Class Notes

AMSC460 - Computational Methods
Professor Stefan
University of Maryland, College Park

Wenyu Chen

Spring 2023

Contents

1 | Preliminary

Big O Notation

Definition 1.0.0.0.1
$f(n) = O(g(n))$ as $n \to \infty$ if $\exists N, M > 0$ such that

$$|f(n)| \leq Mg(n) \forall n \geq N$$

Note: As $n \to 0^+$, n^2 dominates n^3 so $(n^3) = O(n^2)$ as $n \to 0^+$

Example: $n^2 = O(n^3)$ as $n \to \infty$. We want M, N such that

$$|n^2| \leq Mn^3, \forall n \geq N$$

So let $M = 1, N = 2$ since $1 \leq 1 \cdot n$ holds $\forall n \geq 2$

Example: $n^2 \neq O(n)$ as $n \to \infty$. Suppose $\exists M, N$ such that $n^2 \leq Mn$ for all $n \geq N$, then $n \leq M, \forall n \geq N$. However when $n = max(M+1, n+1)$, then $n \geq M$, a contradiction.

Example: $n^3 + 2n^2 - n = O(n^3)$

Proof. By triangle inequality,

$$\begin{aligned} |n^3 + 2n^2 - n| &\leq n^3 + 2n^2 + n \\ &\leq n^3 + 2n^3 + n^3 \\ &\leq 4n^3 \end{aligned}$$

for all $n \geq 1$ □

Definition 1.0.0.0.2
We say $f(h) = O(g(h))$ as $h \to 0^+$ if $\exists M, \sigma > 0$ such that

$$|f(h)| \leq Mg(h)$$

$\forall h \in (0, \delta)$

Example: $h^2 = O(h)$ as $h \to 0^+$. We want M, δ such that

$$|f(h)| \leq Mg(h) \qquad \forall 0 < h < \delta$$

We can set $M = 2, \delta = 2$.

Theorem 1.1
Properties:

1. *$O(n^p \pm n^q) = O(n^p)$ as $n \to \infty$ if and only if $p \geq q$. But if $n \to 0^+$, then $O(n^p \pm n^q) = O(n^q)$*
2. *$O(cn^p) = O(n^p)$ assumed c is constant dependent of n*
3. *$O(f_1 f_2) = O(f_1)O(f_2)$*

Taylor Expansions

Definition 2.0.0.0.1
Taylor Expansion

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)(x_0)}}{k!}(x - x_0)^k$$

Another version:
Let $x = x_0 + h$, then

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{1}{2}h^2 f''(x_0) + \cdots$$

Truncating, we have

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0) + \cdots + \frac{1}{n!}f^{(n)}(x_0)(x - x_0)^n$$

Definition 2.0.0.0.2
Lagrange Remainder Term

$$\begin{aligned} f(x) &= \sum_{k=0}^{n} \frac{f^{(k)(x_0)}}{k!}(x - x_0)^k + \frac{f^{n+1}(Cn_1 x)}{(n+1)!}(x - x_0)^{n+1} \\ &= \sum_{k=0}^{n} \frac{f^{(k)(x_0)}}{k!}(x - x_0)^k + O((x - x_0)^n + 1) \end{aligned}$$

as $x \to x_0$. where $Cn_1 x$ is between x and x_0

Binary and Floating Point

Definition 3.0.0.0.1
Binary*, $b_i = 0$ or 1, a bit. 1 byte = 8 bits. 1 kb = $2^1 0 = 1024$ bits.*

$$x = \cdots + b_{-2} \cdot 2^{-2} + b_{-1} \cdot 2^{-1} + b_0 \cdot 2^0 + b_1 \cdot 2^1 + \cdots$$

Example: (terminating binary)

$$\begin{aligned} (100.1)_2 &= 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 + 1 \cdot 2^{-1} \\ &= (4.5)_{10} \end{aligned}$$

Example: (non terminating binary)
$x = (.\overline{10})_2 = (.10101010 \cdots)_2$
$2^2 x = (10.\overline{10})_2$, then

$$(2^2 - 1)x = (10)_2 = (2)_{10} \implies x = \left(\frac{2}{3}\right)_{10}$$

Decimal to Binary
Example: $(5.4)_{10}$
Integer Part: $(5)_{10}$, note that

$$\begin{aligned} \frac{5}{2} &= 2R1 \\ \frac{2}{2} &= 1R0 \\ \frac{1}{2} &= 0R1 \end{aligned}$$

So we get 101 as result (keep track of remainders backward). Then, we consider the fractional part $(.4)_{10}$,

$$\begin{aligned} .4 \times 2 &= .8 = .8 + 0 \\ .8 \times 2 &= 1.6 = .6 + 1 \\ .6 \times 2 &= 1.2 = .2 + 1 \\ .2 \times 2 &= .4 = .4 + 0 \end{aligned}$$

Notice it repeats to .4 So the answer is $(.\overline{0110})_2$ (keep track of remainder forward).
Then the final answer is $(5.4)_{10} = (101.\overline{0110})_2$. So we can not store exactly on a computer!

Floating Point Numbers
To **Normalized Base 2 Floats**, we do

$$[\pm]1 \cdot \text{mantissa} \times 2^E$$

where $[\pm]$ is sign and 1 is the leading 1 (normalization), mantissa consists of bites $b_i = 0, 1$ and E is the exponent
Note: 0 is subnormal, i.e not normalized
Example:

$$\begin{aligned} &(101.011)_2 \\ = &+1.01011 \times 2^2 (\text{note in base 2 times } 2^2 \text{ will shift the decimial two times to the right} \end{aligned}$$

precision	sign	mantisa	exponent	Total Bits
Double	1 (bits)	52	11	64
Single	1	23	8	32

So How floats are stored internally? It is the process

$$x \in \mathbb{R} \to float(x) \to \text{machine word}$$

s(0,1)	$E_1 \cdots$ (Exponent bits $E\_i = 0, 1$) $\cdots E_{11}$	$b_1 \cdots$ (mantissa bits $b\_i = 0, 1$) $\cdots b_{52}$

Remark:

1. True Range for E is

$$-1022 \leq E \leq 1023 \text{ in double}$$

   So $1023 + 1 + 1022 = 2046$ possible exponents. Note that $2^{11} = 2048$ possible exponents, the missing two are used for special cases (0,subnormal, infty, NaN,...)

2. The sign of the true exponent is not stored. Instead we store $E + (2023)_{10}$ where $(1023)_{10}$ is the exponent bias, which avoids storing an extra sign
   Example: If $E = 1$, we store $E + 2023 = 1024 = 2^{10} = (01000 \cdots 00)_2$

Example: In order to store repeating number , we have to round, so

$$\begin{aligned} &(9.4)_{10} = (1001.\overline{0110})_2 \\ &fl(9.4) = +1.(00101100 \cdots 1100)\overline{1100} \times 2^3 \end{aligned}$$

where the the number in paranthesis is b_{52} and we need to somehow round the rest bits

1.3.1 Rounding To Nearest Rule

Given a double precision floating number, we have

$$\pm 1.(b_1 \cdots b_{52})b_{53}b_{54} \times 2^E$$

Then,

1. if $b_{53} = 1$ and $b_n \neq 0$,for some $n > 53$. Then round up by adding 1 to b_{52}
2. if $b_{53} = 0$, round down by cutting away $b_n, \forall n \geq 53$
3. if $b_{53} = 1$ and $b_n = 0, \forall n > 53$, then if $b_{52} = 1$, round up by adding 1 to b_{52} and cutting remainder. If $b_{52} = 0$, round down by cutting remainder.
   The point is to cancel out equally likely errors.

Example:

$$\begin{aligned} &(9.4)_{10} = (1001.\overline{0110})_2 \\ &fl(9.4) = +1.(00101100 \cdots 1100)\overline{1100} \times 2^3 \end{aligned}$$

By rounding rule (in double precision), we lose $.\overline{1100} \times 2^3 \times 2^{-52} = 0.8 \times 2^{-49}$ and by rounding b_{52} up, we gain $2^{-52} \times 2^3 = 2^{-49}$, and by rounding b_{52} up, we gain $2^{-2} \times 2^3 = 2^{-49}$. Therefore

$$fl(9.4) = +1.(00101100 \cdots 1101) \times 2^3$$

In decimial,

$$\begin{aligned} fl(9.4) &= 9.4 - 0.8 \times 2^{-49} + 2^{-49} \\ &= 9.4 + 0.2 \times 2^{-49} \end{aligned}$$

which is actually stored in double precision.

Machine Epsilon

Definition 4.0.0.0.1
We define ϵmach (or ϵm) to be the distance between 1 and the next largest float.

Example:

$$\begin{aligned} fl(1) &= +1.0 \cdots 0 \times 2^0 \\ fl(1+\epsilon) &= +1.0 \cdots 01 \times 2^0 \\ &= 1 + 2^{-52} \end{aligned}$$

So in double precision,

$$\epsilon\text{mach} = 2^{-52}$$

Definition 4.0.0.0.2
Relative Error: *For normalized floats with rounding to nearest rule,*

$$\frac{|fl(x) - x|}{|x|} \leq \frac{1}{2}\epsilon m$$

for any storable $x \in \mathbb{R}$

Note: Some sources define $\frac{1}{2}(2^{-52}) = 2^{-53}$ as the machine epsilon.

2 Root Finding

Bisection Method

Root finding is find x such that $f(x) = 0$.

Theorem 1.1
Let f be a continuous function on $[a, b]$. If f changes sign from positive to negative, then there is a root $c \in [a, b]$ such that $f(c) = 0$

Given a function where it change sign on $[a_0, b_0]$, then we know by IVT, $\exists r \in [a, b]$ where $f(r) = 0$. Then, we bisect to get c_0. If $f(a_0)f(c_0) < 0$, let $[a_1, b_1] = [a_0, c_0]$, else if $f(c_0)f(b_0) < 0$, let $[a_1, b_1] = [c_0, b_0]$. Else if $f(c_0) = 0$, let $r = c_0$.
And we loop the entire process, until error tolerance reached.

Error Analysis:
$c_0 = \frac{a_i+b_i}{2}$

$$\begin{aligned}
\text{error } e_0 &:= |r - c_0| \leq \frac{b_0 - a_0}{2} \\
\text{error } e_1 &:= |r - c_1| \leq \frac{b_0 - a_0}{2^2} \\
&\vdots \\
\text{error } e_n &:= |r - c_n| \leq \frac{b_0 - a_0}{2^{n+1}}
\end{aligned}$$

If tolerence given,

$$\frac{b_0 - a_0}{2^{n+1}} < Tol$$

and solve for n to get number of steps needed.

Note that bisection can fail. For example, $|x|$

Fixed Point Iteration

Consider $\cos(x) = 0$, then $cos(x) + x = x$. Setting $g(x) = \cos x + x$, we have $g(x) = x$

Definition 2.0.0.0.1
A point x^* such that

$$f(x^*) = x^*$$

is called a fixed point of f

FPI: We guess x_0 for a solution, and

$$\begin{aligned}
f(x_0) &= x_1 \\
f(x_1) &= x_2 \\
f(x_2) &= x_3
\end{aligned}$$

In general, evaluate $x_{n+1} = f(x_n), x_0 =$ starting guess. Then loop until error less than TOL. The code will be something like

$x = x_0$
While error $<$ TOL:
$x = f(x)$

Theorem 2.1
Let $f \in C[a,b]$. If FPI converges, it converges to solution.

Let $x_n \to x$, then

$$\lim_{n\to\infty} x_{n+1} = f(x_n)$$
$$\implies x = f(x)$$

Example:

1. $x = g(x) = \frac{1}{10}x + 1$. Then $\begin{cases} x_{n+1} = \frac{1}{10}x_n + 1 \\ x_0 = 0 \end{cases}$. So we have

$$\begin{aligned} x_0 &= 0 \\ x_1 &= 1 \\ x_2 &= 1.11 \\ &\vdots \\ x_\infty &= 1.\bar{1} = \frac{10}{9} \end{aligned}$$

2.$x = g(x) = 3x + 1$. Then $\begin{cases} x_{n+1} = 3x_n + 1 \\ x_0 = 0 \end{cases}$. So we have

$$\begin{aligned} x_0 &= 0 \\ x_1 &= 1 \\ x_2 &= 4 \\ x_3 &= 13 \\ &\vdots \\ &\text{diverges!} \end{aligned}$$

Even though $x = -\frac{1}{2}$ is a solution, but we fail to find it with the method.

2.2.1 Convergence of FPI

Definition 2.1.0.0.1
A contraction on $[a,b]$ $f(x)$ satisifies that

$$|f(x) - f(y)| < L|x - y|$$

For all $x, y \in [a,b]$ where $0 \leq L < 1$

Theorem 2.2
Contraction Mapping Theorem:
Let $g : [a,b] \to [a,b]$ be a contraction on $[a,b]$. Then,

1. *$\exists x^* \in [a,b]$ such that $g(x^*) = x^*$*
2. *FPI converges starting from any $x_0 \in [a,b]$*

Remark: Suppose $|g'(x)| < 1$ for all $x \in [a,b]$. Then g is a contradiction.

Proof.

$$|g(x) - g(y)| = |g'(c)(x - y)|$$

By mean value theorem that $g'(c) = \frac{g(x)-g(y)}{x-y}$ for some c between x and y. Then

$$\leq L|x - y|$$

since $|g'(x)| < 1$ for all $x \in [a,b]$ □

Remark: If we know $g'(x^*) < 1$ at the fixed point, then FPI is logically convergent, i.e $\exists$ a possibly small interval in $[a,b]$ such that with any x_0 in the interval, FPI works.
Remark: Stooping criterion

1. $|x_n - x_{n-1}|$ <TOL (absolute error)
2. $|g(x_n) - x_n|$ <TOL (backward error)
3. $\left|\frac{x_n - x_{n-1}}{x_n}\right|$ <TOL (relative error)

Newton Method

Recall, Taylor Expansion is

$$\begin{aligned} f(r) &= f(x_0) + f'(x_0)(x - x_0) + O((x - x)^2) \\ &\approx f(x_0) + f'(x_0)(x - x_0) \\ &=: \lambda(x) \end{aligned}$$

So

$$\lambda(x_1) = f(x_0) + f'(x_0)(x_1 - x_0)$$

So

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

Here is the method,

$$\begin{cases} x_0 & = \text{starting guess} \\ x_{n+1} & = x_n - \frac{f(x_n)}{f'(x_n)} \end{cases} = \qquad g(x_n)$$

Local Convergence?
Let r be the solution.

$$g'(r) = \left| 1 - \frac{(f'(r))^2 - f(r)f''(r)}{[f'(r)]^2} \right| = 0$$

locally convergent!

Secant Method

Recall Newton, $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$. Then, we can approximate

$$f'(x_n) \approx \frac{f(x_n) - f(x_{n-1})}{f(x_n) - f(x_{n-1})}$$

Therefore, the method is

$$\begin{cases} x_{n+1} = x_n - (x_n - x_{n-1}) \frac{f(x_n)}{f(x_n) - f(x_{n-1})} \\ x_0, x_1 = \text{starting guesses} \end{cases}$$

Rate of Convergence:
Let $e_n := |x_n - r|$, the absolute error. We say an iterative method convergence with order p if for some $c \in \mathbb{R}$, we have

$$\lim_{n\to\infty} \frac{e_{n+1}}{e_n^p} = c$$

For $p = 1$, we need $0 < c < 1$

3 Linear Systems

Example:
We want find $p \in P_{n-1}$ such that

$$y_i = p(x_i), \leq i \leq n$$

Let

$$p(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1 x + a_0$$

Find $\{a_i\}_{i=0}^{n-1} = \vec{a}$ is a susyem of sequences in n unkown

$$A\vec{a} = \vec{y}$$

with $A \in \mathbb{R}^{n\times n}$.

$$\vec{x} \in R^n, \vec{b} \in \mathbb{R}^n, A \in R^{n\times n}$$

Solve

$$A\vec{x} = \vec{b}$$

Componentwise Calculations

1. $\vec{x} \cdot \vec{y} = \sum_{i=1}^n x_i y_i$
2. $(A\vec{x})_i = \sum_{i=1}^n a_{ij} x_j$
3. $(AB)_{ij} = \sum_{k=1}^n A_{ik} B_{kj}$

Flop Count

To computer $A\vec{x}$, for every $i \in \{i, \cdots, n\}$, we need $n-1$ additions, **n** mulplications. So Total is $2n-1 = O(n)$. Grand Total

$$\begin{aligned} n(2n-1) &= 2n^2 - n \\ &= O(n^2) \end{aligned}$$

Cost to obtain AB is $O(n^2)$ i.e $(AB)x$ cost $O(n^3)$ while $A(Bx)$ costs $O(n^2)$

3.1.1 Gaussian Elimination

$$\left(\begin{array}{ccc|c} 1 & 2 & -1 & 2 \\ 0 & 3 & 1 & 4 \\ 2 & -2 & 1 & 2 \end{array}\right) \xrightarrow[R_3 - 2R_1]{} \left(\begin{array}{ccc|c} 1 & 2 & -1 & 2 \\ 0 & 3 & 1 & 4 \\ 0 & -6 & 3 & -2 \end{array}\right) \xrightarrow[R_3 + 2R_3]{} \left(\begin{array}{ccc|c} 1 & 2 & -1 & 2 \\ 0 & 3 & 1 & 4 \\ 0 & 0 & 5 & 6 \end{array}\right)$$

From here, backsolve to get

$$\begin{aligned} x_3 &= \frac{6}{5} \\ x_2 &= \frac{4 - x_3}{3} = \frac{14}{15} \\ x_1 &= \frac{2 + x_3 - 2x_2}{1} \\ &= 2 + \frac{14}{15} - 2\frac{6}{5} = \frac{4}{3s} \end{aligned}$$

Note that elimination step is $O(n^3)$ but backsolving is $O(n^2)$
Algorithm:
Suppose have $A\vec{x} = \vec{c}$, and have eliminated to get

$$U\vec{x} = \vec{b}$$

where U is upper triangular
Then

$$\begin{aligned} U_{11}x + 1 + U_{12}x_2 + \cdots + U_{1n}x_n &= b_1 \\ U_{22}x_2 + \cdots + U_{2n}x_n &= b_2 \\ \ddots \vdots &= \vdots \\ u_{(n-1)(n-1)}x_{n-1} + u_{(n-1)n}x_n &= b_{n-1} \\ u_{nn}x_n &= b_n \end{aligned}$$

Bavksolving: we will have

$$x_n = \frac{b_n}{u_{nn}}$$

Then, for $i = n-1, n-2, \cdots, 2, 1$ we have

$$x_i = \frac{1}{u_{ii}}\left(b_i - \sum_{j=i+1}^{n} u_{ij}x_j\right)$$

Exercise: Show flop count to consturct $\vec{x}$ is exactly n^2.

MATLAB Code:

```
% Backsolving
x(n) = b(n)/u(n,n);
for i = n-1:-1:1
    x(i)=(1/(u(i,i)))[b(i)-u(i,i+1:n)*x(i+1:n)];
end
```

3.1.2 Lu Decomposition

Matrix representation of Guass elmination:

$$A = LU$$

where L is lower triangular matrix and U is upper triangular matrix.
To solve $A\vec{x} = b$, we have

$$LU\vec{x} = \vec{b}$$

Let $\vec{c} = U\vec{x}, L\vec{c} = \vec{b}$. Solve $L\vec{c} = \vec{b}$ for $\vec{c}$ by forward substitution (costs $O(n^2)$)
Then solve $U\vec{x} = \vec{c}$ by back substitution (costs $O(n^2)$)

Example:

$$\begin{aligned} A\vec{x}^1 &= \vec{b}^1 \\ A\vec{x}^2 &= \vec{b}^2 \\ \vdots &= \vdots \\ A\vec{x}^r &= \vec{b}^r \end{aligned}$$

Suppose $r \approx n$. If using LU, costs to solve is

$$O(n^3 + n^2(2r)) = O(n^3)$$

where n^2 is elmination and $n^2(2r)$ is backward/forward solve.

If elminating each time, it cost

$$O(n^3 \cdot r + n^2(2r)) = O(n^4)$$

Obtaining LU
Example:

$$A = \begin{pmatrix} 1 & 2 & -1 \\ 0 & 3 & 1 \\ 2 & -2 & 1 \end{pmatrix}$$

1 is the pivot entry. Subtract $2R_1$ from R_3, $0R)1$ from R_2 we have

$$\begin{pmatrix} 1 & 2 & -1 \\ (0) & 3 & 1 \\ (2) & -6 & 3 \end{pmatrix}$$

for the number in (), there are really 0 here, we are just storing the multipliers. Then, subtract $(-2) * R_2$ from R_3, we have

$$\begin{pmatrix} 1 & 2 & -1 \\ (0) & 3 & 1 \\ (2) & (-2) & 5 \end{pmatrix}$$

Then, we get

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & -2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 & -1 \\ 0 & 3 & 1 \\ 0 & 0 & 5 \end{pmatrix}$$

Pitfalls

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Since 1 can not be eliminated, so A has no LU decomposition.

$$A = \begin{pmatrix} \epsilon & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \frac{1}{\epsilon} & 1 \end{pmatrix} \begin{pmatrix} \epsilon & 1 \\ 0 & 1 - \frac{1}{\epsilon} \end{pmatrix}$$

with $\epsilon = 10^{-20}$ which is less than $\epsilon_m \approx 10^{-16}$. Therefore, in double precision

$$\begin{pmatrix} 1 & 0 \\ \frac{1}{\epsilon} & 1 \end{pmatrix} \begin{pmatrix} \epsilon & 1 \\ 0 & -\frac{1}{\epsilon} \end{pmatrix} = \begin{pmatrix} \epsilon & 1 \\ 1 & 0 \end{pmatrix} \neq A$$

Since $1 - \frac{1}{\epsilon} = 1 - 10^{20}$ which is stored as $= 10^{20}$.
But if

$$\begin{pmatrix} 1 & 1 \\ \epsilon & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \epsilon & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 - \epsilon \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \epsilon & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ \epsilon & 1 + \epsilon \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ \epsilon & 1 \end{pmatrix}$$

Pa=LU Decomposition

$$A = \begin{pmatrix} \epsilon & 1 \\ 1 & 1 \end{pmatrix} \implies PA = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \epsilon & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ \epsilon & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \epsilon & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = LU$$

$P = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ is the permutation matrix, a matrix with a single 1 in each row and column.
Therefore, $PA = LU$.
To solve $Ax = b$ effectively by $PA = LU$, we do

$$Ax = b$$
$$PAx - Pb$$
$$LUx = Pb$$

Let $Ux = c$. First solve $Lc = Pb$ for c, cost is $O(n^2)$. Then, solve $Ux = c$ for x, cost $O(n^2)$
Example: 3×3 example

$$A = \begin{pmatrix} 2 & 1 & 5 \\ 4 & 4 & -4 \\ 1 & 3 & 1 \end{pmatrix}$$

Swap R_1 and R_2 using $P_1 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & \\ 0 & 0 & 0 \end{pmatrix}$. Then, we have

$$P_1 A = \begin{pmatrix} 4 & 4 & -4 \\ 2 & 1 & 5 \\ 1 & 3 & 1 \end{pmatrix}$$

And elminated, we get

$$\begin{pmatrix} 4 & 4 & -4 \\ (\frac{1}{2}) & -1 & 7 \\ (\frac{1}{4}) & 2 & 2 \end{pmatrix}$$

And Swap R_2 and R_3 using $P_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$, we get

$$\begin{pmatrix} 4 & 4 & -4 \\ (\frac{1}{4}) & 2 & 2 \\ (\frac{1}{2}) & -1 & 7 \end{pmatrix}$$

And subtract $(-\frac{1}{2})R_2$ from R_3,

$$\begin{pmatrix} 4 & 4 & -4 \\ (\frac{1}{4}) & 2 & 2 \\ (\frac{1}{2}) & (-\frac{1}{2}) & 8 \end{pmatrix}$$

Now, $P_2 P_1 A = LU = \begin{pmatrix} 1 & 0 & 0 \\ (\frac{1}{4}) & 1 & 0 \\ (\frac{1}{2}) & (-\frac{1}{2}) & 1 \end{pmatrix} \begin{pmatrix} 4 & 4 & -4 \\ 0 & 2 & 2 \\ 0 & 0 & 8 \end{pmatrix}$ and $P_2 P_1 = P$

Errors in Linear Systems

x_a = approximate solutions. $Ax = b, x, b \in R^n$. Yhen

Definition 3.0.0.0.1
Residual:

$$r := Ax_n - b$$

Definition 3.0.0.0.2
Backward error

$$||r|| = ||Ax_a - b||$$

Definition 3.0.0.0.3
Forward Error

$$|x - x_a|||$$

Definition 3.0.0.0.4
Norms on R^n:

1. *Euclian or 2-norm*

$$||u||_2 = \left(\sum_{i=1}^{n} |u_i|^2 \right)^{\frac{1}{2}}$$

2. *1-norm*

$$||u||_1 = \sum_{i=1}^{n} |u_i|$$

3. *Max norm*

$$||u||_\infty = max_{i \leq \leq n} |u_i|$$

4. *p norm,* $1 \leq < \infty$

$$||u||_p = \left(\sum_{i=1}^{n} |u_i|^p \right)^{\frac{1}{p}}$$

Definition 3.0.0.0.5
Matrix Norms:

1. *Frobeneous norm*

$$||A||_F = \left(\sum_{i,j=1}^{n} |a_{ij}|^2 \right)^{\frac{1}{2}}$$

2. ***Operator p-norm***

$$\begin{aligned} ||A||_p &:= max \frac{||Ax||_p}{||x||_p} \\ &= max_{||x||=1} ||Ax||_p \end{aligned}$$

where $x \neq 0$

Example:

$$\begin{aligned} ||A||_1 &= \max_{1 \leq j \leq n} \left(\sum_{i=1}^{n} |a_{ij}| \right) \\ &= \text{maximum absolute column sum} \end{aligned}$$

Example:

$$\begin{aligned} ||A||_\infty &= \max_{1 \leq i \leq n} \left(\sum_{i=j}^{n} |a_{ij}| \right) \\ &= \text{max absolute row sum} \end{aligned}$$

Example:

$$||A||_2 = \sqrt{p(A^t A)}$$

where $p(b) = \max_{1 \leq i \leq n} |\lambda_i|$ where λ_i is an eigenvalue of B.

3.3.1 Errors for Ax=b

Let $A\widetilde{x} = \widetilde{b}$, where $\widetilde{b}$ is computer store b in double precision.
Then

$$\begin{aligned} A(x - \widetilde{x}) &= b - \widetilde{b} \\ x - \widetilde{x} &= A^{-1}(b - \widetilde{b}) \\ ||x - \widetilde{x}|| &\leq ||A^{-1}|| ||b - \widetilde{b}|| \end{aligned}$$

Also,

$$||b|| = ||Ax|| \leq ||A|| ||x||$$

Combine them, we have

$$\frac{||x-\widetilde{x}||}{||A||||x||} \leq \frac{||A^{-1}||||b-\widetilde{b}||}{||b||}$$

Then,

$$\frac{||x-\widetilde{x}||}{||A||||x||} \leq \frac{||A||_p||A^{-1}||_p||b-\widetilde{b}||_p}{||b||_p}$$

A matrix with large condition number is called ill conditioned

Theorem 3.1
In finite dimensions, all norms are equivalent

Iterative Methods

$A\vec{x} = \vec{b}$. Then if $A \in \mathbb{R}^{10^5 \times 10^5}$, A has 10^10 entries. In double precision, each $a_{ij}$is 8 bytes of memory. Therefore, $8 \times 10^10 = 80$ Gb memory. So, we can only store $\approx 30 \times 10^3$ entries, which is

$$8(30 \times 10^3) = 24 \times 10^4 \text{bytes}$$

Therefore, we had to store A as a sparse data structure.

Goal: We want to solve of the form

$$\begin{aligned} \vec{x}^{(n+1)} &= \vec{g}(\vec{x}^n) \\ \vec{x}^(0) &= \text{starting guess} \end{aligned}$$

3.4.1 Jacobi Method

:

$$Ax = b$$

Split $A = D + R$, where D is a diagonal matrix and R is matrix with 0 in diagonal line. Then, We have

$$\begin{aligned} (D+R)x &= b \\ Dx &= b - R_x \\ \implies x &= D^{-1}(b - Rx) \end{aligned}$$

Example:

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix} + \begin{pmatrix} 0 & 2 \\ 3 & 0 \end{pmatrix}$$

So Jacobi method:

$$\begin{cases} \vec{x}^{(n+1)} = D^{-1}(\vec{b} - R\vec{x}^n) \\ x\vec{0} = \text{starting guess} \end{cases}$$

Remark: D^{-1} is just take recipical of diagonal elements.
Remark: Convergence guranteed if

$$p(D^{-1}R) < 1$$

where p is spectral radius of $D^{-1}R$.

Remark: Convergence guranteed also If A is strictly diagonally dominant, i.e

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

Remark: Componentwise version:

$$\vec{x}_i^{(n+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} \vec{x}_j^n \right)$$

$1 \leq i \leq n$ Example:

$$2x + y = 5$$
$$x + 3y = 4$$

Starting guess $x^0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$. Then note that

$$\begin{cases} x &= \frac{5-y}{2} \\ y &= \frac{4-x}{3} \end{cases}$$

We get $x^1 = \begin{pmatrix} \frac{5}{2} \\ \frac{4}{3} \end{pmatrix}$. Remark: When we stop? check if

1. $||A\vec{x}^n - \vec{b}|| <$TOl
   or $\frac{||\vec{x}^{n+1} - \vec{x}^n}{||\vec{x}^n||}$
2. or if $n > N = (100)$

3.4.2 Gauss-Sedel Method

Let $A = L + D + U$, then

$$\begin{cases} \vec{x}^{(n+1)} = (L+D)^{-1} L\vec{b} - U\vec{x}^{(b)} \\ \vec{x}^{(0)} = \text{starting guess} \end{cases}$$

Componentwise

$$\vec{x}_i^{(n+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} \vec{x}_j^{(n+1)} - \sum_{j>i} a_{ij} \vec{x}_i^n \right)$$

4 Nonlinear Systems

$\begin{cases} \vec{x} = \vec{g}(\vec{x}) \\ \vec{x} \in \mathbb{R}^n \\ \vec{F}(\vec{x}) = \vec{0} \end{cases}$, and $\vec{g}$ is possibly non linear.

Example:

$$f_1(x, y) = e^x - y = 0$$
$$f_2(x, u) = xy - e^x = 0$$

Then,

$$\vec{F}(\vec{x}) = \begin{pmatrix} f_1(\vec{x}) \\ f_2(\vec{x}) \end{pmatrix} = 0\vec{0}$$

with $\vec{x} = \begin{pmatrix} x \\ j \end{pmatrix} = \mathbb{R}^2$.

Recall the 1d newton, $f(x) = 0$. And

$$f(x_1) = f(x_0) + f'(x_0)(x_1 - x_0) + O((x_1 - x_0)^2)$$

as $x_1 \to x_0$. Set $L(x_1) = f(x_0) + f'(x_0)(x_1 - x_0)$. Note that

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

Multivariable Newton's Method

$$\vec{F}(\vec{x}) = \vec{F}(\vec{x_0}) + D\vec{F}(\vec{x_0})(\vec{x} - \vec{x_0}) + \text{higher order terms}$$

Where

$$D\vec{F}(\vec{x_0}) = \begin{pmatrix} \partial_{x_1} f_1 & \partial_{x_2} f_1 & \cdots & \partial_{x_n} f_1 \\ \partial_{x_1} f_2 & \partial_{x_2} f_2 & \cdots & \partial_{x_n} f_2 \\ \vdots & \vdots & \ddots & \vdots \\ \partial_{x_1} f_m & \partial_{x_2} f_m & \cdots & \partial_{x_n} f_m \end{pmatrix} = \begin{pmatrix} \Delta f_1 \\ \Delta f_m \\ \vdots \\ \Delta f_m \end{pmatrix}$$

Set $\vec{x_1}$ such that

$$\begin{aligned} \vec{0} &= \vec{L}(\vec{x_1}) \\ &= \vec{F}(x_0) + D\vec{F}(\vec{x_0})(\vec{x_1} - \vec{x_0}) \\ -\vec{F}(x_0) &= D\vec{F}(\vec{x_0})(\vec{x_1} - \vec{x_0}) \end{aligned}$$

Therefore,

$$\vec{x_0} - [D\vec{F}(\vec{x_0})]^{-1}\vec{F}(\vec{x_0}) = \vec{x_1}$$

4.1.1 Netown Method

To solve ($\vec{F}(\vec{x}) = \vec{0}$), we do the following

1. Starting guess $\vec{x_0} \in \mathbb{R}^n$

2. At step n, have $\vec{x_n}$, to update, use

$$\vec{x_n} - [D\vec{F}(\vec{x_n})]^{-1}\vec{F}(\vec{x_0}) = \vec{x_{n+1}}$$

To avoid inverse, substitute $[D\vec{F}(\vec{x_n})]^{-1}\vec{F}(\vec{x_0})$ as $\vec{s}$. Then, solve

$$DF(\vec{x_n})\vec{s} = \vec{F}(\vec{x_n})$$

for $\vec{s}$. And then update

$$\vec{x_{n+1}} = \vec{x_n} - \vec{s}$$

Example:

$$f_1(x,y) = e^x - y = 0$$
$$f_2(x,u) = xy - e^x = 0$$

Then

$$\vec{F}(x,y) = \begin{pmatrix} e^x - y \\ xy - e^x \end{pmatrix}$$
$$D\vec{F}(x,y) = \begin{pmatrix} \partial_x f_1 & \partial_y f_1 \\ \partial_x f_2 & \partial_y f_2 \end{pmatrix}$$
$$= \begin{pmatrix} e^x & -1 \\ y - e^x & x \end{pmatrix}$$

Let's try $\vec{x_0} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ to start solve $D\vec{F}(0,0)\vec{s} = \vec{F}(0,0)$. Then

$$\begin{pmatrix} 1 & -1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$
$$\implies s_1 = 1, s_2 = 0, \vec{s} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Now $\vec{x}_1 = \vec{x_0} - \vec{s} = \vec{0} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$

4.1.2 Convergence Theory

Get local convergence provided that

$$p(D\vec{G}(\vec{r})) < 1$$

where $p()$ is the spectral radius, Then

$$\vec{G}(\vec{x}) = \vec{x} - [D\vec{F}(\vec{x})]^{-1}\vec{F}(\vec{x})$$

and $\vec{r}$ is the actual solution.

5 | Interpolation

Polynomial Interpolation

Suppose there are $n+1$ points. We can fit a unique $p_n \in P_n$ provided $\{x_i\}_{i=0}^n$ are distinct.

Proof. Suppose both $p, q \in P_n$ interpolate. Let $h(x) = p(x) - q(x)$. Then, $h \in P_n$ and $h(x) = 0$ for all $\{x_i\}_{i=0}^n$ (i.e at $n+1$ points.)
Then, by fundemental theorem of algebra, h is zero polynomial. Therefore, $p = q$. □

To find $P_n \in P_n$ interplotating $\{(x_i, y_i)\}_{i=0}^n$, we need a basis of P_n, which is

$$\mathscr{B} = \{1, x, x^2, \cdots, x^n\}$$

spans P_n and is linearly independent. So let

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

for unknown $\{a_i\}_{i=0}^n$. We force $y_i = p(x_i), \forall i$, to get

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Solve $V\vec{a} = \vec{y}$ for $\vec{a}$.
Remark: If $\{x_i\}$ distinct, then V^{-1} exists, so we can find $\vec{a}$
Remark: V is called a Vandermonde matrix which becomes very ill-conditioned for large n.

Intutively, for large n, basis vectors x^n become very similar, so columns of V get close to linear dependence (so V close to being singular)

Lagrange Basis

$$P_1 \to p_1(x) = y_0 \frac{x - x_1}{x_0 - x_1} + y_1 \frac{x - x_0}{x_1 - x_0}$$

$\frac{x-x_1}{x_0-x_1} = l_0(x)$ and $y_1 \frac{x-x_0}{x_1-x_0} = l_1(x)$. They are called Lagrange Basis of P_1.

Note:

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

Note: $p = a_0 l_0(x) + a_1 l_1(x)$, we get the systm

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix}$$

where $\text{cond}(I) = 1$, best possible.

In general, for $n+1$ data point,

$$p(x) = \sum_{i=0}^{n} y_i l_i(x)$$

where

$$l_i(x) = \prod_{i=0, j\neq i}^{n} \left(\frac{x - x_j}{x_i - x_j}\right)$$

Example: $(1, 2), (3, 7), (5, 8)$

$$p(x) = 2\frac{(x-3)(x-5)}{(1-3)(1-5)} + 7\frac{(x-1)(x-5)}{(3-1)(3-5)} + 8\frac{(x-1)(x-3)}{(5-1)(5-3)}$$

Newton Basis

$$\{(x_n, y_n)\}_{n=0}^{n+1}$$

$p \in P_n$ interplotion. Then,

$$p(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)\cdots(x - x_{n-1})$$

or, if prefered we can said

$$\begin{cases} N_0(x) = 1 \\ N_i(x) = \sum_{j=0}^{i-1}(x - x_j) \text{for } 1 \leq i \leq n \end{cases}$$

Example: 3 points case.

$$\begin{aligned} y_0 &= p(x_0) = a_0 \\ y_1 &= p(x_1) = a_0 + a_1(x_1 - x_0) \\ y_2 &= p(x_2) = a_0 + a_1(x_1 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) \end{aligned}$$

can eficently solve in $O(n^3)$ time by forward solving.

5.3.1 Divided differences

x	y	1st div difference	2nd div difference
1	1		
2	5	$\frac{5-1}{2-1} = \frac{4}{1}$	
3	4	$\frac{4-5}{3-2} = -1$	$\frac{-1-4}{3-1} = -\frac{5}{2}$

We will use the diagonal lines. Therefore,

$$\begin{aligned} p(x) &= 1 + 4(x - 1) \\ &= -\frac{5}{2}(x - 1)(x - 2) \end{aligned}$$

Since

$$f(x) = a_0 + a_1(x - x_0 a_2(x - x_0)(x - x_1)$$

1. $y_0 = f(x_0) = a_0$
   We define $f[x_0] := f(x_0)$, the 0th divided difference.

2. $y_1 = f(x_1) = a_0 + a_1(x_1 - x_0)$
   This implies

$$\begin{aligned} a_1 &= \frac{f(x_1) - f(x_0)}{x_1 - x_0} \\ &= \frac{f[x_1] - f[x_0]}{x_1 - x_0} = f[x_0, x_1] \end{aligned}$$

   the second divided difference.

General Theory
Let

$$f[x_0] = a_0$$
$$f[x_0, x_1] = a_1$$
$$f[x_0, x_1, x_2] = a_2$$
$$\vdots$$

where $f[x_0] = f(x_0)$ and we recursively define

$$f[x_i, \cdots, x_j] = \frac{f[x_{i+1}, \cdots, x_j] - f[x_i, \cdots, x_{j-1}]}{x_j - x_i}$$

for $0 \le i < j \le n$.
Then, the interploent is

$$p(x) = \sum_{k=0}^{n} f[x_0, \cdots, x_j] \prod_{i=0}^{j-1} (x - x_j)$$

with $\prod_{i=0}^{-1}(x - x_1) := 1$

Interpolation Error and Chebyshev Interpolation

Error

1. $||f - p||_{L^2(a,b)} = \left(\int_a^b |f(x) - p(x)|^2 dx \right)$
2. $||f - p||_{L^\infty[a,b]} = \max_{x \in [a,b]} |f(x) - p(x)|$

Theorem 4.1
Let $f \in C^{n+1}[a, b]$, and let $p \in P_n$ interpolate f at $\{x_0, x_1, \cdots, x_n\}$ (distinct points.)
Then, $\forall x \in [a, b]$, we have

$$e(x) := f(x) - p(x) = \frac{f^{(n+1)}(C_{n,x})}{(n+1)!} \prod_{j=0}^{n} (x - x_j)$$

Where $C_{n,x} \in (a, b)$

Note:

$$||f - p||_{L^\infty[a,b]} \le \frac{||f^{(n+1)}||_{L^\infty[a,b]}}{(n+1)!} \prod_{j=0}^{n} (x - x_j)$$

Which $\{x_j\}_{j=0}^{n}$ minimize

$$\omega(x) = \prod_{j=0}^{n} (x - x_j)$$

Note the polynomial.

Theorem 4.2
Optimal Node Spacing *for $P_n$interpolation*
Fix f. Fix $n \in \mathbb{N}$(degree of polynomial in P_{n-1})

Theorem 4.3
Chebyshev's Theorem
The choice of $\{x_j\}_{j=1}^{n}$ that minimize $|\omega(x)|$ on $[-1, 1]$ is

$$x_j = \cos \frac{(2i-1)\pi}{2n} (i = 1, 1, 2, \cdots, n)$$

and

$$||\omega||_{L^\infty[-1,1]} \le \frac{1}{2^{n-1}}$$

Example: Find an upper bound on error for approxinating $f(x) = e^x$ on $[-1, 1]$ using $p \in P_4$.
Using Chebyshev nodes, we know $x_j = \cos\frac{(2i-1)\pi}{2(5)}$,
We get $p \in P_r$ satisifying $j = 1, \cdots, n$. Then

$$||e^x - p||_{L^\infty[-1,1]} \leq \frac{||f^{(5)}||_{L^\infty[-1,1]}}{5!} \prod_{j=1}^{5} |x - x_k|$$
$$\leq \frac{e^1}{5!} \frac{1}{2^{5-1}}$$

Let $y = T(x) = \frac{b-a}{2}x + \frac{b+a}{2}$. Then $T : [-1, 1] \to [a, b]$ is linear. So $\{x_j\}_{k=1}^n$ Chebyshev on $[-1, 1]$, become $\{y_j\} = \{T(x_j)\}_{j=1}^n$ on $[a, b]$.

How does error transform?

$$|\omega(y)| = \left|\prod_{j=1}^{n}(y - y_j)\right|$$
$$= \left|\left(\frac{b-a}{2}\right)^n \prod_{j=1}^{n}(x - x_j)\right|$$
$$\leq \left(\frac{b-a}{2}\right)^n \frac{1}{2^{n-1}}$$

upper bound for $|\omega(y)|$ on $[a, b]$

Hermite Interpolation

Goal:Interploate $f(x_i)$ and also $f^{(e)}(x_i)$.

Let

$$p(x) = a + bx + cx^2$$

and force p to nterpolate $(0, 1), (1, -1)$ and $p'(1) = -1$ and solve for a, b, c
Example: Taylor polynomial

$$p(x) = \sum_{k=0}^{n} \frac{f^k(x_0)}{k!}(x - x_0)^k$$

is a Hermite interpolent satisfying $p^l(x_0) = f^l(x_0)$ for $0 \leq l \leq n$

x_1	$y_1 = f[x_1]$		
x_2	$y_2 = f[x_2]$	$f[x_1, x_2]$	
x_3	$y_3 = f[x_3]$	$f[x_2, x_3]$	$f[x_1, x_2, x_3]$

circle the diagonal entries, we have Newton form of $P_2(x)$, which is

$$p_2(x) = f[x_1] + f[x_1, x_2](x - x_1) + f[x_1, x_2, x_3](x - x_1)(x - x_2)$$

Let $x_2 \to x_1$, we have

$$p_2(x) = f[x_1] + \lim_{x_2 \to x_1} f[x_1, x_2](x - x_1) + \lim_{x_2 \to x_1} f[x_1, x_2, x_3](x - x_1)(x - x_2)$$
$$p_2(x) = f[x_1] + f[x_1, x_1](x - x_1)f[x_1, x_2, x_3](x - x_1)(x - x_2)$$

Example:

x	y		
1	$y_1 = -1$		
0	$y_2 = -1$ slope constraint at $x = 1$	-1	
0	$y_3 = 1$	$\frac{1-(-1)}{0-1} = -2$	$\frac{-2-(-1)}{0-1} = 1$

So $P_2 = -1 - 1(x-1) + 1(x-1)^2$

Remark:

1. Repeated points have to be grouped together
2. For points with mulpllicity of m (m copies of x_i), we suppose we know
$$f^{(l)}(x_i)$$
for $0 \le l \le m-1$. for Existinces, uniquness
3. $f[x_j, \cdots, x_j] = \frac{f^{m-1}(x_j)}{(m-1)!}$ where x_j has m repeations.

Example: Interpolate $f^{(l)}(x_0), 0 \le l \le 2$ with $p_2 \in \mathbf{P}_2$, note that then

x	$f[x_i]$	$f[x_i, x_{i+1}]$	$f[x_{i-1}, x_i, x_{i+1}]$
x_0	$f(x_0)$		
x_0	$f(x_0)$	$\frac{f'(x_0)}{1!}$	
x_0	$f(x_0)$	$\frac{f'(x_0)}{1!}$	$\frac{f''(x_0)}{2!}$

then

$$p_2(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{f''(x_0)}{2!}(x-x_0)^2$$

Piecewise Polynomial Interpolation

Have an $[x_{i-1}, x_i]$ the interpolant where

$$s_i(x) = f(x_{i-1}) + f[x_{n-1}, x_i](x - x_{i-1})$$

We want to check at x_{i-1}, we will have $\begin{cases} s_i(x_{i-1}) = f(x_{i-1}) \\ s_i(x_i) = f(x_i) \end{cases}$

5.6.1 Error

Suppose $\{(x_i, y_i)\}$ are obtained from some $f(x)$ and let $s(x)$ be the piecewise linear interpolant on $[a, b]$.
We know that $[x_{i-1}, x_i]$

$$|f(x) - s_i()| = \left|\frac{f^{(2)(c_i)}}{2!}\right| |(x-x_i)(x-x_{i-1})|$$

Note that

$$\frac{f^{(2)(c_i)}}{2!} \le \frac{||f''||_{L^\infty[a,b]}}{2}$$

Set $\phi(x) = (x-x_i)(x-x_{i-1})$.
Note:

$$\begin{aligned} |\phi'(x)| &= 0 \\ \implies (x-x_i) + (x-x_{i-1}) &= 0 \\ \implies x &= \frac{x_i + x_{i-1}}{2} \end{aligned}$$

so max of $|\phi|$ is $\left|\phi\left(\frac{x_i+x_{i-1}}{2}\right)\right| = \left|\left(\frac{x_{i-1}-x_i}{2}\right)\left(\frac{x_i-x_{i-1}}{2}\right)\right|$

So Let's set $h := \max_i(x_i - x_{i-1})$, so $|f(x) - Si(x)| \le \frac{||f''||_\infty}{2}\frac{h^2}{4}$, this implies that

$$||f-s||_{L^\infty[a,b]} \le \frac{h^2}{8}||f'||_{L^\infty}[a,b]]$$

5.6.2 Cubic Spline

Definition 6.0.0.0.1
*A **spline** is a piecewise $c^k[a,b]$ function that is globally in $c^{k-1}[a,b]$*

$$\begin{aligned} s_1(x) &= y_1 + b_1(x-x_1) + c_1(x-x_1)^2 + d_1(x-x_1)^3 & x_1 \le x \le x_2 \\ &\dots \\ s_{n-1}(x) &= y_{n-1} + b_{n-1}(x-x_{n-1}) + c_{n-1}(x-x_{n-1})^2 + d_{n-1}(x-x_{n-1}) & x_{n-1} \le x \le x_n \end{aligned}$$

unknowns are $\{(b_i, c_i, d_i)\}_{i=1}^{n-1}$, so there are $3n-3$ unknowns.

1. Enforce continuity at interior points, $s_i(x_i) = s_{i+1}(x_i)$ for $1 \le i \le n-1$. Get $n-1$ equations
2. Enforce $s_i'(x_i) = s_{i+1}'(x_i), 2 \le i \le n-2$, $n-2$ equations
3. Enforce $s_i''(x_i) = s_{i+1}''(x_i), 2 \le i \le n-1$, $n-2$ equations

So there are total of $3n-5$ equations

6 Linear Least Squares

Example: $\{x_i, y_i\}_{i=1}^n$ where $n >> 1$, we want to fit a line $y = mx + c$, where m, c appear linearly. And we force the data to fit model

$$\begin{cases} mx_c = y_1 \\ \vdots mx_n + c = y \end{cases}$$

We get

$$\begin{pmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{pmatrix} \begin{pmatrix} m \\ c \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$
$$\implies A \begin{pmatrix} m \\ c \end{pmatrix} = \vec{b}$$

We want to find $\vec{x} \in R^n$ such that $A\vec{x} - b$ is minimized in the $2 - norm$, i.e

$$\vec{x} = \text{argmin}_{x \in \mathbb{R}^n} ||Ax - b||_2$$

where $||Ax - b||_2$ is the residual error.

Calc III:
Set $\delta_x ||Ax0b||_2 = \vec{0}$ and solve for x. We get

$$A^T A\vec{x} = A^T b$$

the normal equation, which can be solved for $\vec{X}$

Derivation

Derivation of $A^T A\vec{x} = A^T b$.
We force $b - A\bar{x} \perp Ax, \forall x \in \mathbb{R}^n$ in order to minimize the residual error. Also note that $||x||_2 =< x, x >$, the dot inner product and $< x, y >= x^T y$ And because $b - A\bar{x} \perp Ax$

$$\begin{aligned} < b - A\bar{x}, Ax > &= 0 \\ (b - A\bar{x})^T Ax &= 0 \\ (b^T - \bar{x}^T A^T)Ax &= 0 \\ (b^T A - \bar{x}^T A^T A)x &= 0 \\ X^T(Ab^T - A^T A\bar{x}) &= 0 \end{aligned}$$

for all $x \in \mathbb{R}^n$ By lemma below, we get

$$A^T b - A^T A\vec{x} = 0$$

Lemma 1.0.1
If $< x, y >= 0$, for all $\vec{x} \in R^n$, then $\vec{y} = \vec{0}$

Proof. Let $x = y$, then $< y, y >= 0 \implies y = 0$ □

Theorem 1.1
If columns of A are linearly independent, then $A^T A$ is invertible

Example: $A = \begin{pmatrix} \epsilon & 0 \\ 0 & \epsilon \\ 1 & 1 \end{pmatrix}$, where $0 < \epsilon < \sqrt{\epsilon_{mach}} \approx 10^{-8}$, then

$$\begin{aligned} A^T A &= \begin{pmatrix} \epsilon & 0 & 1 \\ 0 & \epsilon & 1 \end{pmatrix} \begin{pmatrix} \epsilon & 0 \\ 0 & \epsilon \\ 1 & 1 \end{pmatrix} \\ &= \begin{pmatrix} 1+\epsilon^2 & 1 \\ 1 & 1+\epsilon^2 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \end{aligned}$$

So $A^T Ax = A^T b$ is a singular system numerically

QR Decomposition

Let $A \in \mathbb{R}^{m\times n}, m \geq n$ so it is overdetermined system. Then, we can decompose A in the following

$$A = QR$$

where Q is orthogonal $Q \in \mathbb{R}^{m\times n}$ and $R = \left(\begin{array}{ccc} r_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ O & \cdots & r_{nn} \\ \hline O & \ddots & O \\ & \ddots & \vdots \end{array}\right)$ where the upper part is size $n \times n$ and the bottom part is size $(m-n) \times n$.

Definition 2.0.0.0.1
Orthogonal matrix means

1. $Q^{-1} = Q^T$
2. *columns of Q are orthogonormal and Q is square matrix*

Idea: Qr encodes the gram schmidt process

Matlab Code: $[Q, R] = qr(A)$

We can also write as

$$A = QR = [\hat{Q} \quad \overline{Q}] \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix} = \hat{Q}\hat{R} \text{ Reduced QR}$$

where $\hat{R} = \begin{pmatrix} r_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ O & \cdots & r_{nn} \end{pmatrix} \in \mathbb{R}^{n\times n}$ and $\hat{Q} \in \mathbb{R}^{m\times n}$, this potentially not square but still has orthonormal columns.

Matlab Code: $[\hat{Q}, \hat{R}] = qr(A, 0)$ for reduced QR

Theorem 2.1
Orthogonal matrices Q are isometries. i.e

$$||Qx||_2 = ||x||_2$$

Corollary 2.1.1

$$||Q^T x||_2 = ||x||_2$$

Least-Sqaures

Back to the least=sqaures,

$$Ax = b$$

Note that

$$\overline{x} = \text{argmin}_{x\in\mathbb{R}^n} ||Ax - b||_2$$

Suppose we have $A = QR$, so

$$\begin{aligned}&||Ax - b||_2\\ =&||QRx - b||_2\\ =&||Q^T(QRx - b)||_2\\ =&||Rx - Q^T b||_2\end{aligned}$$

which is the min.
Example: Let $A \in \mathbb{R}^{4\times 2}, b \in \mathbb{R}^4$ then

$$||Rx - Q^T b||_2 = \left\|\begin{pmatrix} r_{11} & r_{12}\\ 0 & r_{22}\\ 0 & 0\\ 0 & 0\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix} - \begin{pmatrix}c_1\\ c_2\\ c_3\\ c_4\end{pmatrix}\right\|_2$$

Then, choose $x = \begin{pmatrix}x_1\\ x_2\end{pmatrix}$ such that

$$\begin{pmatrix} r_{11} & r_{12}\\ 0 & r_{22}\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix} = \begin{pmatrix}c_1\\ c_2\end{pmatrix}$$

or $\hat{R}\vec{x} = \vec{c}_{1:2}$. Then we end up with $||Ax - b||_2 = \left\|\begin{pmatrix}0\\ 0\\ c_3\\ c_4\end{pmatrix}\right\| = \sqrt{c_3^2 + c_4^2}$ as the minimal value

6.3.1 Method

To solve $A\vec{x} = \vec{b}$ in the least square sense, form $A = QR = \hat{Q}\hat{R}$ and let $\vec{c} = Q^T\vec{b}$, and solve $\hat{R}\vec{x} = \vec{c}_{1:n}$ for $\bar{x}$, which is the least square solution

Approximation Theory

Goal: Approximate f by some simpler function in the sense of least squares

Definition 4.0.0.0.1
*Let V be a vector space. A **norm** $||\cdot|| : V \to \mathbb{R}$ satisifies:*

1. $||f|| \geq 0$, *and* $||f|| = 0$ *if and only if* $f = 0$
2. $||\alpha f|| = |\alpha|||f||$ *for all* $\alpha \in \mathbb{R}$
3. $||f + g|| \leq ||f|| + ||g||$

Let $f \in C[a, b]$, then the norm choices can be

1. $||f||_{L^2[a,b]} = \left(\int_a^b |f(x)|^2 dx\right)^{\frac{1}{2}}$
2. $||f||_{L^1[a,b]} = \int_a^b |f(x)|dx$
3. $||f||_{L^\infty[a,b]} = \max_{x\in[a,b]}|f(x)|$

Definition 4.0.0.0.2
Distance:

$$||f - g||_{L^2[a,b]}$$

Definition 4.0.0.0.3
Inner Product

$$< f, g >:= \int_a^b f(x)g(x)dx$$

Axioms

1. $< f, g >=< g, f >$
2. $< f + g, h >=< f, h > + < g, h >$
3. $< \alpha f, g >= \alpha < f, g >=< f, \alpha g >, \forall\alpha \in \mathbb{R}$
4. $< f, f >\geq 0$ *and* $< f, f >= 0$ *if and only if* $f = 0$

6.4.1 Problem

Let $f \in V$, and let V_n be a finite dimensional space with basis $\{g_i\}_{i=1}^n$, so

$$V_n = \text{span}\{g_1, \cdots, g_n\}$$

Goal: Find $v \in V_n$ such that the error

$$||f = v||_{L^2[a,b]}$$

is minimal

Let

$$v(x) = \sum_{i=1}^n c_i g_i(x)$$

where c_i is unkown coefficients. Then

$$\begin{aligned}
&||f - v||_{L^2}^2 \\
=&||f||_{L^2}^2 - 2 < f_1, v > + ||v||_{L^2}^2 \\
=&||f||_{L^2}^2 - 2 < f_1, \sum_{i=1}^n c_i g_i > + < \sum_{i=1}^n c_i g_i, \sum_{j=1}^n c_i g_i > \\
=&||f||_{L^2}^2 - 2 \sum_{i=1}^n c_i < f, g_i > + \sum_{i=1}^n \sum_{j=1}^n c_i g_i < g_i, g_j > \\
=&\phi(\vec{c})
\end{aligned}$$

where $\vec{c} = (c_1, \cdots, c_n)$. To minimize, set

$$\Delta_c \phi(\vec{c}) = \vec{0}$$

We will do this componentwise, i.e show $\frac{\partial}{\partial c_k}\phi(\vec{c}) = 0, \forall k = 1, \cdots, n$. Note that

$$\begin{aligned}
\phi(\vec{c}) &= ||f||_{L^2}^2 - 2 \sum_{i=1}^n c_i < f, g_i > + \sum_{i=1}^n \sum_{j=1}^n c_i g_i < g_i, g_j > \\
\frac{\partial}{\partial c_k}\phi(\vec{c}) &= 0 = 2 \sum_{i=1}^n \frac{\partial c_i}{\partial c_k} < f, g_i > + \sum_{i=1}^n \sum_{j=1}^n \frac{\partial}{\partial c_k}(c_i c_j) < g_i, g_j >
\end{aligned}$$

Note that $\frac{\partial c_i}{\partial c_k} = \delta_{ki} = \begin{cases} 1 & i = k \\ 0 & i \neq k \end{cases}$ and $\frac{\partial}{\partial c_k}(c_i c_j) = c_j \delta_{kj} + c_j \delta_{ki}$.
Therefore,

$$\begin{aligned}
0 &= -2 < f, g_k > + \sum_{i=1}^n c_i < g_i, g_k > + \sum_{j=1}^n c_j < g_k, g_j > \\
&= -2 < f, g_k > +2 \sum_{i=1}^n c_i < g_i, g_k >
\end{aligned}$$

Finally,

$$\sum_{i=1}^n c_i < g_i, g_k > =< f, g_h >$$

This is a linear system for unknowns $\vec{c} = \begin{pmatrix} c_1 \\ \vdots \\ v_n \end{pmatrix}$, $A\vec{c} = \vec{b}$, where $A_{ik} =< g_i, g_k >$ and $b_k =< f, g_k >$.
If we have an orthonormal basis of V_n, say $\{g_1, \cdots, g_n\}$ with $< g_i, g_j >= 0$ if $i \neq j$ and $< g_i, g_i >= 1$ Then, $< g_i, g_k >= \delta_{ij}$, and so

$$c_k =< f, g_k >$$

and we get

$$v = \sum_{k=1}^n < f, g_k > g_k = \text{proj}_{v_n} f$$

as our least squares solution. Note that

$$\text{proj}_{\vec{x}}\vec{y} = \frac{< \vec{y}, \vec{x} >}{||\vec{x}||}\vec{x}$$

Weighted Least-Squares

$$||f||_w = ||f||_{L^2_w[a,b]} = \left(\int_a^b |f(x)|^2 w(x)dx\right)^{\frac{1}{2}}$$

and

$$\langle f, g\rangle = \int_a^b f(x)g(x)w(x)dx$$

To find $v \in V_n$, the finite dimensional space best approximating f in the $L^2 + w$-sense, i.e

$$v = \text{argmin}_{u\in V_n}||f-u||_{L^2_w}$$

Set $V(x) = \sum_{i=1}^n \langle f, g_i\rangle_{L^2_w} g_i(x) = \text{proj}_{V_n} f$, where $\{g_i\}_{i=1}^n$ is a basis of V_n.

Here are some **assumptions** on w **weighted function:**

1. $w(x) \geq 0$
2. $w(x) = 0$ at atmost a finite number of x

Example: $w(x) = \frac{1}{\sqrt{1-x^2}}$ on $(-1, 1)$ (chebyshev weight)

Note that

$$\min_{u\in V_n} \int_{-1}^1 |f(x) - u(x)|^2 \frac{1}{\sqrt{1-x^2}} dx$$

The weight ensures that our least-sqaures solution approximation f much more closely near end-points -1 and 1. It is to choose a weight that is large on $[a, b]$ (where good datas lie) and small on $[b, c]$ (where bad data lie)

Example: Let $w(x) = x$ on $[0, 1]$, approximate e^x on $[0, 1]$ in L^2_w-sense with $p \in P_1$.
We need ONB of P_1 on $[0, 1]$ with $w(x) = x$. Give a basis $\{1, x\}$, note that $\langle 1, x\rangle = \int_0^1 1\cdot x\cdot xdx \neq 0$.
Therefore, we need Gram-Schmiat.

$$\begin{aligned} v_1 &= 1 \\ v_2 &= x - \frac{\langle x, 1\rangle}{||1||^2}\cdot 1 \ (\text{proj}_1 x) = x\cdot \frac{\int_0^1 x\cdot 1\cdot xdx}{\int_0^1 xdx}\cdot 1 \\ &= x - \frac{\frac{1}{3}}{\frac{1}{2}} \\ &= x - \frac{2}{3} \end{aligned}$$

Then, we need normalized.

$$e_1 = \frac{v_1}{||v_1||_w} = \frac{1}{\left(\int_0^1 1^2\cdot xdx\right)^{\frac{1}{2}}} = \sqrt{2}$$

$$e_2 = \frac{v_1}{||v_1||_2} = \frac{x-\frac{2}{3}}{\left(\int_0^1 \left(x-\frac{2}{3}\right)\right)^2 xdx}^{\frac{1}{2}} = K(x - \frac{2}{3})$$

Just to check orthogonality,

$$\begin{aligned} \langle x - \frac{2}{3}, 1\rangle &= \int_0^1 \left(x - \frac{2}{3}\right) xdx \\ &= \frac{1}{3} - \frac{2}{3}\frac{1}{2} = 0 \end{aligned}$$

Then solution is

$$v = \langle e^x, \sqrt{2}\rangle_w \sqrt{2} + \langle e^x, K\left(x - \frac{2}{3}\right)\rangle K\left(x - \frac{2}{3}\right)$$

Projection formula

$$\text{proj}_{\vec{x}}\vec{y} = \frac{\langle \vec{y}, \vec{x}\rangle}{||\vec{x}||^2}\vec{x}$$

The orthogonal is

$$\langle x, y - \frac{\langle y, x\rangle}{||x||^2}x\rangle$$

6.5.1 Gram-Schmidt

Suppose $\{w_1, w_2, w_3\}$ is a basis for vector space V. To orthogonolize, use new basis $\{v_1, v_2, v_3\}$, where

$$\begin{aligned} v_1 &= w_1 \\ v_2 &= w_2 - \text{proj}_{v_1} w_2 \\ v_3 &= v_3 - \text{proj}_{v_1} w_3 - \text{proj}_{v_2} w_3 \end{aligned}$$

6.5.2 Legendre Polynomials

$w(x) = 1$ on $[-1, 1]$
$P_0(x) = 1$, $P_1(x) = x$. Then

$$(n+1)P_{n+1}(x) - (2n+1)xP_n(x) + nP_{n-1}(x) = 0 n \geq 1$$

when $n = 1$, $2P_2(x) - 3x \cdot x + 1 \cdot 1 = 0$, so $P_2(x) = \frac{3x^2-1}{2}$.

To get Legendre $p(x)$ on $[a, b]$, use a linear change of variable. So on $[a, b]$, with $w(x) = 1$, an orthogonal basis of P_1 is $\{1, 2t - 1\}$

7 | Numerical Differentiation

Note that slope $f'(x) \approx \frac{f(x_2)-f(x_1)}{x_2-x_1}$, and

$$f'(x) = \lim_{h\to 0} \frac{f(x+h)-f(x)}{h}$$

Two Point Forward Difference Quotient

given $x, x+h$, then

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

Error

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

The error term. And error estimate will be

$$||f'(x)-D_h f(x)||_{L^\infty[a,b]} \le \frac{h}{2}||f''||_{L^\infty[a,b]}$$

So Theoretically, as $h \to 0$, we get $D_h f \to f'$

Two Point Backward Difference Quotient

given $x, x+h$, then

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

Have

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

So for backward error, we still get

$$||f'(x)-\frac{f(x)-f(x-h)}{h}||_{L^\infty[a,b]} \le \frac{h}{2}||f''||_{L^\infty[a,b]}$$

Centered Differencing

Given $x-h, x, x+h$. Note that using taylor,

$$\begin{aligned}f(x+h) &= f(x)+hf'(x)+\frac{h^2}{2}f''(x)+\frac{h^3}{3!}f'''(\xi_1)\\ f(x-h) &= f(x)-hf'(x)+\frac{h^2}{2}f''(x)-\frac{h^3}{3!}f'''(\xi_2)\end{aligned}$$

Subtracting them, we have

$$f(x+h)-f(x-h) = 2hf'(x)+\frac{h^3}{6}(f'''(\xi_1)+f'''(\xi_2))$$

So

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + \frac{h^2}{12}[f'''(\xi_1) + f'''(\xi_2)]$$

$$f'''(\xi_1) + f'''(\xi_2) = 2f'''(\xi) \text{ by Intermediate value theorem with } \xi \text{ between } \xi_1 \text{ and } \xi_2$$

Therefore

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + \frac{h^2}{6}[2f'''(\xi)]$$

So

$$||f'(x) - \frac{f(x+h) - f(x-h)}{2h}|| \leq \frac{h^2}{6}||f'''||_{\text{inf}} = O(h^2)$$

Theorem 3.1
Intermediate Value Theorem
Let $f \in C[a,b]$ and $a_i > 0$ or $(a_i < 0)$ for all i. Then, $\exists d \in [a,b]$ such that

$$\sum_{i=1}^{n} n a_i f(x_i) = f(d) \sum_{i=1}^{n} a_i$$

Method of Undetermined Coefficient

Supposed we want $f'(x)$ approximation using $f(x), f(x+h)$. Find best possible formula

$$f'(x) = Af(x) + Bf(x+h) \qquad = Af(x) + B[f(x) + hf'(x) + \frac{h^2}{2}f''(\xi)]$$

Let's set

$$\begin{aligned} A + B &= 0 \\ Bh &= 1 \end{aligned}$$

This implies $B = \frac{1}{h}, A = -\frac{1}{h}$. So we get

$$\begin{aligned} f'(x) &= -\frac{1}{h}f(x) + \frac{1}{h}f(x+h) + \frac{h}{2}f'''(\xi) \\ &= \frac{f(x+h) - f(x)}{h} + \frac{h}{2}f''(\xi) \end{aligned}$$

Error

Recall if

$$D_h f(x) := \frac{f(x+h) - f(x)}{h}$$

then

$$||f' - D_h f||_{L^\infty} \leq \frac{h}{2}||f''||_{L^\infty}$$

In finite preicsion

$$\widetilde{f}(x) = f(x) + e(x)$$

Since D_h is a linear operation, then

$$D_h \widetilde{f}(x) = D_h f(x) - G_h e(x)$$

And note that

$$D_h e(x) = \frac{e(x+h) - e(x)}{h}$$

And

$$\begin{aligned} |D_h e(x)| &= \left| \frac{e(x+h) - e(x)}{h} \right| \\ &\leq \frac{|e(x+h)| + |e(x)|}{h} \\ &\leq \frac{2\epsilon}{h} \end{aligned}$$

where $\epsilon \approx \epsilon_{mach}$.
So

$$||D_h \widetilde{f} - D_h f||_{L^\infty} \leq \frac{2\epsilon}{h}$$

Then,

$$\begin{aligned}
&||f' - D_h \widetilde{f}||_{L^\infty} \\
=&||f' - D_h f + D_h f - D_h \widetilde{f}||_{L^\infty} \\
\leq&||f' - D_h f||_{L^\infty} + ||D_h f - D_h \widetilde{f}||_{L^\infty} \\
\leq&\frac{h}{2}||f''||_{L^\infty} + \frac{2\epsilon}{h}
\end{aligned}$$

Set $m = ||f''||_{L^\infty}$.
To minimize error bound, set

$$\begin{aligned}
\frac{d}{dh}\left(\frac{h}{2}m + \frac{2\epsilon}{h}\right) &= 0 \\
\frac{m}{2} - \frac{2\epsilon}{h^2} &= 0 \\
h &= \sqrt{\frac{4\epsilon}{m}} \approx 10^{-8}
\end{aligned}$$

Richordson Extrapolation

Recall

$$\begin{aligned}
f'(x) &= \frac{f(x+h) - f(x-h)}{2h} - \frac{h^2}{6}f'''(\xi) \\
Q &= F_2(h) + O(h^2)
\end{aligned}$$

where 2 for $F_2(h)$ is the error order.
The extrapolation formula is

$$Q \approx \frac{2^n F_n\left(\frac{h}{2}\right) - F_n(h)}{2^n - 1}$$

Here, $F_n(h)$has $O(h^n)$ error, and approximates Q.
And $F_{n+1}(h)$ is an (at least) $O(h^{+1})$ method for approx Q, i.e $F_{n+1}(h) + O(h^{n+1})$
For centered differences, get

$$f'(x) = \frac{2^2\left(\frac{f(x+\frac{h}{2})-f(x-\frac{h}{2})}{2(\frac{h}{2})}\right) - \frac{f(x+h)-f(x-h)}{2h}}{2^2 - 1} + O(h^3)$$

using $n = 2$,

$$= \frac{1}{6h}\left(f(x-h) - 9f\left(x - \frac{h}{2}\right) + 8f\left(x + \frac{h}{2}\right) - f(x+h)\right)$$

Note that Error is $O(n^4)$. This since the above equation as unchanged when replacing h with $-h$, so error can only depend on even powers of h

Proof. Supposed

$$\begin{aligned}
Q &= F_n(h) + Kh^n + O(h^m) \\
2^n Q &= 2^n F_n\left(\frac{h}{2}\right) + K\left(\frac{h}{2}\right)^n + O(h^m)
\end{aligned}$$

subtract to get

$$\begin{aligned}
(2^n - 1)Q &= 2^n F_n\left(\frac{h}{2}\right) + F_n(h) + O(h^m) \\
Q &= \frac{2^n F_n\left(\frac{h}{2}\right) - F_n(h)}{2^n - 1} + O(h^m)
\end{aligned}$$

□

8 Quadrature

Quadrature numerical integration. The common way is

$$\int_a^v f(x)dx \approx \sum_{i=0}^{n-1} f(x_i^*)(x_{i+1} - x_i)$$
$$\int_a^v f(x)dx = \lim_{n\to\infty} \sum_{i=0}^{n} f(x_i^*)\delta x_i$$

Note that

$$\int_a^b f(x)dx = \sum_{i=1}^{n} f(x_i)w_i + \text{ Error}$$

where x_i is nodes and w_i is weights.

Newton-Cotes Quadrature

Uses

$$\int_a^v f \approx \int_a^b p$$

, where P is an interpolating polynomial at equally spaced nodes.

closed rules includes endpoints values $f(a), f(b)$. open rule is do not use the endpoints values $f(a), f(b)$

8.1.1 Midpoint Rule

use the mid point $\frac{a+b}{2}$. By Taylor,

$$\begin{aligned}
f(x) &= f(m) + (x-m)f'(m) + \frac{(x-m)^2}{2} f''(\xi_x) \\
\int_a^b f(x) &= \int_a^b f(m) + \int_a^b (x-m)f'(m) + \int_a^b \frac{(x-m)^2}{2} f''(\xi_x) \\
&= f(m)(b-a) + 0 + \frac{f''(\xi)}{2} \int_a^b (x-m)^2 dx \\
&= f(m)(b-a) + 0 + \frac{1}{2} f''(\xi) \frac{(b-a)^3}{12}
\end{aligned}$$

we used Intgeral Mean Value Theorem.

Theorem 1.1
IF $f \in C[a,b]$ and $g \geq 0$ or $g \leq 0$ on $[a,b]$, then $\exists \xi \in (a,b)$ such that

$$\int_a^b fg = f(\xi) \int_a^b g$$

So Mid point rule tells us

$$\int_a^b = f(m)(b-a) + \frac{1}{24} f''(\xi)(b-a)^3$$

or

$$\left| \int_a^b f dx - f(m)(b-a) \right| \leq \frac{(b-a)^3}{24} ||f''||_{L^\infty[a,b]}$$

Trapezoid Rule

Recall langurage remainder interpolate is

$$f(x) = f(a)\frac{x-b}{a-b} + f(b)\frac{x-a}{b-a} + \frac{f''(\xi)}{2!}(x-a)(x-b)$$

where $\xi \in (a,b)$. We do integrate, we have

$$\begin{aligned}\int_a^b f &= \int_a^b f(a)\frac{x-b}{a-b} + f(b)\frac{x-a}{b-a} + \int_a^b \frac{f''(\xi)}{2!}(x-a)(x-b) \\ &= \frac{1}{2}(b-a)(f(a)+f(b)) + \frac{1}{2}f''(\xi)\int_a^b (x-a)(x-b)dx \\ &= \frac{1}{2}(b-a)(f(a)+f(b)) + \frac{1}{2}f''(\xi) - \frac{1}{6}(b-a)^3\end{aligned}$$

So Trapezoid rule give us

$$\int_a^b = \frac{b-a}{2}(f(a)+f(b)) - \frac{1}{12}(b-a)^3 f''(\xi)$$

Simpson's Rule

Given $a, b, m = \frac{a+b}{2}$
Then,

$$\int_a^b = \frac{b-a}{6}[f(a)+f(b)+4f(m)] - \frac{(b-a)^5}{90(32)}f^5(\xi)$$

Note: Simpsons is exact for $p \in P_e$. So

$$\left|\int_a^b f - \sum w_i f(x_i)\right| \le \frac{|b-a|^5}{90(32)}||f^4||_\infty$$

we have Given $2n+1$ nodes, we have $2n$ subintervals, and $h = \frac{b-a}{2n}$ = cosnstant. Then Simpsons' Rule on $[x_{2i}, x_{2i+1}]$ is

$$\int_{x_{2i}}^{x_{2i+2}} \approx \frac{2h}{6}[f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2}) - \frac{(2h)^5}{90\cdot 32}f^{(4)}(\xi_i)$$

where ξ_i is between x_{2i} and x_{2i+2}
So,

$$\begin{aligned}\int_a^b f &= \sum_{i=0}^{n-1}\int_{x_{2i}}^{x_{2i+2}} f \\ &= \frac{h}{3}\sum_{i=0}^{n-1}[f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})] - \frac{h^5}{90}\sum_{i=0}^{n-1} f^4(\xi_i)\end{aligned}$$

where $\sum_{i=0}^{n-1} f^4(\xi_i) = nf^4(\xi)$ by IVT for some $\xi \in (a,b)$. Therefore

$$= \frac{h}{3}\sum_{i=0}^{n-1}[f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})] - \frac{h^5 n}{90}f^4(\xi)$$

use $hn = \frac{b-a}{2}$, we have

$$= \frac{h}{3}\sum_{i=0}^{n-1}[f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})] - \frac{b-a}{180}h^4 f^4(\xi)$$

Therefore, erros is $-\frac{b-a}{180}h^4 f^4(\xi)$.
And also

$$x_i = a + ih = a + i\frac{b-a}{2n}$$

for $0 \le i \le 2n$

(Weighted) Gaussian Quadrature

$$\int_a^b f(x)w(x)dx \approx \sum_{i=1}^n A_i f(x_i)$$

where A_i is quadrature weights and x_i is nodes. Very general many options:

$$\int_{-1}^1 e^x \cdot 1$$
$$\int xe^x \frac{1}{x}$$
$$\int 1 \cdot e^x$$

Orthogonal Polynomial
Recall $\{P_i\} \subseteq P_n$ are orthogonal on $[a, b]$ with $w(x)$ if

$$< p_i, p_j >_w = \int_a^b p_i(x)p_j(x)w(x)dx$$

Theorem 4.1
If $\{p_i\}_{i=0}^n \subseteq P_n$ is orthogonal and $deg(p_i) = i$, then p_i has exactly i distinct zeroes

Example: $\{1, x, \frac{3}{2}, \frac{3}{2}(x^2 - \frac{1}{3})\}$ and the first legrendre polynomial with $w(x) = 1$ on $[-1, 1]$.

Guass-Legrende quadrature
Fix n. Let $\{x_i\}_{i=1}^n$ be the n zeros of the degree n legrendre polynomial $P_n(x)$.
Then,

$$\int_{-1}^1 f(x) \cdot 1dx \approx \sum_{i=1}^n A_i \cdot f(x_i)$$

with $\{A_i\}$ given by undetermined coefficients and the rule is exact for any $f \in P_{2n-1}$

Example: $\int_{-1}^1 f(x) \cdot 1dx$
Gauss-Legrende with two nodes. Degree legrende polynomial is

$$\frac{3}{2}(x^2 - \frac{1}{2})$$

with zeros $x = \pm\sqrt{\frac{1}{3}}$, so

$$\int_{-1}^1 f(x)dx \approx A_1 f\left(-\frac{1}{\sqrt{3}}\right) + A_2 f\left(\frac{1}{\sqrt{3}}\right)$$

It will be exact for $f \in P_3$

Force rule to be exact for $f = 1, x$, we have

$$\begin{aligned} \implies 2 &= \int_{-1}^1 1 = A_1 + A_2 \\ 0 &= \int_{-1}^1 x = A_1 f\left(-\frac{1}{\sqrt{3}}\right) + A_2 f\left(\frac{1}{\sqrt{3}}\right) \\ \implies A_1 &= A_2 = 1 \end{aligned}$$

For x^2, check

$$\int_{-1}^1 x^2 = \frac{2}{3} = f\left(-\frac{1}{\sqrt{3}}\right)^2 + f\left(\frac{1}{\sqrt{3}}\right)^2$$

Explicit quadrature weight formula:

$$\int_a^b f(x)w(x)dx \approx \sum_{i=1}^n A_i f(x_i)$$

Let $P_{n-1}(x) = \sum_{i=1}^{n} f(x_i)l_i(x)$, where L_i is the lagrange basis, be the interpolant through

$$\{(x_i, f(x_i))\}_{i=1}^{n}$$

So

$$\begin{aligned}
&\int_a^b f(x)w(x)dx \\
\approx &\int_a^b P_{n-1}w(x)dx \\
= &\sum_{i=1}^{n} f(x_i)\left(\int_a^b l_i(x)w(x)dx\right)
\end{aligned}$$

And $\left(\int_a^b l_i(x)w(x)dx\right)$ is weights A_i, i.e $A+i = \int_a^b l_i(x)w(x)dx$

Composite Faussian Rules

$$\begin{aligned}
&\int_a^b f(x)w(x)dx \\
= &\sum_{i=1}^{n}\int_{x_i}^{x_{i+1}} f(x)w(x)dx
\end{aligned}$$

where

$$\int_{x_i}^{x_{i+1}} f(x)w(x)dx = \frac{x_{i+1}-x_i}{2}\int_{-1}^{1} f(\phi(t))w(\phi(t))dt$$

where $\phi(t) = \frac{x_i+x_{i+1}}{2} + \frac{x_{i+1}-x_i}{2}t$ and can approximate the $\int_{-1}^{1}$ integral with Gaussian rule.

Proof. Let $p \in P_{2n-1}$. Let $\{x_i\}_{i=1}^{n}$ be ther zeros of the degree n orthogonal $p_n(x)$ on $[a,b]$ with $w(x)$. By long division

$$p(x) = s(x)p_n(x) + r(x)$$

where $deg(s), deg(r) \leq n-1$. Also note $p(x_i) = r(x_i)$
then, integrate,

$$\int_a^b p(x)w(x)dx = \int_a^b s(x)p_n(x) + \int_a^b r(x)w(x)dx$$

Note that $\int_a^b s(x)p_n(x) = 0$ since $s(x) = \sum_{i=0}^{n-1} c_i p_i(x)$ where $p_i(x)$ is our orthogonal basis, and then use $< p_i, P_n >_w = 0$
And note that $r(x) = \sum_{i=1}^{n} r(x_i)l_i(x)$ where $l_i(x)$ is the lagrange basis.
Therefore, we have

$$\int_a^b p(x)w(x)dx = \sum_{i=1}^{n} p(x_i)\left(\int_a^b l_i(x)w(x)dx\right)$$

$\square$

9 Rowberg Intergration

Subdividing partition until error tolerance met on each subinterval.
What we do is given a, b we take the mid point $\frac{1}{a+b}$, and then take $\frac{a+\frac{a+b}{2}}{2}$ again, until we are satisified.

Fact:

$$
\begin{aligned}
I(f) = \int_a^b (f) &= \frac{h}{2}\left(f(a) + f(b) + 2\sum_{i=1}^{n} f(x_i)\right) + \sum_{i=1}^{n} c_i h^{2i} \\
&= T(h) + c_1 h^2 + c_2 h^4 + c_3 h^6 + \cdots
\end{aligned}
$$

Each Richardson extrapolation gains a power of 2 with h, and

$$
c_i \sum f^{2i}(x_0)
$$

And the subinterval width will be

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

And we have

$$
\begin{aligned}
R_{1,1} := T(h_1) &= \frac{h_1}{2}[f(a) + f(b)] \\
R_{2,1} := T(h_2) &= \frac{h_2}{2}[f(a) + f(b) + 2f(a + h_2)] \\
&= \frac{1}{2} R_{1,1} + h_2 f(a + h_2) \\
\vdots &= \vdots \\
R_{j,i} := T(h_j) &= \frac{1}{2} R_{j-1,1} + h_j \sum_{i=1}^{2^{j-2}} f(a + (2u-1)h_j)
\end{aligned}
$$

for $j \geq 2$. Also,

$$
\begin{aligned}
R_{2,2} &= \frac{2^2 R_{2,1} - R_{1,1}}{2^2 - 1} \\
&= \frac{2^2 T\left(\frac{h_1}{2}\right) - T(h_1)}{2^2 - 1}
\end{aligned}
$$

Then the Table of Extrapolations is

$R_{1,1}$					
$R_{2,1}$	$R_{2,2}$				
$R_{3,1}$	$R_{3,2}$	$R_{3,3}$			
$\vdots$					
$R_{j,1}$					

The first column has composite trapeziod rules with error $O(h^2)$
Second column has errors $O(h^4)$, and for third column error is $O(h^6)$

Also,

$$\begin{aligned}
R_{2,2} &= \frac{2^2 R_{2,1} - R_{1,1}}{2^2 - 1} \\
&= \frac{2^2 T\left(\frac{h_1}{2}\right) - T(h_1)}{2^2 - 1} \\
R_{3,2} &= \frac{2^2 R_{3,1} - R_{2,1}}{2^2 - 1} \\
R_{3,3} &= \frac{2^4 R_{3,2} - R_{2,2}}{2^4 - 1}
\end{aligned}$$

Here, $R_{j,i}$, j represents $T(h_j)$ is using subinterval width h_j, and k will represent number of extrapolating.

In general,

$$R_{j,k} = \frac{1}{2^{2k-2}} \left(2^{2k-2} R_{j,k-1} - R_{j-1,k-1}\right)$$

Stopping Criteria:

$$|R_{j,j} - R_{j+1,j+1}| < TOL$$

Then $R_{j+1,j+1}$ is the best approximation of $\int_a^b f$

10 Unknown

Backward Euler Method

$$\begin{cases} y = f(t, y(t)) & t \in [0, T] \\ y(0) = y_0 \end{cases}$$

We approximate

$$y'(t) \approx \frac{y(t) - y(t-h)}{h}$$

where $h = t_{i+1} - t_i$ is constant. Therefore,

$$\begin{aligned} & y(t) \approx y(t-h) + hf(t, y(t)) \\ \implies & y(t_i) \approx y(t_{i-1}) + hf(t_i, y(t_i)) \end{aligned}$$

Let $w_i \approx y(t_i)$, discrete approximating solution values. Then,

$$w_i = w_{i-1} + hf(t_i, w_i)$$

Then it give backward euler method as following

$$\begin{aligned} w_{i+1} &= w_i + hf(t_{i+1}, w_{i+1}) \\ w_0 &= y_0 \end{aligned}$$

This is an implicit method, since we need to solve for w_{i+1}, set $z = w_{i+1}$, then we need to find root of

$$F(z) = z - [w_i + hf(t_{i+1}, z)]$$

using newton, bisection, fzero

10.1.1 Local Truncation Error

Given

$$w_{i+1} - [w_i + hf(t_{i+1}, w_{i+1})]$$

Replace, we have

$$\begin{aligned} & y(t_i + h) - [y(t_i) + hy'(t_i + h)] \\ =& y(t_i) + hy'(t_i) + \frac{h^2}{2} y''(\xi_i) - y(t_i) - h[y'(t_i) + hy''(\eta_i)] \\ =& h^2 [\frac{1}{2} y''(\xi_i) - y''(\eta_i)] \\ =& O(h^2) \end{aligned}$$

Since local error is $O(h^2)$, global error is $O(h)$. Why local $O(h^p)$ becomes $O(h^{p-1})$ globally?
Set

$$h = \frac{T - t_0}{N}$$

where $[t_0, T]$ ODE time interval and N = number of time steps. Therefore, global error is apprixmately $N \cdot O(h^p)$, which is $\frac{T-t_0}{h} O(h^p) = O(h^{p-1})$

Stability?

Test equation $\begin{cases} y' = \lambda y \\ y(0) = 1 \end{cases}$, where $\lambda < 0$, Solution is $y(t) = e^{\lambda t} \to 0$ as $t \to \infty$.
Apply Backward Euler, we have

$$\begin{aligned} w_{i+1} &= w_i + hf(t_{i+1}, w_{i+1}) \\ &= w_i + h(\lambda w_{i+1}) \\ \implies w_{i+1} &= \left(\frac{1}{1-h\lambda}\right) w_i = \cdots = \left(\frac{1}{1-h\lambda}\right)^i w_0 \end{aligned}$$

As $i \to \infty$, need

$$|\frac{1}{1-h\lambda}| < 1$$

so that $w_i \to 0$. since

$$\lim_{i\to\infty} c^i = 0$$

provided $|c| < i$
Recall $h > 0$ is a positive constant. But this is true for any $h > 0$, so backward euler is A-stable (abosolutely stable)

Trapezoid Method

Given

$$\begin{aligned} y' &= f(t, y(t)) \\ \int_{t_i}^{t_{i+1}} y'(t)dt &= \int_{t_i}^{t_{i+1}} f(t, y(t))dt \\ y(t_{i+1}) - y(y_i) &= \int_{t_i}^{t_{i+1}} f(t, y(t))dt \end{aligned}$$

Note that

$$\int_{t_i}^{t_{i+1}} f(t, y(t))dt \approx \frac{h}{2}\left(f(t_i, y(t_i)) + f(t_{i+1}, y(t_{i+1}))\right)$$

using trapezoid rule. And the method we have is the following

$$\begin{cases} w_{i+1} = w_i + \frac{h}{2}\left(f(t_i, w_i) + f(t_{i+1}, w_{i+1})\right) \\ w_0 = y_0 \end{cases}$$

where $w_{i+1} = w_i + hf(t_i, w_i)$
This is implicit method, and one step. The local error is $O(h^3)$, and this is A stable

Forward Euler Method on Systems

$$\vec{y}(t) = \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix}$$

$\begin{cases} \frac{d}{dt}\vec{y}(t) = \vec{f}(t, \vec{y}(t)) \\ \vec{y}(t_0) = \vec{y_0} \end{cases}$

Let $\vec{w_i} \approx \vec{y}(t_i)$, then $\begin{cases} \vec{w_{i+1}} = \vec{w_i} + h\vec{f}(t_i, \vec{w_i}) \\ \vec{w_0} = \vec{y_0} \end{cases}$.

Or componentwise, for $k = 1, 2$, we have $\begin{cases} \vec{w_{i+1}}^k = \vec{w_i}^k + h\vec{f}^k(t_i, \vec{w_i}) \\ \vec{w_0}^k = \vec{y_0}^k \end{cases}$, where $h = t_{i+1} - t_i$.
Also note that $\vec{w_i}_i^k$, i is the iteration index for time t_i and k

Example: Suppose $\vec{f}(t, \vec{h}) = \begin{pmatrix} t + y^{(n)}(t) \\ y^{(1)}(t)y^2(t) \end{pmatrix}$ and $\vec{y_0} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$.

LEt $h = \frac{1}{2}$, and compute $\vec{w_1} = \vec{y}\left(\frac{1}{2}\right)$ using Forward Euler.
Then we have

$$\begin{aligned}
\vec{w_1} &= \vec{w_0} + \frac{1}{2}\vec{f}(0, \vec{w_0}) \\
&= \begin{bmatrix} 2 \\ 4 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 0 + w_0^1 \\ w_0^{(1)} w_0^2 \end{bmatrix} \\
&= \begin{bmatrix} 2 \\ 4 \end{bmatrix} + \frac{1}{2}\begin{pmatrix} 2 \\ 2(4) \end{pmatrix} \\
&= \begin{bmatrix} 3 \\ 8 \end{bmatrix}
\end{aligned}$$

Therefore, $\vec{y}\left(\frac{1}{2}\right) \approx \begin{bmatrix} 3 \\ 8 \end{bmatrix}$

Leapfrog Method

$$\begin{cases} y' = f(t, y(t)) \\ y(0) = y_0 \end{cases}$$

we are arrpximate

$$y'(t) = \frac{y(t+h) - y(t-h)}{2h} + O(h')$$

using Centered Differences.
We get

$$y(t_i + h) \approx y(t_i - i) + 2hf(t_i . y(t_i))$$

So with $w_i \approx y(t_i)$, we get

$$\begin{cases} w_{i+1} = w_{i-1} + 2hf(t_i . w_i) \\ w_0 = y_0, w_1 = y_0 + hf(t_0, y_0) \end{cases}$$

where w_1 is obtained by other method

Remark: This is a two-step, explicit method with $O(h^3)$ local trunction error. So $O(h^2)$ global error

10.4.1 Stability

$\begin{cases} y \lambda y \\ y(0) = 1 \end{cases}$

With leapfrog, we get $w_{i+1} = w_{i-1} + 2h(\lambda w_i)$. Assume $w_i = r^i$ and find r such that this is a solution, we have

$$\begin{aligned} r^{i+1} &= r^{i-1} + 2h\lambda r^i \\ r &= r^{-1} + 2h\lambda \end{aligned}$$

Solutions are

$$r = r_\pm = -h\lambda \pm \sqrt{1 + (h\lambda)^2}$$

A general solution is

$$w_i = c_1 r_+^i + c_2 c^i$$

for some c_1, c_2. Note that

$$r_\pm = -h\lambda + \sqrt{1 + (h\lambda)^2} > 1$$

for any $h > 0$. So for any $h > 0$, we have

$$w_i \to \infty$$

Therefore Leapfrog is totally unstable

For ODE45, the 4 in the name stands for having an order of 4 ODE solver, with global error $O(h^4)$ to get z_i, approximating values at t_i; And the 5 in the name stands for having an order of 5 solver, with global error $O(h^5)$ to get y_i, approximating value at t_i.

Let $e_i = |z_i - y_i|$, if $e_i <$TOL, then timestep and the value y_i is accepted, and proceed to t_{i+1}.

Definition 4.0.0.0.1
For an informal definition, an ODE is said to be stiff provided that explicit numerical solver require very small h for stability.

Example: The following ODE is an example of stiff ODE:

$$\begin{cases} y' = y^2(1-y) \\ y(0) = 10^{-6} \end{cases}$$

ODE45 solver uses 10^6 timesteps, and ODE15s uses 117 timesteps, while the ODE15s gives better solution.

Runge-Kutta Methods

1-stage RK method is one-step, explicit, first order, and not stable. For deriving the method, suppose we are given an ODE system

$$\begin{cases} y' = f(t, y(t)) \\ y(t_0) = y_0 \end{cases}$$

We write $w_i \approx y(t_i)$. Suppose we can write $w_{i+1} = w_i + as_1$, where $s_1 = hf(t_i, w_i)$ is the state of the method. We are interested in finding a suitable constant value for a such that the method has a smallest local truncation error. Consider we write the following:

$$w_{i+1} - (w_i + ahf(t_i, , w_i)) = y(t_i + h) - \big(y(t_i) + ahy'(t_i)\big)$$

comparing with Taylor expansion $y(t_i) + hy'(t_i) + \frac{h^2}{2}y''(\xi_i)$, and choosing $a = 1$, we obtain the error term $(h^2/2)y''(\xi_i)$, and $w_{i+1} = w_i hf(t_i, w_i)$, which yields the forward Euler's method.

The 2-stage RK method is two-step, explicit, second order, and not stable. Assuming here that we have $w_{i+1} = w_i + as_1 + bs_2$, where $s_1 = hf(t_i, w_i)$, and $s_2 = hf(t_i + \alpha h, w_i + \beta s_i)$. Here we are interested in, using the two stages, finding suitable constants for a, b, α, β such that the method has a smallest local truncation error. Expanding we obtain:

$$\begin{aligned} w_{i+1} - & (w_i + ahf(t_i, w_i) + bhf(t_i + \alpha h, w_i + \beta hf(t_i, w_i))) \\ & = y(t_i + h) - \big(y(t_i) + ahy'(t_i) + bhf(t_i + \alpha h, y(t_i) + \beta hy'(t_i))\big) \end{aligned}$$

Denoting the term $y(t_i + h)$ as (*) and the term $f(t_i + \alpha h, y(t_i) + \beta hy'(t_i))$ as (**), we can write:

$$\begin{aligned} (*) &= y(t_i) + hy'(t_i) + \frac{h^2}{2}y''(t_i) + O(h^3) \\ (**) &= f(t_i, y(t_i)) + \alpha hf_t(t_i, y(t_i)) + \beta hy'(t_i)f_y(t_i, y(t_i)) + O(h^2) \end{aligned}$$

combining to solve for the appropriate α, β, a, b, we obtain a system of equations:

$$\begin{cases} 1 - a - b = 0 \\ 1/2 - \alpha b = 0 \\ 1/2 - b\beta = 0 \end{cases}$$

that is there exists a set of infinitely many solutions for the system, for which a natural choice is $a = b = 1/2$, with $\alpha = \beta = 1$, yielding the explicit Trapezoid method.