aligned}$$

Error:

$$\int x_0{}^{x_1} E(x)dx = -\frac{h^3}{12} f''(\tilde{c})$$

$\tilde{c} \in [x_0, x_1]$

5.2.1 Trapezoidal Rule

:

$$\begin{aligned}\int_{x_0}^{x_1} f(x)dx &\approx \frac{f(x_1) + f(x_0)}{2} \cdot h \\ Error &= -\frac{h^3}{12} f''(\tilde{c}) & \tilde{c} \in [x_0, x_1]\ h = x_1 - x_0\end{aligned}$$

Question : For what type f, the rule is exact?

$$f'' \equiv 0 \iff f(x) = a + bx \forall a, b \text{ constant}$$

Composite numerical quadrature:
Consider an equally spaced grid:

$$a = x_0 < x_1 \cdots < x_{n-1} < x_n = b \ \ x_{j+1} - x_j = h - \frac{b - a}{N}$$

On each subinterval (panel):

$$\int_{x_j}^{x_{j+1}} f(x) = \frac{h}{2}(f(x_{j+1}) + f(x_j)) - \frac{h^3}{12} f''(c_j) \ \ c_j \in [x_j, x_{j+1}]$$

Sum up in j:

$$\begin{aligned}&\int_a^b f(x)dx = \sum_{j=1}^{N-1} \frac{h}{2}(f(x_{j+1}) + f(x_j)) - \frac{h^3}{12} \sum_{j=0}^{N-1} f''(c_j) \\ &\sum_{j=0}^{N-1} f''(c_j) = N f''(c) & Nh = b - a \\ &\int_a^b f(x)dx = \frac{h}{2}(f(a) + 2f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-1}) + f(b)) + \frac{h^2}{12}(b - a) f''(c)\end{aligned}$$

Suppose $\min_{x \in [a,b]} g(x) < \alpha < \max_{x \in [a,b]} g(x)$, g is continuous, $\exists c$, s.t. $g(c) = \alpha$
Therefore,

$$N \min_{x \in [a,b]} f''(x) < \sum_{j=0}^{N-1} f''(c_j) < N \max_{x \in [a,b]} f''(x)$$

and

$$\min_{x\in[a,b]} f''(x) < \frac{\sum_{j=0}^{N-1} f''(c_j)}{N} = f''(c) < \max_{x\in[a,b]} f''(x)$$

5.2.2 Simpson's Rule

:

$x_2 - x_1 = x_1 - x_0 = h,\ f_1(x) \approx P_2(x)$

$$\begin{aligned}\int_{x_0}^{x_2} f(x)dx &\approx \int_{x_0}^{x_2} P_2(x)dx\\ &= \frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2))\\ Error &= -\frac{h^5}{90} f^{(4)}(c) && \text{for some } c \in [x_0, x_2]\end{aligned}$$

Composite Simpson's rule:

$$a = x_0 < x_1 < x_2 \cdots < x_{2N} = b$$

On each panel, $[x_{2j}, x_{2j+2}]$

$$\begin{aligned}\int_{x_{2j}}^{x_{2j+2}} f(x)dx &= \frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2)) - \frac{h^5}{90} f^{(4)}(c_j)\\ \int_a^b f(x)dx &= \frac{h}{3}(f(a) + f(b) + 4\sum_{j=1}^{N} f(x_{2j+1}) + 2\sum_{j=1}^{N-1} f(x_{2j})) - \frac{b-a}{180} h^4 f^{(4)}(\tilde{c}) \qquad \tilde{c} \in [a, b]\end{aligned}$$

Question: When will the rule exact?
When $f^{(4)} \equiv 0 \iff f(x) = a + bx + cx^2 + dx^3$

Definition 5.1 (Degree of Precision (D.O.P)). Degree of Precision (D.O.P) of a numerical interpolation formula is the largest integer k, such that the formula is exact when the integrand is any polynomial of degree up to k, so far: Trap (DOP =1), Simpson(DOP=3)

Next Question: How to find DOP?
Change of variable:
Reference element: [0,1], [-1,1]:
Suppose we have

$$\int_0^1 f(x)dx \approx \sum_{j=1}^{n} \omega_j f(x_j)$$

on a physical interval $[a, b]$

$$\begin{aligned}
\int_a^b f(y)dy \xrightarrow{\frac{y-a}{b-a}=x} &= \int_0^1 f(a+(b-a)x)dx(b-a) \\
&\approx \sum_{j=1}^{n} \omega_j F(x_j)(b-a) \\
&= \sum_{j=1}^{n} \{(b-a)\omega_j\} f(a+(b-a)x_j) \\
&= \sum_{j=1}^{n} \hat{\omega_j} f(\hat{x_j}) \\
&\begin{cases} \hat{\omega_j} &= (b-a)\omega_j \\ \hat{x_j} &= a+(b-a)x_j \end{cases}
\end{aligned}$$

DOP reserved under the linear change of variable.
Exercise: Trap

$$\int_0^1 f(x)dx \approx \frac{1}{2}(f(0)+f(1))$$

$$f(x) = 1 \qquad \begin{cases} LHS &= \int_0^1 1dx = 1 \\ RHS &= \frac{1}{2}(1+1) = 1 \end{cases}$$

$$f(x) = x \qquad \begin{cases} LHS &= \int_0^1 xdx = \frac{1}{2} \\ RHS &= \frac{1}{2}(0+1) = \frac{1}{2} \end{cases}$$

$$f(x) = x^2 \qquad \begin{cases} LHS &= \int_0^1 x^2dx = \frac{x^3}{3} \\ RHS &= \frac{1}{2}(f(0)+f(1)) = \frac{1}{2} \end{cases}$$

Therefore, DOP of trap is 1.
Example:
Find D.O.P of the following numerical interpolations:

$$\int_{x_0}^{x_1} f(x)dx \approx \begin{cases} f(x_0)\cdot h & \text{LEFT - RECTANGLE} \\ f(\frac{x_0+x_1}{2})\cdot h & \text{MIDPOINT} \\ f(x_1)\cdot h & \text{RIGHT - RECTANGLE} \end{cases}$$

On a reference element $(0,1)$ formula become

$$\int_0^1 f(x)dx \approx \begin{cases} f(0) \\ f(1/2) \\ f(1) \end{cases}$$

$$\begin{aligned}
LHS &= \int_0^1 x^k dx = \frac{x^{k+1}}{k+1}\Big|_{x=0}^{x=1} \\
&= \frac{1}{k+1} \qquad\qquad k = 0, 1, \cdots
\end{aligned}$$

Rectangle Rules : Have DOP =0
Mid-Point Rules : Have DOP =1
Remark: Composite mid-point method:

$$\begin{aligned}
a = x_0 < x_1 \cdots < x_{N-1} < x_N &= b \\
x + j - x_{j-1} &= h \\
\int_a^b f(x)dx &\approx h \sum_{j=0}^{N-1} f(\omega_j) \\
\omega_j &= \frac{x_j + x_{j+1}}{2} \\
Error &= \frac{b-a}{24} h^2 f''(c) && c \in [a,b] \\
h &= \frac{b-a}{N}
\end{aligned}$$

5.3 Romberg Interpolation

(Richardson extrapolation)
Consider composite Trap. rule:

$$a = x_0 < x_1 < x_2 \cdots < x_{N-1<x_N=b}$$

$$\int_a^b f(x)dx = \frac{h}{2}(f(a) + f(b) + 2\sum_{j=1}^{N-1} f(x_j)) + c_2h^2 + c_4h^3 + c_6h^6 + \cdots$$

c_2, c_4 can be derived analytically,

$$c_2 = \frac{f'(a) - f'(b)}{2}$$

$N = \dfrac{b-a}{h}$
Consider a sequence of meshes:

$$\begin{aligned}
h_1 &= \frac{b-a}{1} \\
h_2 &= \frac{b-a}{2} \\
h_3 &= \frac{b-a}{4} \\
&\vdots \\
h_j &= \frac{b-a}{2^{j-1}}
\end{aligned}$$

Apply composite Trap. method on each mesh and get a numerical integral, R_{j1}
Recall Richardson Extrapolation:

$$\begin{aligned}
Q &= F_n(h) + kh^n + \mathcal{O}(h^{n+r}) \\
Q &= F_n(h/2) + k(h/2)^n + \mathcal{O}((h/2)^{n+r}) \\
\Longrightarrow Q &= \frac{2^nF_n(h/2) - F_n(h)}{2^n - 1} + \mathcal{O}(h^{n+r})
\end{aligned}$$

Based on this, based on R_{j1}, which is 2^{nd} order accurate, which is $n=2$

$$\begin{aligned}&\frac{2^n R_{j,1}-R_{j-1,1}}{2^n-1}\\=&\frac{4R_{j,1}-R_{j-1,1}}{3}\\=:&R_{j,2}\end{aligned}$$

This provided a $4th$ order approximation for $\int_a^b f(x)dx$
Once we have a 4^{th} order approximation $R_{j,2}$, apply extrapolation (n=4)

$$\begin{aligned}\frac{2^n R_{j,2}-R_{j-1,2}}{2^n-1}&=\frac{16R_{j,2}-R_{j-1,2}}{15}\\&=: R_{j,3}\end{aligned}$$

This gives a sixth order approximation

R_{11}				
R_{21}	R_{22}			
R_{31}	R_{32}	R_{33}		
R_{41}	R_{42}	R_{43}	R_{44}	
R_{51}	R_{52}	R_{53}	R_{54}	R_{55}
2^{nd}	4^{th}	6^{th}	8^{th}	10^{th}

$$R_{jk}=\frac{4^{k-1}R_{j,k-1}-R_{j-1,k-1}}{4^{k-1}-1}$$

j: Divided into 2^{j-1} meshes
k: Interpolate $k-1$ times, error is $2k$
To compute R_{j1} recursively:

$$\begin{aligned}R_{11}&=\frac{h_1}{2}(f(a)+f(b))\\R_{21}&=\frac{h_2}{2}(f(a)+f(b)+2f(a+h_2))\\&=\frac{1}{2}R_{11}+h_2f(a+h_2)\\R_{31}&=\frac{h_3}{2}(f(a)+f(b)+2f(a+h_3)+2f(a+2h_3)+2f(a+3h_3))\\&=\frac{1}{2}R_{21}+h_3(f(a+h_3)+f(a+3h_3))\end{aligned}$$

In general, $R_{j1}=\frac{1}{2}R_{j-1,1}+h_j\sum_{i=1}^{2j-2}f(a+(2i-1)h_j)$
To control the error, one can run till the level m, when $|R_{mm}-R_{m-1,m-1}|\leqslant myTol$

5.4 Adaptive Quadrature

To compute $\int_a^b f(x)dx$ numerically, recall on a single panel $\int_{x_j}^{x_{j+1}} f(x)dx\approx$ computed value $+ch^m f^{(n)}(c)$

$$x_{j+1}-x_j=h$$

To achieve a given level of error, the larger $f^{(n)}$ is, the small h should be. In practice, $f^{(n)}$ is often unknown.

Goal: To compute $\int_a^b f(x)dx$ with a given level of error efficiency. Using "Adaptive Quadrature".
To see the **main ingredient**, given $f(x)$ on $[a,b]$, let $S[x_L,x_R]$ be a numerical strategy to compute

$$\int_{x_L}^{x_R} f(x)dx$$

(Error unknown)
Goal: Compute $\int_a^b f(x)dx$ with a given error tolerance.
First,

$$\begin{aligned} s[a,b] \\ s[a,c]+s[c,b] \end{aligned}$$

The error is unknown. Based on these approximation, design an error indicator.
Second, Based on error indicator, decide whether you want to accept:

$$s[a,c]+s[c,b]\begin{cases}\text{YES} : INT = s[a,c]+s[c,b] \text{ STOP}\\ \text{NO} : \text{Repeatly treat}[a,c],[c,b]\text{as a starting error}\end{cases}$$

- Design an error indicator
- Accept the result? Or not?
- How to track or organize multiple subinterval?

First, we want to design an error indicator, use Trap. rule as an example:

$$\begin{aligned}\int_a^b f(x)dx &= s[a,b]-\frac{h^3}{12}f''(c_0)\\ &= \frac{(f(a)+f(b))(b-a)}{2}\end{aligned}$$

Error unknown,

$$\begin{aligned}\int_a^b f(x)dx &= \int_a^c f(x)dx+\int_c^b f(x)dx\\ &= s[a,c]-\left(\frac{h}{2}\right)^3\frac{f''(c_1)}{12}+s[c,b]-\left(\frac{h}{2}\right)^3\frac{f''(c_2)}{12}\\ &= s[a,c]+s[c,b]-\frac{h^3}{4}\frac{f''(c_3)}{12}\end{aligned}$$

Consider

$$\begin{aligned}s[a,b]-(s[a,c]+s[c,b]) &= \frac{h^3 f''(c_0)}{12}-\frac{h^3}{4}\frac{f''(c_3)}{12}\\ &\approx 3\cdot\left(\frac{h^3}{4}\frac{f''(c_3)}{12}\right)\\ &\quad(\text{Error in} s[a,c]+s[c,b])\end{aligned}$$

Error indicator (must be computable):

$$errI = |s[a,b]-(s[a,c]+s[c,b])|$$

Given f on $[a,b]$
Given an error tolerance, $myTol$
We want to get an approximation I_{num} for $\int_a^b f(x)dx$
such that $|I_{num}-\int_a^b f(x)dx| < myToL$

- Adaptively
- Error indicator (Computable)
- Using Trap. rule as an example

Let $s[x_L, x_R] = \dfrac{f(x_L) + f(x_R)}{2}(x_R - x_L)$
Start with $b - a = h$, get an approximation $I_1 = S[a, b]$ for $\int_a^b f(x)dx$
Error $I_1 = \int_a^b f(x)dx + \dfrac{h^3 f(c_0)}{12}$, $c_0 \in [a, b]$.

Next, we compute a second approximation, c midpoint,

$$
\begin{aligned}
I_2 &= s[a, c] + s[c, b] \\
\text{Error } I_2 &= \int_a^b f(x)dx = \frac{h^3}{4}\frac{f''(c_3)}{12} && c_3 \in [a, b] \\
|I_1 - I_2| &= \left|\frac{h^3}{12}f''(c_0) - \frac{h^3}{4}\frac{f''(c_3)}{12}\right| \\
&\approx 3\left|\frac{h^3 f''(c_3)}{4 \cdot 12}\right|
\end{aligned}
$$

Error in I_2 to approximate $\int_a^b f(x)dx$
Define error indicator:

$$
\begin{aligned}
errI &= |I_1 - I_2| \\
\text{if } errI &\leqslant 3myToL && \iff (|I_2 - \int_a^b f(x)dx| \leqslant myTol)
\end{aligned}
$$

Accept I_2, then set $I_{num} = I_2$
Otherwise, split $[a, b]$ to $[a, c].[c, b]$
Repeat the progress as for $[a, b]$

$$
\begin{aligned}
|I_{num} - \int_a^b f(x)dx| &\leqslant myToL \\
|I_{num} - \int_a^c f(x)dx| &\leqslant \frac{myToL}{2}
\end{aligned}
$$

How to track and manage multiple subintervals?

- Create a list which records all the subintevcals to be processed.
- Follow "last come first serve" strategy

Initial list

$$a(1) = a \;\; b(1) = b \;\; ToL(1) = myToL$$

To begin with, $n = 1$ is the queue length, $I_{num} = 0$, c midpoint.
If error indicator $\leqslant 3tol(1)$
Accept and set $I_{num} = s[a(1), c] + s[c, b(1)]$
$n < -n - 1$

STOP
Otherwise:
Set $[a(i), b(i)]$, $i = 1, 2$
$n < -n + 1$, (n=2 now)
$ToL(1) < -\frac{Tol(1)}{2}$, $ToL(2) = ToL(1)$
We follow last come first serve and consider
If error indicator ¡ 3 $ToL(2)$
Then $I_{num} = I_{num} + s[a(2), c] + s[c, b(2)]$
$n < -n - 1$
That us, to delete $[a(2), b(2)]$ from the queue.
Otherwise, split $[a(2), b(2)]$ into subintervals $[a(2), b(2)], [a(3), b(3)]$
$n < -n + 1$ $(n = 3)$
$ToL(2) < -\frac{Tol(2)}{2}$, $ToL(3) = ToL(2)$
Exercise
If simpson's rule is used to replace Trap. rule, which part need to be changed?

- $s[a, b]$
- error indicator $< 15ToL$

5.5 Gaussian Quadrature

Newton- Cote's formula: $P(x)$ is based on equally spacing points
(n+1)points: DOP = n, when n is odd, (Trap, n=1)
DOP = n+1, when n is even, (Simpson, n=2)
Question: If the number of points is $n + 1$, what is the highest d.o.p a quadrature (based on these points) can be achieved?
Answer: The highest d.o.p = 2n+1 (Associate with $n + 1$ points)
This is achieved by Guassian Quadrature.
Start with the simplest case:
$n = 0$: 1-point, mid-point rule, d.o.p =1
$n = 1$: 2-point, $\int_{-1}^{1} f(x)dx \approx \sum_{i=1}^{2} \omega_i f(x_i)$ Find the highest k such that the formula is exact for $f(x) = x^k$.

$$LHS = \int_{-1}^{1} x^k dx = \begin{cases} 0 & k \text{ is odd} \\ 2\int_0^1 x^k dx = \dfrac{2}{k+1} & k \text{ is even} \end{cases}$$

k	LHS	RHS	
0	2	$\omega_1 + \omega_2$	(1)
1	0	$\omega_1 x_1 + \omega_2 x_2$	(2)
2	$\frac{2}{3}$	$\omega_1 x_1^2 + \omega_2 x_2^2$	(3)
3	0	$\omega_1 x_1^3 + \omega_2 x_2^3$	(4)
4	$\frac{2}{5}$	$\omega_1 x_1^4 + \omega_2 x_2^4$	(5)

Nonlinear: We hope to find ω_1, ω_2, x_1, $x_2$such that $(1) - (4)$ hold. Then

$$\omega_1 x_1 \neq 0$$
$$\omega_2 x_2 \neq 0$$

(2)(4)

$$\begin{aligned} x_1^2 &= x_2^2 \\ x_1 &\neq x_2 \\ \implies x_1 &= -x_2 = \alpha < 0 \end{aligned}$$

(1): $\omega_1 + \omega_2 = 2$
(2): $\omega_1 - \omega_2 = 0$
$\omega_1 = \omega_2 = 1$
(3) $(\omega_1 + \omega_2)\alpha^2 = \dfrac{2}{3}$
$\alpha = -\dfrac{1}{\sqrt{3}}$
(4) $(\omega_1 - \omega_2)\alpha^3 = 0 \implies$ 2 point Gaussian $x_1 = -\sqrt{\dfrac{1}{3}}$, $x_2 = \sqrt{\dfrac{1}{3}}$
$\omega_1 = \omega_2 = 1$
$\int_{-1}^{1} f(x)dx \approx \int_{i=1}^{2} \omega_i f(x_i)$
DOP =3 =2n+1, n=1
In general, Gaussian quadrature are related to legendre polynomials
Legendre Polynomials, defined on $[-1, 1]$, defined in different ways

1. Recursively defined

$$\begin{aligned} P_0(x) &= 1 \\ P_1(x) &= x \\ (n+1)P_{n+1}(x) &= (2n+1)xP_n(x) - nP_{n-1}x && n \geqslant 1 \end{aligned}$$

   For example, $n = 1$,

$$\begin{aligned} 2P_2(x) &= 3xP_1(x) - P_0(x) \\ \implies P_2(x) &= \frac{1}{2}(3x^2 - 1) \end{aligned}$$

   when $n = 2$, $P_3(x) = \dfrac{1}{2}(5x^3 - 3x)$

2. $P_n(x) = \dfrac{1}{n \cdot 2^n}$: $\dfrac{d^n}{dx^n}((x^2-1)^n)$, $n = 0, 1, \cdots$
3. Solutions of special differential equation, (Legendre DE)

Some properties:

1. Orthogonal $\int_{-1}^{1} P_m(x)P_n(x)dx = 0$, if $m \neq n$
2. $P_n(x)$ has n distinct roots over $[-1, 1]$
3. $\{P_0(x), P_1(x) \cdots P_n(x)\}$ form a basis for polynomial up to degree n.
4. $P_n(1) = 1$, $P_n(-1) = (-1)^n$

Gaussian Quadrature with $(n+1)$ points:

$$\int_{-1}^{1} f(x)dx \approx \sum_{j=1}^{n+1} \omega_j f(x_j)$$

$\{x_j\}_{j=1}^{n+1}$ re the roots of Legendre polynomial $P_{n+1}(x)$ of degree $n+1$.
What are the roots of Legendre polynomial $P_{n+1}(x)$ of degree $n+1$?
What are the weights $\{\omega_j\}_{j=1}^{n+1}$?

$$\int_{-1}^{1} f(x)dx \approx \int_{-1}^{1} Q(x)dx \xrightarrow{\text{lead to}} \sum_{j=1}^{n+1} \omega_j f(x_j)$$

Given $\{x_j\}_{j=1}^{n+1}$,$Q(x) = \sum_{j=1}^{n+1} L_j(x)f(x_j)$, and

$$\begin{aligned} L_j(x) &= \frac{\Pi_{i\neq j}(x-x_i)}{\Pi_{i\neq j}(x_j-x_i)} \\ \int_{-1}^{1} Q(x)dx &= \int_{-1}^{1} \sum_{j=1}^{n+1} L_j(x)f(x_j)dx \\ &= \sum_{j=1}^{n+1} \int_{-1}^{1} L_j(x)dx f(x_j) \end{aligned}$$

Property: $\sum_{j=1}^{n+1} \omega_j = 2$
Proof: Due to DOP $\geqslant 0$, and quadrature formula is exact when $f(x) = 1$, ($\int_{-1}^{1} 1dx = \sum_{j=1}^{n+1} \omega_j \cdot 1$)

6 Numerical Method for Solving initial value problems (IVP)

Given a function $f(t, y)$
Considering the following differential equation:

$$\begin{aligned} y'(t) &= f(t, y(t)) \\ y(0) &= \alpha \end{aligned}$$

$t > 0$, α is given.
This is an example of IVP.
Goal: Solve IVPs numerically, (find approximations for $y(t)$.)
Examples:

1. Radioactive decay:

$$\begin{aligned} y'(t) &= -ry(t) & t > 0, (r > 0 \text{ is constant}) \\ y(0) &= \alpha \end{aligned}$$

RHS $= -ry$, linear in y, first order linear, $y(t) = \alpha e^{-rt}$

2. Population models

$$\begin{aligned} y' &= -\lambda y & t > 0 \\ y(0) &= \alpha \end{aligned}$$

$y(t) = \alpha e^{\lambda t}$. First order linear, $\lambda > 0$ is given
Not a good ,pdel for large y.

Logistic equation (with a capacity)

$$\begin{aligned} y' &= \lambda(1-y)\cdot y \qquad\qquad t > 0 \\ y(0) &= \alpha \end{aligned}$$

First order nonlinear.

Exact solution:

$$y(t) = \frac{\alpha}{\alpha + (1-\alpha)e^{-\alpha t}} \qquad\qquad t \geqslant 0$$

Two steady states $y = 1$, and $y = 0$.

3. Predator-Prey model:

$$\begin{aligned} \frac{dy_1}{dt} &= -by_1 + \gamma y_1 y_2 \\ \frac{dy_2}{dt} &= ay_1 - \beta y_1 y_2 \qquad\qquad a, b, \gamma, \beta > 0 \end{aligned}$$

First order system, nonlinear.
Vector form:

$$\begin{aligned} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} &= \bar{y} \\ \frac{dy}{dt} &= f(t, y) \qquad\qquad t > 0 \\ y &= \alpha \end{aligned}$$

4. Newton's second law of motion
$F = ma$. $y(t)$: Position of an object in motion.

$$\begin{aligned} my'' &= F(t, y, y') \qquad\qquad t > 0 \\ y(0) &= \alpha \\ y'(0) &= \beta \end{aligned}$$

(Second Order)
This equation can be rewritten as a first order system, let $y'(t) = \omega(t)$

$$\begin{aligned} y'(t) &= \omega(t) \qquad\qquad t > 0 \\ \omega'(t) &= \frac{1}{m}F(t, y, \omega) \qquad\qquad t > 0 \\ y(0) &= \alpha \\ \omega(0) &= \beta \end{aligned}$$

First order system.

Here, we only focus on

$$\begin{aligned} y' &= f(t, y) t > 0 \\ y(0) &= \alpha \end{aligned}$$

y: Scalar or system
Assumption: Existence and uniqueness of the solution.

6.1 Forward Euler Method

Consider

$$\begin{aligned} y' &= f(t,y)t > 0 \\ y() &= \alpha \end{aligned}$$

One can think of this differential equation as a field of slopes.
Starting from α, direction or slope field.
Discrete Version:
$t_{j+1} - t_j = h$, time step, h is constant.

$$\begin{aligned} y_1 &= y_0 + hf(t_0, y_0) \\ y_2 &= y_1 + hf(t_1, y_1) \\ &\vdots \\ y_{j+1} &= y_j + hf(t_j, y_j) \end{aligned}$$

We hope $y_j = y(t_j)$, where $y(t)$ is the exact solution. $|y_j - y(t_j)|$ is not too big.
Revisit Forward Euler method:

$$\begin{aligned} \frac{y_{j+1} - y_j}{h} &= f(t_j, y_j) && j \geqslant 0 \\ y_0 &= \alpha \end{aligned}$$

Recall IVP:

$$\begin{aligned} y'(t) &= f(t, y(t)) && t > 0 \\ y(0) &= \alpha \end{aligned}$$

This seems to be related to

$$y'(t_j) \approx \frac{y(t_{j+1} - y(t_j))}{h}$$

Forward difference for $y'(t)$
Rederive the forward euler method:
4 steps:

$$\begin{aligned} y'(t) &= f(t, y(t)) && t \in [0, T] \\ y(0) &= \alpha \end{aligned}$$

1. Define a mesh/ partition of $[0, T]$
   For simplicity,

$$\begin{aligned} t_1 &= t_0 + h \\ t_2 &= t_1 + h \\ &\cdots \\ t_M &= T = t_{M-1} + h && h = \frac{T}{M} \end{aligned}$$

2. Read the equation at t_j, $j = 1, 2, \cdots M$

$$y'(t_j) = f(t_j, y(t_j))$$

3. Replace $y'(t_j)$ by its numerical differentiation

$$y'(t_j) = \frac{y(t_{j+1}) - y(t_j)}{h} - \frac{h}{2}y''(c_j)$$

   $c_j \in [t_j, t_{j+1}]$
   We have

$$\frac{y(t_{j+1}) - y(t_j)}{h} - \frac{h}{2}y''(c_j) = f(t_j, y(t_j))$$

4. Drop $O(h)$, change $y(t_j)$ to y_j.

$$\frac{y_{j+1} - y_j}{h} = f(t_j, y_j) \qquad j \geqslant 0$$
$$y_0 = \alpha$$

   This is the forward euler method.
   Numerical solution: $y_1, y_2 \cdots y_M$
   $y_j \sim y(t_j)$

Discussion:

1. $y(t_j)$: Exact solution.
   y_j: Approximation for $y(t_j)$
   We hope $y_j \approx y(t_j)$
2. F.E is a one-step method. To get y_{j+1}, one only needs y_j
3. It's explicit: To solve y_{j+1}, one does not need to solve an algebraic equation, all you need is algebraic evaluation.
4. The term we drop is $\mathcal{O}(n)$, this seems to imply sd h decrease, the computed solution will improve.
   $|y(t_j) - y_j|$ decreases as h decreases.
   For F.E, this is the case. (This is not always the case for other scheme)
   Stability

Backward Euler Method: In step3,

$$y'(t_j) = \frac{y(t_j) - y(t_{j-1})}{h} + \frac{h}{2}y''(\tilde{c_j})$$

$\tilde{c_j} \in (t_{j-1}, t_j)$

$$\implies \frac{y(t_j) - y(t_{j-1})}{h} + \frac{h}{2}y''(\tilde{c_j}) = f(t_j, y(t_j))$$

Step4: Drop $O(h)$, replace $y(t_j)$ by y_j, we get the scheme,

$$\frac{y_j - y_{j-1}}{h} = f(t_j, y_j) \qquad j \geqslant 1$$
$$y_0 = \alpha$$

Discuss:

1. One step

2. Implicit, an algebraic equation need to be solved to get y_j

   As an example:

$$y' = f(t, y) := y^3 + \sin(t)$$

   Apply B.E to it:

$$\begin{aligned} y_j &= y_{j-1} + h(y_j^3 + \sin(t_j)) \\ y_j - hy_j^3 &= y_{j-1} + h\sin(tj) \end{aligned}$$

   Over one time step, BE is in general more expensive than F.E

3. Similar as F.E., the term dropped is $\mathcal{O}(h)$
4. Geometric Interpolation

6.2 Basic Concepts

- LOCAL : local truncation error
- LOCAL : Consistency
- GLOBAL : Error
- GLOBAL : Stability

Using F.E method as an example:

$$y_{j+1} = y_j + hf(t_j, y_j)$$

Rewrite the scheme compatible to the IVP:

$$\begin{aligned} y' &= f(t, y) \\ \frac{y_{j+1} - y_j}{h} &= f(t_j, y_j) \end{aligned}$$

Definition 6.1 (Local Truncation Error)**.** Let $y(t)$ be an exact solution of $y' = f(t, y)$

$$\tau_j = \frac{y(t_{j+1}) - y(t_j)}{h} - f(t_j, y(t_j))$$

is the local truncation error, $\tau_j = \mathcal{O}(h)$.

The scheme is said to be consistent if $\tau_j \to 0$ as $h \to 0$.
For FE, $\tau_j = \dfrac{h}{2}y''(c_j)$, $c_j \in (t_j, t_{j+1})$
To derive the above formula:

$$\begin{aligned} \tau_j &= \frac{y(t_{j+1}) - y(t_j)}{h} - f(t_j, y(t_j)) \\ &= \frac{y(t_{j+1}) - y(t_j)}{h} - y'(t_j) \\ &= \frac{y(t_j) + hy'(t_j) + \frac{h^2}{2}y''(c_j) - y(t_j)}{h} - y'(y_j) \\ &= \frac{h}{2}y''(c_j) \end{aligned}$$

Remarks

1.

$$
\begin{aligned}
y_j &\approx y(t_j) \\
y_j &\neq y(t_j) && \text{In general} \\
y_j &= y_{j-1} + hf(t_{j-1}, y_{j-1}) \\
y(t_j) &= y(t_{j-1}) + hf(t_{j-1}, y(t_{j-1})) + h\tau_{j-1}
\end{aligned}
$$

2. LTE measure the error locally, over one step. It can be defined for other scheme, for instance, for B.E:

$$
\begin{aligned}
\tau_j &= \frac{y(t_j) - y(t_{j-2})}{h} - f(t_j, y(t_j)) \\
&= -\frac{h}{2} y''(c_j) \\
&= \mathcal{O}(h) \\
c_j &\in (t_{j-1}, t_j)
\end{aligned}
$$

$y(t_j)$: Exact solution at t_j y_j : Computed solution at t_j, $j = 1, 2, \cdots n$

Question: $y(t_j) - y_j$?
Two ways to measure:

$$
\begin{aligned}
e_M &= |y(t_M) - y_M| \\
&= |y(T) - y_M|
\end{aligned}
$$

$$
e_{M,n} = \max_{1 \leqslant j \leqslant M} |y(t_j) - y_j|
$$

When $M \to \infty$, or M increase, same as h decreases, whether $e_M, e_{M,n}$ decrease?
With same assumption of f, one can show:

$$
\begin{aligned}
|e_M| &\leqslant c \max_{1 \leqslant j \leqslant M} |\tau_j| \\
|e_{M,n}| &\leqslant \tilde{c} \max_{1 \leqslant j \leqslant M} |\tau_j|
\end{aligned}
$$

For FE or BE, $e_M, e_{M,n} = \mathcal{O}(h)$
What we are seen here are:
B.E. F.E are consistent, with L.T.E Local error : $\mathcal{O}(h)$
Global error : $\mathcal{O}(h)$
Convergent scheme of order 1
Consistency + Stability $\implies$ Convergence

6.3 Stability

:
Many notions of stability:
$0-$ stability: when h decrease to 0.
Absolute stability: Behavior of a scheme when h is not so small.
Here, we examine absolute stability.

Consider the test equationL

$$\begin{aligned} y' &= -ry && t > 0 \\ y(0) &= \alpha && r > 0 \text{ is some constant} \end{aligned}$$

Exact solution: $y(t) = \alpha e^{-rt}$
Absolute Stability of a numerical method:
Apply the method to the test equation of $y' = ry$, $t > 0$, $y(0) = \alpha$, we require $|y_j| = c < \infty$, for any j for some c, that is $\{y_j\}_{j=1}^{\infty}$ is bounded.
We'll see this is the same requiring $|y_{j+1}| \leqslant |y_j|, \forall j$.
Take F.E. as an example:

$$y_{j+1} = y_j + hf(t_j, y_J)$$

Apply it to a test equation $f(t, y) = -ry$, $(r > 0)$, then the scheme is

$$\begin{aligned} y_{j+1} &= y_j + h(-ry_j) \\ &= (1 - hr)y_j \end{aligned}$$

Q" growth factor, amplication factor
Recursively:

$$\begin{aligned} y_j &= Q(hr)y_{j-1} \\ &= Q(hr)Q(hr)y_{j-2} \\ &\cdots \\ &= (Q(hr))^j y_0 = \alpha \end{aligned}$$

$\{y_j\}_j$ being bounded $\iff |Q(hr)| \leqslant 1$
$\iff |y_{j+1}| \leqslant |y_j|$
For F.E.

$$\begin{aligned} Q(hr) &= 1 - hr \\ \iff |1 - hr| &\leqslant 1 \\ \iff -1 \leqslant 1 - hr &\leqslant 1 && r > 0,\ h > 0 \\ \iff h &\leqslant \frac{2}{r} \end{aligned}$$

That is , when F.E. is applied to the test equation, the solution $\{y_j\}_{j=0}^{\infty}$ is bounded if and only if $h \leqslant \dfrac{2}{r}$
This makes F.E. to be conditionally stable.
Now, apply B.E. method to

$$\begin{aligned} y' &= -ry && t > 0 \\ y(0) &= \alpha \end{aligned}$$

$$\begin{aligned} y_{j+1} &= y_j + hf(t_{j+1}, y_{j+1}) \\ &= y_j - hry_{j+1} \\ \implies y_{j+1} &= \frac{1}{1 + hr} y_j \end{aligned}$$

Absolute stability $\iff |Q(hr)| \leqslant 1$, $\dfrac{1}{1 + hr} \leqslant 1$
This holds for all $h > 0$
This makes B.E.method "unconditionally" stable.

6.4 More example of numerical methods

$$\begin{aligned} y' &= -ry & t > 0 \;\; t \in [0, T] \\ y(0) &= \alpha \end{aligned}$$

Following the similar derivations:

- Step1: Mesh

$$0 = t_0 < t_1 < \cdots < t_{M-1} < t_M = T$$
$$h = \frac{T}{M}$$

- Step2: Read equation at t_j

$$y'(t_j) = f(t_j, y(t_j))$$

  $j = 1, \cdots M$

- Step3: Replace $y'(t_j)$

$$y'(t_j) = \begin{cases} \dfrac{y(t_{j+1} - y(t_{j-1}))}{2h} - \dfrac{h^2}{2} g'''(c_j) \\ \dfrac{3y(t_j) - 4y(t_{j-1}) + y(t_{j-2})}{2h} + \dfrac{h^2}{3} g'''(\tilde{c_j}) \end{cases}$$

- Step4: Drop $\mathcal{O}(h^r)$, $r = 1, 2 \cdots$
  Replace $y(t_j)$ by y_j

LEAP-FROG Scheme:

$$\begin{aligned} y_{j+1} &= y_{j-1} + 2hf(t_j, y_j) \\ y_0 &= \alpha \\ y_1 &= something \end{aligned}$$

$j = 1, 2 \cdots$

$$\begin{aligned} 3y_j - 4y_{j-1} + 4y_{j-2} &= 2hf(t_j, y_j) \\ y_0 &= \alpha \\ y_1 &= something & j \geqslant 2 \end{aligned}$$

- For both local truncation error $\tau_j = \mathcal{O}(h^2)$
- Two-step method
- Leap-frog is explicit, the other one is implicit

For multi-step method: How to get y_j?
Recall:

$$\begin{aligned} y(t_1) &= y(t_0) + hy'(t_0) + \frac{h^2}{2} y''(t_0) + \mathcal{O}(h^3) \\ y'(t_0) &= f(t_0, y(t_0)) = f(0, \alpha) \\ \implies y(t_1) &= \alpha + hf(0, \alpha) + \mathcal{O}(h^2) \end{aligned}$$

One way to get y_1: $y_1 = \alpha + hf(0,\alpha)$ (second order approximation)
One can also examine absolute stability for 2-step methods:
For instance: Apply leap-frog method to

$$\begin{aligned} y' &= -ry \qquad\qquad t>0 \\ y(0) &= \alpha \end{aligned}$$

$y_{j+1} = y_{j-1} - 2hry_j$
To solve: Set

$$\begin{aligned} y_j &= s^j \\ s &= s_1, s_2 \\ y_j &= c_1 s_1^j + c_2 s_2^j \end{aligned}$$

$\implies |y_j| \to \infty$ as $j \to \infty$
$\implies$ leap- frog is unstable for any $h > 0$

6.5 Numerical method based on numerical integration

To solve

$$\begin{aligned} y' &= f(t,y) \qquad\qquad t>0 \\ y(0) &= \alpha \end{aligned}$$

1. Mesh:
$$0 = t_0 < t_1 < \cdots < t_M = T$$
$h = \dfrac{T}{M}$
2. Integrate with the equation over "some intervals"
For example, we can take $[t_j, t_{j+1}]$
$$\begin{aligned} \int_{t_j}^{t_{j+1}} y'(t)dt &= \int_{t_j}^{t_{j+1}} f(t,y(t))dt \\ y(t_{j+1}) - y(t_j) &= \int_{t_j}^{t_{j+1}} f(t,y(t))dt \end{aligned}$$
3. Replace integral by numerical integration
$$\int_{t_j}^{t_{j+1}} f(t,y(t))dt = \begin{cases} hf(t_j,y(t_j)) - \dfrac{h^2}{2}f'(c_j) & \text{Left-rect} \\ hf(t_{j+1},y(t_{j+1})) + \dfrac{h^2}{2}f'(c_j) & \text{Right-rect} \\ \dfrac{h}{2}(f(t_j,y(t_j)) + f(t_{j+1},y(t_{j+1}))) - \dfrac{h^3}{12}f''(c_j) \end{cases}$$
4. Drop $\mathcal{O}(h^r)$ term and replace $y(t_j)$ by y_j, we'll get the scheme
$$y_{j+1} = y_j + \begin{cases} hf(t_j,y(t_j)) & \text{FE} \\ hf(t_{j+1},y(t_{j+1})) & \text{BE} \\ \dfrac{h}{2}(f(t_j,y(t_j)) + f(t_{j+1},y(t_{j+1}))) & \text{Trap} \end{cases}$$

Trapezoidal method:

- one- step
- Implicit

Absolute stability of Trap. method
Apply the method to the test equation:

$$y' = -ry \qquad t > 0 (r > 0)$$

$$\text{and get } y_{j+1} = y_j + \frac{h}{2}(-ry_j - ry_{j+1})$$
$$\implies y_{j+1} = \frac{2-hr}{2+hr} y_j$$

Absolute stability $\iff |Q(hr)| \leqslant 1 \iff \left|\frac{2-hr}{2+hr}\right| \leqslant 1$
This holds for any $h > 0$, hence Trap. method is unconditionally stable.
Local Truncation error of Trap. method:
Compatible form of the scheme to the equation:

$$y' = f(t, y)$$
$$\frac{y_{j+1} - y_j}{h} = \frac{1}{2}(f(t_j, y_j) + f(t_{j+1}, y_{j+1}))$$

Local truncation error: $y(t)$ is exact solution

$$\tau_j = \frac{y(t_{j+1}) - y(t_j)}{h} - \frac{1}{2}(f(t_j, y(t_j) + f(t_{j+1}, y(t_{j+1}))))$$
$$= \frac{h^2}{12} y'''(t_j) + \mathcal{O}(h^3)$$
$$= \mathcal{O}(h^2)$$

Three method: when h is big, BE may be better than Trap, but for small h, Trap is the best.

Runge-Kutta Method

Revisit Trap. Method:

$$y_{j+1} = y_j + \frac{h}{2}(f(t_j, y_j) + f(t_{j+1}, y_{j+1}))$$

(Implicit)
A variant, by predicting y_{j+1}

$$\hat{y_{j+1}} = y_j + hf(t_j, y_j)$$
$$y_{j+1} = y_j + \frac{h}{2}(f(t_j, y_j) + f(t_{j+1}, \hat{y_{j+1}}))$$

(one step, second order, "explicit trap method")
Based on numerical integration: "using midpoint rule"

$$y(t_{j+1}) - y(t_j)$$
$$= \int_{t_j}^{t_{j+1}} f(t, y(t)) dt$$
$$\approx f(t_{j+\frac{1}{2}}, y(t_{j+\frac{1}{2}})) \cdot h$$
$$y_{j+1} = y_j + hf(t_{j+\frac{1}{2}}, y_{j+\frac{1}{2}})$$

So, predicted:

$$\hat{y}_{j+\frac{1}{2}} = y_j + \frac{h}{2} f(t_j, y_j)$$
$$y_{j+1} = y_j + h f(t_{j+\frac{1}{2}}, \hat{y}_{j+\frac{1}{2}})$$

The one step methods we've seen so far, are all special case for RK methods.
General methods of RK methods:
To solve $y = f(t, y)$, $t > 0$, given $y_j \approx y(t_j)$.
Goal: To get $y_{j+1} \approx y(t_{j+1})$
A RK method can be written as

$$y_{j+1} = y_j + h \sum_{i=1}^{S} b_i(t_j + c_i h, Y_i)$$
$$Y_i = y_j + h \sum_{k=1}^{S} a_{ik} f(t_j + c_k h, Y_k)$$

Butcher table: $\begin{array}{c|c} c & \text{A} \\ \hline & b^T \end{array}$

FE: (s=1) $\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}$

BE: (s=1) $\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array}$

Explicit Trapezoidal: (s=2) $\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$

Explicit midpoint: (s=2) $\begin{array}{c|cc} 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \hline & 0 & 1 \end{array}$

Trapezoidal: (s=2) $\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$

Example: The classical 4-stage, 4th order explicit RK method(RK4) (s=4) $\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$

$$y_{j+1} = y_j + \frac{h}{6}(f(t_j, Y_1) + 2f(t_{j+\frac{1}{2}}, Y2) + 2f(t_{j+\frac{1}{2}}, Y3) + f(t_{j+1}, Y4)$$

where

$$Y_1 = y_j$$
$$Y_2 = Y_1 + \frac{h}{2} f(t_j, Y_1)$$
$$Y_3 = y_j + \frac{h}{2} f(tj + \frac{1}{2}, Y_2)$$
$$Y_4 = y_j + h f(t_{j+1}, Y_3)$$