Chapter 1 Introduction.

1.1 Examples? Option Pricing

Options : Simple examples are European Options.

Call, Payoff

$$p(T) = \max\{S-K, 0\}$$

American options allow early exercise. adds complexity.

when to exercise,

what value of the option today. $U(S,0)$

Black-sholes Model.

(1) stock price follows a geometric motion.

(2) the risk-free interest rate r is constant.

(3) All risk-free portfolios earn the risk-free rate. No arbitrage.

Gives a PDE ($\S$ 3.1)

$$\frac{\partial V}{\partial t} + \frac{\sigma^2 S^2}{2} + \frac{\partial^2 V}{\partial S^2} + r \cdot S \cdot \frac{\partial V}{\partial S} - rV = 0$$

V = Value of S.

we buy options to hedge.

(1) sell an option V to get cash

(2) borrow $S \cdot \frac{\partial V}{\partial S} - V$ from bank

(3) buy $\frac{\partial V}{\partial S}$ shares of stock.

Dynamic :

Hedge : buy & sell S to keep $\frac{\partial V}{\partial S}$ amount of stock

so, $\Delta = \frac{\partial V}{\partial S}$ is important.

For value of European options we have numeritical

$$V(S,t) = SN(d_1) - Ke^{-r(T-t)} N(d_2)$$

$$N(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{y^2}{2}} dy$$

$$d_\pm = \frac{\log(S/K) + (r \pm \frac{1}{2}v^2)(T-t)}{v\sqrt{T-t}}$$

1.2 Numerical Issues

(1) The PDE cannot be solved by analytical means except a few cases. Approximation is needed. Sec. II b-d.

(2) Even if an analytical solution exists it might need to be approximated. eg. $N(x)$ must be approximated. Sect II c.

(3) Real binomial method.

$$V_j^{n+1} = e^{-r\Delta t} [p V_{j+1}^n + (1-p) V_j^n]$$

$$S_j = j \cdot \Delta S$$

$$t_n = n \cdot \Delta t$$

to hedge, need $\frac{\partial V}{\partial S}$ from discrete data II c.

(4) The volaticity is an inferred quantity.

$$V(S,t) - SN(d_+ \sigma) - KEe^{-r(T-t)} N(d_- \sigma) = 0$$

(5) Data is given discretely, not as functions. We approximate data by smooth functions. II b.

(6) Sometimes we must simulate process to set solutions. eg. Mont carlo Methods. II S.

Character 2 : Basic Numerical Methods

2.1 Errors and Conditions.

2.1.1 Floating point numbers

EX:

```
float sum = 0.0f
for (int i=1 ; j<=10 ; i++ )
{
  sum += 0.1f
}
cout << sum - 1.0f
```

Result : $\sum_{i=1}^{10} 0.1 - 1 = 1.192093 \times 10^{-7}$

Computer approximation real numbers.

A real number X.

$$X = \pm \sum_{k=1}^{\infty} \frac{d_k}{\beta^k} \beta^e$$

$\beta \rightarrow$ base

$d_k \rightarrow$ digits $\quad 0 \leq d_k \leq \beta - 1$

ex: $63.58 = 0.6358 \times 10^{+2} = 0.06358 \times 10^{3}$

takes place of an exponent.

so, we required $d_1 \neq 0$, called normalized.

a real number is called normalized, if it is in the form:

$$\pm d_0 . d_1 d_2 d_3 \cdots \times 10^{n}$$

required $d_0 \neq 0$.

Converting a number to base 2 and normalizing it at the former step in storing a real number as a floating-point number in a computer.

Computers approximate X with floating-point number.

$$Fl(X) \approx X$$

$$Fl(x) = \pm\, S \cdot \beta^{e}$$

$S \to$ significate $\qquad S = \sum_{k=1}^{t} \dfrac{\hat{d}_k}{\beta^k}$ $\qquad$ finite sum

$\beta \to$ base

$e \to$ exponent $\qquad 0 \le \hat{d}_k \le \beta - 1$

normalized F.P. #, $\hat{d}_1 \neq 0$

$$L \le e \le u$$

F.P. # have different properties from real #'s.

(1) There is a finite # of F.P. #'s

$$Fl(X) = \pm \sum_{k=1}^{t} \frac{\hat{d}_k}{\beta^k} \beta^{e}$$

$2 \qquad (\beta-1)\beta^{t-1} \qquad u+|L|+1$

$$\# = 2 \times (\beta-1)\beta^{t-1}(u+|L|+1)$$

(2) They are discrete. eg: $\underbrace{\beta=10,\ L=-1,\ u=1,\ t=2}$

$\pm\, 0.\_\,\_ \times 10^{e} \qquad -1 \le e \le 1$

big gap.

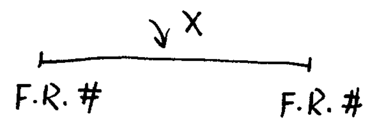
(3) They are not uniformly distributed.

(4) There is an upper and lower value.

eg. $|X| > 9.9$ is as good as ∞.

(5) They don't get arbitrage close to 0.

How do we approximate x by $fl(x)$?

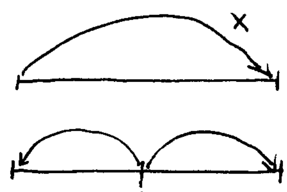


we Round

(1) chopp $\pm N6$

(2) round $\pm N6$

$$Fl(x) = \pm \sum_{k=1}^{t} \frac{d_k}{\beta^k} \cdot \beta^e$$



chop → float-to-integer $-3.9 \approx -3$

round $-3.9 \approx -4$

Today, all machines use IEEE 754 standard.

Defines 2 precisions.

(1) Single precision (float in C++)

$E_{min} = -126$ $\quad Fl(x) = (-1)^s \, 2^{e-127} \, 1.M$ $\quad$ maximum representable value is $2^{127}(2-2^{-23}) \approx 3.4 \times 10^{38}$

$E_{max} = 127$ $\quad 0 < e < 255, \quad M = \sum_{k=1}^{23} \frac{\hat{d}_k}{\beta^k}$ $\quad$ smallest representable value is $2^{-126} = 1.17549e^{-38}$

$\epsilon_{s.p} = 2^{-23} = 1.19209e^{-7}$

$\beta = 2$

$S \to 1$ bit, $e \to 8$ bits $\quad$ The largest spacing $2^{127}(2-2^{-23}) - 2^{127}(2-2^{-23}-2^{-23}) = 2^{104}$

$Fl(x) \to 32$ bits $\Rightarrow M = 23$ bits $\quad$ The smallest $2^{-126}(1+2^{-23}) - 2^{-126} = 2^{-149}$

(2) double precision

$E_{min} = -1022$ $\quad Fl(x) = (-1)^s \, 2^{e-1023} \, 1.M$ $\quad$ largest $1.79769e^{308} = 2^{1023}(2-2^{-52})$

$E_{max} = 1023$ $\quad e \to 11$ bits, $M = 52$ bits. $\quad$ smallest $2^{-1022} \approx 2.2251e^{-308}$

Total → 64 bits. $\quad$ epsilon $2.2204e^{-16} = 2^{-52}$

The extra value of e flag exceptions

① NaN. (Not a number) eg. $\frac{0}{0}$, $\sqrt{-1}$,

$e = 255$, $M \neq 0$ (S.P.)

② INF (Infinity) eg. $\frac{1}{0}$, $e = 255$, $M = 0$ (S.P)

③ denormalization.

$\cancel{Fl(x) = \pm 0.Me}$ $\cancel{Fl(x) = \pm 0.M2e}$

$Fl(x) = \pm 0.M\,2^{e-127}$ S.P. $e = 0$

3.14159265357928 ...

$\beta = 10$

$t = 6$

$u = 4 = -L$

0.314159×10^{1} /10

0.314159×10^{0} /10

0.314159×10^{-1} /10

0.314159×10^{-2} /10

0.314159×10^{-3} /10

0.314159×10^{-4} → the largest negative denomali

0.314159×10^{-4} /10 since $u = 4 = -L$.

0.031416×10^{-4} /10

0.003142×10^{-4}

2.1.2 Measuring Errors.

Define. let p^* approximates p.

Absolute Error. $E_{abs} = p^* - p$

Relative Error. $E_{rel} = E_{abs} / p$ $p \neq 0$

Eg: $p = 0.1$, $p^* = 0.2$

$E_{abs} = p^* - p = 0.1$

$E_{rel} = \frac{E_{abs}}{p} = \frac{0.1}{0.1} = 1$

Erel is more important, it tells us the significants of error.

Define: p^* approximates p to M significant digits (sig. figures, sig. figs)

if $\quad E_{rel} \le 0.5 \times 10^{-M} = 5 \times 10^{-(m+1)}$

$E_{rel} \le 0.5 \times 10^{-M} = 5 \times 10^{-(m+1)}$

$p = \pi, \quad p^* = 3.1415$

$|E_{abs}| = 9.2 \times 10^{-5}$

$$|E_{rel}| = \frac{9.2\times 10^{-5}}{\pi} \approx 3 \times 10^{-5} < 5 \times 10^{-5} \Rightarrow m = 4$$

$Fl(x) = (1+\varepsilon)\,x$

$$\Rightarrow \left|\frac{Fl(x)-x}{x}\right| = |\underset{\substack{\uparrow \\ E_{rel}.}}{\varepsilon}|$$

what is ε?

$$x = \sum_{k=1}^{\infty} \frac{d_k}{\beta^k}\cdot \beta^e$$

$$Fl(x) = \sum_{k=1}^{t} \frac{d_k}{\beta^k}\beta^e \qquad \text{chopped arithmetic}$$

$$-\varepsilon = \frac{\sum_{k=1}^{\infty}\frac{d_k}{\beta^k}\beta^e - \sum_{k=1}^{t}\frac{d_k}{\beta^k}\cdot\beta^e}{\sum_{k=1}^{\infty}\frac{d_k}{\beta^k}\beta^e}$$

$$= \sum_{k=t+1}^{\infty}\frac{d_k}{\beta^k}\beta^e \Big/ \sum_{k=1}^{\infty}\frac{d_k}{\beta^k}\cdot\beta^e \le \frac{\frac{1}{\beta^{t+1}}\sum_{k=0}^{\infty}\frac{d_{k+t+1}}{\beta^k}}{\frac{1}{\beta}}$$

$$\le \frac{1}{\beta^{t+1}}\cdot\beta \Big/ \frac{1}{\beta} = \beta^{1-t}$$

$|\varepsilon| \le \beta^{1-t} = \epsilon = $ machine epsilon (chop)

$\epsilon_{round} = \frac{1}{2}\epsilon_{chop} = \frac{1}{2}\beta^{1-t}$

IEEE defines ϵ as the unit round: Find ϵ, $1+\epsilon > 1$

eg. S.P. $1.000\cdots0$ 23 bits

$0.000\cdots1 = 2^{-23} = 2^{-t}$

↑ 23rd place

$1.000\cdots1 > 1$

D.P. $\epsilon = 2^{-52}$

$Fl(x) = (-1)^s\, 1.M \times 2^{e-127}$

significant figures

IEEE S.P. $\epsilon_{S.P} = 2^{-23} = 1.192\times10^{-7}$ $m=6$

$\epsilon_{D.P} = 2^{-52} = 2.2\times10^{-16}$ $m=15$

2.1.3 Conditioning.

EX: $p(x) = (x-1)(x-2)\cdots(x-20) = x^{20} - 210x^{19} + \cdots$ Roots ?

Suppose we perturb 210 to $210 + 2^{-23}$

How do the root change ?

$$E_{rel,\,IN} = 2^{-23}/210 = 5.7\times10^{-10} \qquad m = 8$$
$$= 0.57\times10^{-9}$$

One root.

$x=20$ changes to $x^* = 20.846908101$

$E_{abs,\,out} = 0.846908101$

$E_{rel,\,out} = 0.846908101/20 = 0.042345405$

$= 4.2345405\times10^{-2}$ $m=1$

lost 7 digits.

amplification $\dfrac{E_{rel,\,out}}{E_{rel,\,in}} = \dfrac{0.04}{6\times10^{-10}} = 8\times10^{7}$

Error amplified by 80 million

Example of an ill-conditioned / sensitive problem what is an ill-conditioned problem?

when $\dfrac{E_{rel,out}}{E_{rel,in}} \gg 1$

Float class F ;

FM Float X ;

Fm Float EPS ;

EPS = F · epsilon(X)

Conditioning.

$$\xrightarrow[\text{parameter}]{\text{Input}} \boxed{\text{Problem}} \xrightarrow[\text{Solution}]{\text{Output}}$$

$$\xrightarrow[\text{Para. Error}]{\text{Input Error}} \boxed{\text{Problem}} \xrightarrow[\text{Soln. Err.}]{\text{Output Error}}$$

Relative size of the errors gives the application

$$\left| \frac{E_{rel,\ solution}}{E_{rel,\ input}} \right|$$

if this is large, the problem is ill-conditioned.

Quantify conditions.

Suppose a solution X is a function of a parameter α

$$X = X(\alpha)$$

Let $\Delta\alpha = a$ in α

$$\Delta X = X(\alpha + \Delta\alpha) - X(\alpha)$$

Then relative change in input is $\dfrac{\Delta\alpha}{\alpha}$

output $\dfrac{\Delta X}{X}$

Then $\dfrac{\frac{\Delta X}{X}}{\frac{\Delta\alpha}{\alpha}} = \dfrac{\alpha}{X} \cdot \dfrac{\Delta X}{\Delta\alpha}$

Let $\Delta\alpha \to 0$

$$\lim_{\Delta\alpha \to 0} \left| \dfrac{\frac{\Delta X}{X}}{\frac{\Delta\alpha}{\alpha}} \right| = \left| \dfrac{\alpha}{X} \cdot \dfrac{\Delta X}{\Delta\alpha} \right|$$

And define the condition number

$$K = \max \left| \dfrac{\alpha}{X} \cdot \dfrac{\Delta X}{\Delta\alpha} \right|$$

And a problem is ill-conditioned if $K \gg 1$

what is $\gg 1$?

Answer: It depends.

Note: $\dfrac{\frac{\Delta X}{X}}{\frac{\Delta\alpha}{\alpha}} \sim K_\alpha$

In context

$\dfrac{\Delta X}{X} \to E_{rel,solution}$

$\dfrac{\Delta\alpha}{\alpha} \to E_{rel,in}$

$$\dfrac{E_{rel,soln}}{E_{rel,in}} \sim K_\alpha$$

$$\bar{E}_{rel,soln} \sim K\, E_{rel,in}$$

If $E_{rel,in}$ is due to rounding

$$\bar{E}_{rel,in} \sim \underset{\substack{\uparrow \\ \text{epsilon}}}{\epsilon}$$

$$\bar{E}_{rel,soln} \sim K \cdot \epsilon$$

then $\log(E_{rel,soln}) \sim \log(K) + \log(\varepsilon)$

Then in significant digits if

$$\log(K) + \log(\varepsilon) \le \log 0.5 - m$$

$$m \le \log 0.5 - \underset{\uparrow}{\log(K)} \underbrace{-\log(\varepsilon)}_{>0 \text{ fixed}}$$

For each of 10 in K, we lose 1 significant digit.

What is big? Depends on how many sig. digits needed.

eg. $K = 10^4$

$\epsilon = 10^{-7}$ & $\log(0.5) - \log(\varepsilon) \approx 6$

then $m \sim 6 - 4 = 2$

If you need $m = 4$, then $K = 10^4$ is too big.

On the other hand. (OTOH).

if we need 15 sig. digits in input (eg IEEE D.P)

then $m = 15 - 4 = 11$ ok.

EX:

$$x = e^{\alpha}, \quad K = \left|\frac{\alpha}{x} \cdot \frac{\Delta x}{\Delta \alpha}\right| = \left|\frac{\alpha}{e^{\alpha}} \cdot e^{\alpha}\right| = |\alpha|$$

Suppose $\alpha = 5.5$

lose at most 1 sig. digit.

we can also do systems $\vec{x} = \vec{x}(\vec{\alpha})$

then $x(\vec{\alpha} + \Delta\vec{\alpha}) = x(\vec{\alpha}) + \Big(\underbrace{\frac{\partial \vec{x}}{\partial \vec{\alpha}}}\ \vec{\xi}\Big) \Delta\vec{\alpha}$

Jacobian

so $\Delta\vec{x} = J(\vec{\xi}) \Delta\vec{\alpha}$

$$\|\Delta\vec{x}\| = \|J \Delta\vec{\alpha}\| \leq \|J\| \cdot \|\Delta\vec{\alpha}\|$$

$$\frac{\|\Delta\vec{x}\|}{\|\vec{x}\|} \leq \frac{\|J\|}{\|x\|} \cdot \left(\frac{\|\Delta\vec{\alpha}\|}{\|\vec{\alpha}\|}\right) \cdot \|\vec{\alpha}\|$$

relative error in parameters

$$\frac{\frac{\|\Delta\vec{x}\|}{\|x\|}}{\frac{\|\Delta\vec{\alpha}\|}{\|\alpha\|}} \leq \frac{\|J\| \cdot \|\vec{\alpha}\|}{\|\vec{x}\|}$$

Refine. $K = \max \dfrac{\|J\| \|\vec{\alpha}\|}{\|\vec{x}\|}$

Linear System are often ill-conditioned.

$$Ax = y$$

actually, we solve $(A + \Delta A) X^* = y + \Delta y$

OR $\quad AX^* = y + \Delta y - \Delta A X^* \equiv y^*$

$\Leftrightarrow \quad AX^* = y^*$

$$k = \max \frac{\|J\| \, \|\vec{y}^*\|}{\|X\|}$$

what is J ?

$$J = \frac{\partial \vec{x}^*}{\partial \vec{y}^*} \qquad \vec{x}^* = A^{-1} \vec{y}^*$$

$$\frac{\partial \vec{x}^*}{\partial \vec{y}^*} = A^{-1}$$

also, $y^* = AX^*$

1.20

$$AX^* = y^*$$

$$\frac{\dfrac{\|\Delta X^*\|}{\|X^*\|}}{\dfrac{\|\Delta y^*\|}{\|y^*\|}} = \frac{\|J\| \, \|\vec{y}^*\|}{\|\vec{x}^*\|}$$

$$J = \frac{\partial x^*}{\partial y^*} = A^{-1}$$

$$\frac{\|A^{-1}\| \, \|AX^*\|}{\|\vec{x}^*\|} \leq \frac{\|A^{-1}\| \cdot \|A\| \cdot \|\vec{x}^*\|}{\|X^*\|}$$

$$k(A) = \|A^{-1}\| \cdot \|A\|$$

EX: $A = \begin{vmatrix} 1 & 1.01 \\ 0.99 & 1 \end{vmatrix}$

Norm: $\|A\|_\infty = \max_i \sum_{j=1} \|a_{ij}\|$

Here. $\|A\|_\infty = \max\{2.01, 1.99\} = 2.01$

$$A^{-1} = \underbrace{\frac{1}{1-0.99\times 1.01}}_{= 10^4} \begin{vmatrix} 1 & -1.01 \\ -0.99 & 1 \end{vmatrix}$$

$$\|A^{-1}\|_\infty = 10^4 \max\{2.01, 1.99\} = 2.01\times 10^4$$

solution in single precision with have only 2 sig. figures

$$\kappa(A) = 2.01\times 2.01\times 10^4 = 4\times 10^4$$

2.1.4 Stability of Algorithem

EX: Compute $e^{-5.5}$ is this ill-conditioned?

$\kappa = 5.5$ Not

$e^{-x} = 1 - x + \frac{x^2}{2} - \frac{x^3}{3!} + \cdots$ Taylor Series

choose $\beta = 10$, $t = 5$

$$e^{-5.5} = 1.0000 - 5.5000 + 15.250 - \overset{\text{round}}{27.730} + \overset{\text{round}}{38.129} - \cdots$$

$$= 0.0026363 = 0.26363\times 10^{-2}$$

50% error, No sig. digits.

exact: 0.0040868.

solution is on size of rounding errors.

Fixed? $\frac{1}{e^{5.5}} = \frac{1}{1.0000 - 5.5000 + \cdots} = 0.0040865$.

characteristic of algorithem.

(1) subtraction of nearly equal numbers (catastrophic cancellation).

ex: exact arithmetic

approx. $\beta=10$, $t=\cancel{15}\ 5$

$$\begin{array}{r} 151.72899 \\ -151.71422 \\ \hline 0.01477 \end{array} \text{ exact} \qquad \begin{array}{r} 151.73 \\ -151.71 \\ \hline 0.020000 \end{array}$$

$$E_{rel} = \frac{E_{abs}}{p} = \frac{p^*-p}{p}$$

$$E_{rel} = \frac{0.02-0.01477}{0.01477} = 0.35 < 0.5\times10^0$$

$\downarrow$ no sig. figures.

Rule #1 avoid catastrophic cancellation

we often must be clever to this.

Probabilities:

(a) analytically rearrange the forward to remove subtractions.

ex: $e^{-5.5} = \dfrac{1}{1-5.5+\frac{5.5^2}{2}-\cdots}$

ex: $y = x(\sqrt{x+1}-\sqrt{x})$

$$= x(\sqrt{x+1}-\sqrt{x})\left(\frac{\sqrt{x+1}+\sqrt{x}}{\sqrt{x+1}+\sqrt{x}}\right) = \frac{x}{\sqrt{x+1}+\sqrt{x}}$$

given different answers.

(b) make approximations, so the subtraction can be done analytically.

(2) avoid multplying errors by large factors.

(happens in recursive algorithem)

EX: $Ax=y$, usually solved by Guess-Elimination

$$\begin{vmatrix} 10^{-10} & 1 \\ 1 & 0 \end{vmatrix} = \begin{vmatrix} X \\ Y \end{vmatrix} = \begin{vmatrix} 1+10^{-10} \\ 1 \end{vmatrix}$$

IEEE S.P.
exact $= \begin{vmatrix} 1 \\ 1 \end{vmatrix}$

approximation

$$\begin{vmatrix} 10^{-10} & 1 \\ 1 & 0 \end{vmatrix} \begin{vmatrix} X \\ Y \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \end{vmatrix}$$

$$\begin{vmatrix} 10^{-10} & 1 \\ 0 & -10^{-10} \end{vmatrix} \begin{vmatrix} X \\ Y \end{vmatrix} = \begin{vmatrix} 1 \\ -10^{-10} \end{vmatrix} \Rightarrow$$

$1-10^{10} = -(10^{10}-1) = -10^{10}$

$y=1,\ X=0 \leftarrow 10^{-10}X+1=1$

Fix: Re-order so multiplier <1

$$\begin{vmatrix} 1 & 0 \\ 10^{-10} & 1 \end{vmatrix} \begin{vmatrix} X \\ Y \end{vmatrix} = \begin{vmatrix} 1 \\ 1+10^{-10} \end{vmatrix}$$

approximation $\begin{vmatrix} 1 & 0 \\ 10^{-10} & 1 \end{vmatrix} = \begin{vmatrix} 1 \\ 1 \end{vmatrix}$

$$\begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} \begin{vmatrix} X \\ Y \end{vmatrix} = \begin{vmatrix} 1 \\ 1-10^{-10}=1 \end{vmatrix}$$

$X=1$

$Y=1$

called Guess-Elimination with partial pivoting

B) Avoid Range Errors.

EX: $r = \sqrt{x^2+y^2}$

$x = 10^{20},\ y = 10^{20} \Rightarrow r = \sqrt{2}\ 10^{20}$

S.P. IEEE $\quad r = \text{I.N.F} \quad (10^{20})^2 = 10^{40} > 10^{38}$ Infinity

FIX: IF $|y|>|x|$, $r = |y| * \sqrt{1+(\frac{x}{y})^2}$

else $r = |x| * \sqrt{1+(\frac{y}{x})^2}$

2.2 Solution of nonlinear algebric equations

EX: V = value of an account

P = amount deposited periodically

i = interest / period

$$V = \frac{P}{i}\left((1+i)^N - 1\right)$$

Q: what is the min. interest rate needed to have an amount V in an account after N periods if we invest P.

Find i given V, N, P,

can't do directly

we can write as a root finding problem. Find $x \to F(x) = 0$

eg. $V \to \frac{P}{i}\left((1+i)^N - 1\right) = 0$

Two methods

(1) Bisection

6 totally convergent but slow.

(2) Newton's Method

Fast convergent, but not always convergent.

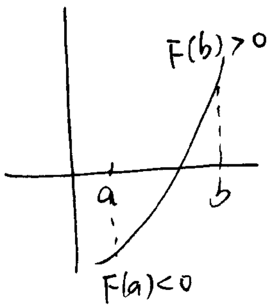
Bisection: Binary Search

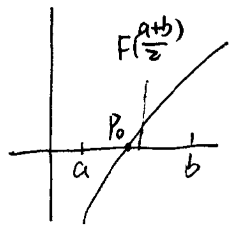
Theorem: (Intermediate value them)

If $I = [a, b]$, $F(x) \in C(I)$. and $F(a)F(b) < 0$

then $F(x) = 0$ at some point $p \in [a, b]$

To bisec.

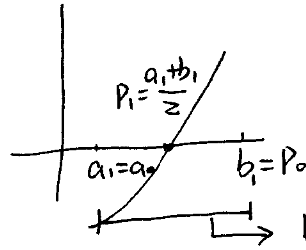




$a_0 = a$

$b_0 = b$

$P_0 = \frac{a+b}{2}$



$a_0 = a_1$

$P_0 = b_1$

$P_1 = \frac{a_1+b_1}{2}$

$b_n - a_n = \frac{b_{n-1} - a_{n-1}}{2}$

$P_n = \frac{a_n+b_n}{2}$

$|P_n - P| \le \frac{b-a}{2^{n+1}} \qquad n \geq 0$

$|p - p_n| \le \frac{1}{2}(b_n - a_n)$

REM: (1) clearly $\lim_{n\to\infty} |P_n - P| = 0$

(2) at each step, the error bond decreases by a factor of 2.

$b_n - a_n = \frac{b_{n-1} - a_{n-1}}{2}$

$P_n = \frac{a_n+b_n}{2}$

Proof:

$$b_n - a_n = \frac{b_{n-1} - a_{n-1}}{2} = \frac{b_{n-2} - a_{n-2}}{2^2} = \cdots = \frac{b_{n-n+1} - a_{n-n+1}}{2^{n-1}} = \frac{b-a}{2^n}$$

$$|p - p_n| \le \frac{1}{2}(b_n - a_n) = \frac{b-a}{2^{n+1}}$$

$$|p - p_n| \le \frac{1}{2}(b_n - a_n) = \frac{1}{2}\cdot\frac{b_{n-1} - a_{n-1}}{2} = \frac{1}{4}\cdot\frac{b_{n-2} - a_{n-2}}{2} = \cdots = \frac{b-a}{2^{n+1}}$$

An algorithem:

$P_n = P_{n-1} + \frac{b_{n-1} - a_{n-1}}{2}$

For $i = 1$ to N, Do $p = \frac{a+b}{2}$

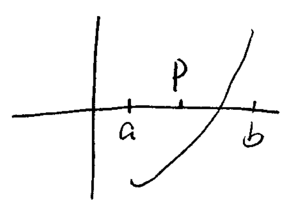
if converged exit

if $F(a)\,F(p) > 0$, $a = p$

else $b = p$

endif

Next i.



$b_n - a_n = \frac{b_{n-1} - a_{n-1}}{2}$

$P_n = \frac{b_n + a_n}{2}$

$|p - P_n| \le \frac{b-a}{2^{n+1}}$

Are there floating point #s ? 1

How do we test convergence . 2

what is N ? 3

1. F.P. Problems

(a) Find $F(a)F(p)$ can overflow or underflow

Test $\text{sign}(F(a)) * \text{sign}(F(p)) > 0$

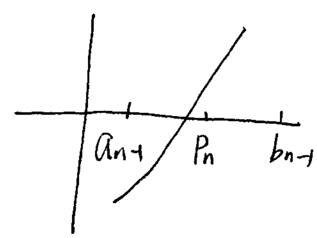
(b) $p = \frac{a+b}{2}$

is not necessarily in $[a, b]$

eg. $\beta = 10$, $t = 3$, $\frac{0.981 + 0.982}{2} = \frac{\overset{\substack{3\text{ digits} \\ \uparrow}}{1.964}}{2} = 0.980 \neq 0.982$

normally rewrite as a correction

$$p_n = p_{n-1} + \frac{b_{n-1} - a_{n-1}}{2}$$



2. Test for convergence .

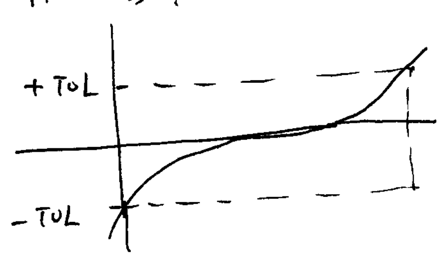
(1) $F(p_n) \leq TOL$ Root test

(2) $|p - p_n| \leq TOL$ absolute error test Tol → tolerance

(3) $\frac{|p - p_n|}{|p|} \leq TOL$ Relative error test

Root test Problems ?

eg.



where is root ?

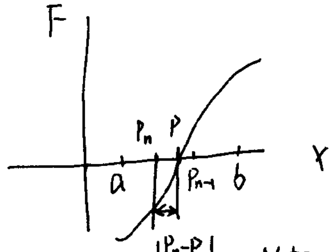
It can be anywhere

in $\boxed{}$

where $dF/dx = 0$

$$K = \left|\frac{\Delta F}{F}\right| / \left|\frac{\Delta x}{x}\right| = \left|\frac{x}{F} \cdot \frac{dF}{dx}\right| \quad F \approx 0$$

estimate error



Note $|P_n - P| \le |P_n - P_{n-1}|$

estimate / bound on error

$|P_n - P| \le |P_n - P_{n-1}|$

Test: $|P_n - P_{n-1}| \le$ absolute tolerance + relative tolerance $* |P_n|$

$|P_n - P_{n-1}| \le absTol + relTol * |P_n|$

when $|P_n| \approx 0$, we has abs. err. test

when $|P_n| >> 0$, whe have rel. err. test

what should tolerance be?

$\epsilon \le$ relative tolorance $\le 5 \times 10^{-(m+1)}$

$\epsilon \le relTol \le 5 \times 10^{-(m+1)}$

$m \to$ desired # sig. figures

$\epsilon \to$ machine epsilon

absolute tolerance $\approx$ smallest distance between 2 #s.

absTol $\approx$ smallest distance between 2 numbers

$2^{-149} \approx 10^{-38}$

3. what is N ?

Recall $|P_n - P| \le \frac{b-a}{2^{n+1}}$

$|P_n - P| \le |P_n - P_{n-1}|$

so we converge if $|P_n - P_{n-1}| \le \frac{b-a}{2^{n+1}}$

and again if $\frac{b-a}{2^{n+1}} \le abstol + reltol * |P_n|$

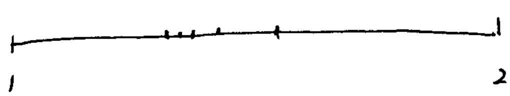
ie. if $\frac{b-a}{abstol + reltol * |P_n|} \le 2^{n+1}$

or $n+1 \geq \log_2 \frac{b-a}{abstol + reltol * |P_n|}$

safe if $N = \log_2 \dfrac{b-a}{abstol + reltol * \left|\frac{a+b}{2}\right|}$

$$N_{it} = \frac{\log \dfrac{b-a}{abstol + reltol * \left|\frac{a+b}{2}\right|}}{\log 2}$$

```
for i=1 to Nit
    p = a + (b-a)/2
    if (b-a)/2 <= abstol + reltol*|p|     exit
    if sign(F(a)) * sig(F(p)) > 0
        a=p
        else
            b=p
    endif
Next i.
```

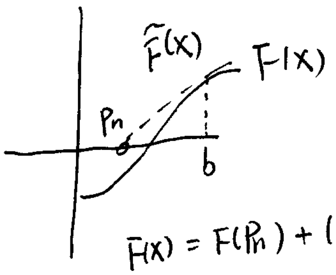


$\frac{1}{2} \cdot \frac{1}{2} \cdots \frac{1}{2^n} = \frac{1}{2^{23}} \Rightarrow n = 23$

1.27 $F(x) = 0$

Newton's Method

Find roots of successive linear approximation of F



$$F(x) = F(P_n) + (x-P_n)F'(P_n) + \frac{(x-P_n)^2}{2!} \cdot F''(\xi)$$

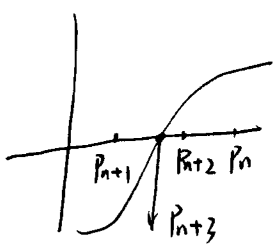
$$\tilde{F} = F(P_n) + (x-P_n)F'(P_n)$$

Find root of $\tilde{F}$, call it P_{n+1}

$$F(P_n) + (P_{n+1} - P_n)F'(P_n) = 0$$

Solve for $P_{n+1} = P_n - \dfrac{F(P_n)}{F'(P_n)}$

Graphically



Given $P = P_0 \rightarrow$ what's initial ?

For $i = 1$ to $N \rightarrow$ what's N

$P = F - \dfrac{F(P)}{F'(P)} \rightarrow$ what if $F'(p) = 0$

If converged exit

↑ How do you tell ?

Next i.

Theorem : let $F \in C^2[a,b]$, suppose $F(p) = 0$, for some $p = [a,b]$ and assume $F'(p) \neq 0$ for $x \in [a,b]$, then $\exists \delta > 0 \ni \{P_n\}_{n=0}^{\infty}$ converge to p for any $P_0 \in [p-\delta, p+\delta]$

REM: For smooth enough F, Newton's Method converges if the initial guess is "close enough"

Proof: Let $g(x) = x - \frac{F(x)}{F'(x)}$

Then $P_{n+1} = g(P_n)$

Note $P = g(p)$ (p is a fixed point of g)

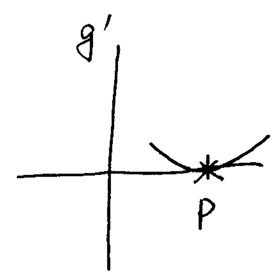
Now $g'(x) = 1 - \frac{F(x)F''}{(F')^2} - 1 = \frac{F(x)F''(x)}{(F')^2}$

so that $g'(p) = 0$, $g'(x)$ is continuous.

so there is an interval $x \in [p-\delta, p+\delta]$

$\ni g'(x) < 1$
so that

i.e. $|g'| \le k < 1$, $\forall |x-p| \le \delta$
for all



p is root
in the neighborhood of P
g has to close to 0.

Next $|g(x) - p| = |g(x) - g(p)|$
$= |g'(\delta)|\,|x-p|$
$\le k|x-p|$

choose $x = P_n$, $\ni P_{n+1} = g'(P_n)$

so $|P_{n+1} - P| \le k|P_n - P|$
$\le k^2 |P_{n-1} - P| \le \cdots \le k^{n+1}|P_0 - P|$

$\ni$ since $k < 1$, $\lim_{n\to\infty} |e_{n+1}| \equiv \lim_{n\to\infty} |P_{n+1} - P|$
$\le \lim_{n\to\infty} k^{n+1} |P_n - P| = 0$

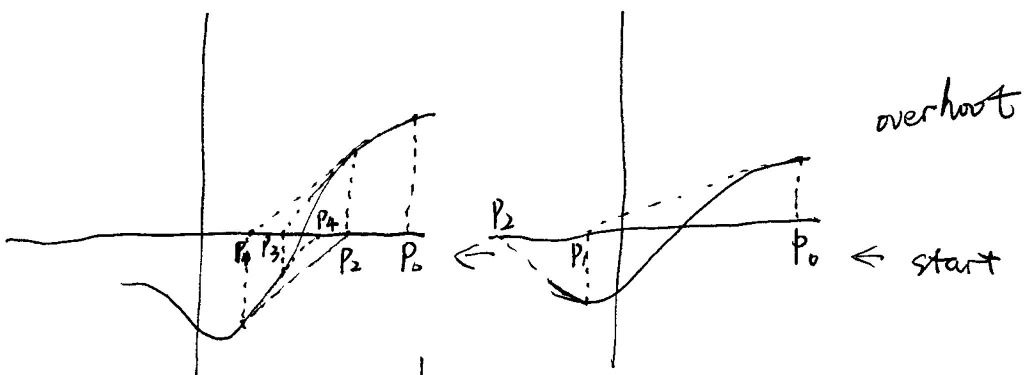
so $P_n \to P$ as $n \to \infty$

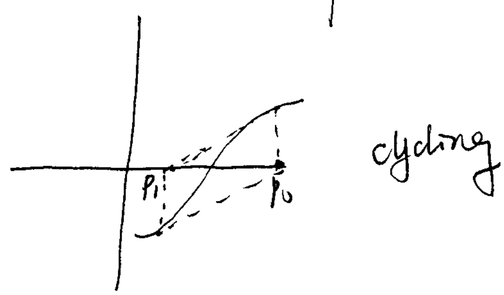
So what is "close enough"

$$\frac{F(X)\,F''(X)}{(F')^2} < 1$$

in practice we don't test this

otherwise it can fail.





1.29 $F(X)=0$

Given $P = P_1$

For $i = 1$ to N

$$P = P - \frac{F(P)}{F'(P)}$$

If converge exit

Next i.

Sometimes this is modified so that P doesn't change too much in each interation:

Newton Method

change: $\Delta = -\dfrac{F(P)}{F'(P)}$ i.e. $P = P + \Delta \Leftrightarrow P \leftarrow P + \Delta$

so required $\frac{|\Delta|}{|p|} \le \beta \rightarrow$ some value

and update β^y

$$\Delta \leftarrow \beta |p| \, \text{sign}(\Delta)$$

algorithm becomes

```
Given P0
For i = 1 to N
    Δ = F(p)/F'(p)
    if |Δ| ≥ β|p|
        Δ = β|p| sig(Δ)
    Endif
    p = p + Δ
    if converged exit
Next i.
```

$\beta \sim 0.10$
$0.5 \times 10^{-1} = 0.05$

Definition:

Suppose $\lim_{n\to\infty} P_n \to P$, Let $e_n = P_n - P$

the sequence $\{P_n\}_{n=0}^{\infty}$ converges with order α if

$$\lim_{n\to\infty} |P_{n+1} - P| \Big/ |P_n - P|^{\alpha} = \lim_{n\to\infty} \frac{e_{n+1}}{|e_n|^{\alpha}} = \lambda \rightarrow \text{constant value}$$

REM: if $\alpha = 1 \rightarrow$ linear convergence

if $\alpha > 1 \rightarrow$ superlinear convergence

if $\alpha = 2 \rightarrow$ quadratic

The bound in bisection is linear $\lambda = \frac{1}{2}$

Theorem: Let p be a fixed point of $x = g(x)$.
suppose $g'(p) = 0$ and g'' is continuous in some open interval about p. Then
$\exists \delta > 0 \Rightarrow$ for $p_0 \in [p-\delta, p+\delta]$
then sequence $\{p_n\}_{n=0}^{\infty}$ generated by $p_{n+1} = g(p_n)$.
converges quadratically.

Proof: we've already shown convergence
write $g(x) = g(p) + g'(p)(x-p) + g''(\xi)(x-p)^2 \frac{1}{2}$

but $g'(p) = 0$, so

$$g(x) = \cancel{g(p)}\ p + \frac{1}{2} g''(\xi)(x-p)^2$$

choose $x = p_n$, then $g(p_n) = p_{n+1}$

$$p_{n+1} = p + \frac{1}{2} g''(\xi)(x-p)^2$$

$$e_{n+1} = \frac{1}{2} g''(\xi)(x-p)^2 = \frac{1}{2} g''(\xi) e_n^2$$

so $\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|} = \lim_{n\to\infty} \frac{1}{2} |g''(\xi)| = \frac{1}{2} |g''(p)| = \lambda$

$\Rightarrow$ quadratic convergence.

Corollary: Newton's Method converges quadratically.

Proof:

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

$$g'(x) = 1 - \frac{F'(x)^2 - F(x)F''(x)}{(F'(x))^2}$$

$$= \frac{F(x)\,F''(x)}{\ldots}$$ so $g'(p) = 0$???

Look at error

$$|e_{n+1}| \to \lambda |e_n|^2 \quad \text{as } n \to \infty$$

$$\log|e_{n+1}| \to 2\log|e_n| + \log\lambda$$

Similarity $\log\frac{|e_{n+1}|}{|p|} \to 2\log\frac{|e_n|}{|p|} + \log|p\lambda|$

→ exponent of the error doubles

suppose we start with $\left|\frac{e_0}{p}\right| \approx 10^{-1}$

$\left|\frac{e_1}{p}\right| \approx 10^{-2}$

D.P. 4 iterations

S.P 1 interation

$\left|\frac{e_2}{p}\right| \approx 10^{-4}$

$\left|\frac{e_3}{p}\right| \approx 10^{-8}$

$\left|\frac{e_4}{p}\right| \approx 10^{-16}$ → why stop here?

machine ϵ

when is it converge?

$$|\Delta| = |P_{n+1} - P_n|$$

Note $\Delta = P_{n+1} - P_n = (P_{n+1} - p) - (P_n - p) = e_{n+1} - e_n$

But $|e_{n+1}| \to \lambda|e_n|^2 \ll \lambda|e_n|$

so $|\Delta| \approx |e_n|$

converge test

$$|\Delta| \leq abstol + |P_{n+1}|\, reltol$$

$$N = 6 \sim 10$$

to get P_0 use bisection

TO A'-rel rel error of 0.5×10^{-1}

2.1

2.3 Interpolation

Motivation: ~~Implet~~ / Implied volatility Black-Scholes model assumes constant volatility in asset prices & gives equation for an option price.

$$V_t + \frac{\gamma^2 S^2}{2} \cdot \frac{\partial^2 V}{\partial S^2} + r S \frac{\partial V}{\partial S} - r V = 0$$

$\gamma \to$ volatility

But σ is not measurable. what's publised are S, t, and V.

At discret times & S, so σ is inferred, and it's not constant.

Implied volatility problems:

Find σ for each S, t & V published.

$$V(S,t) - V_{BS}(S, t_j, \quad) = 0$$

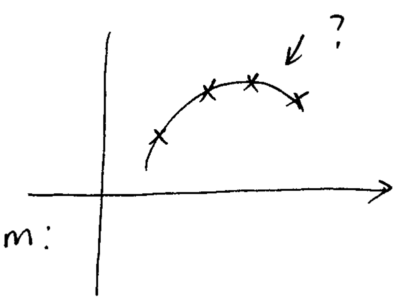
published

This gives $\sigma(S_i, t_j)$, ie. discret values.

Then they compute

$$V_t + \frac{\sigma^2(S,t)}{2} \frac{\partial^2 V}{\partial S^2} + r S \frac{\partial V}{\partial S} - r V = 0$$

need a function ~~form~~ for data

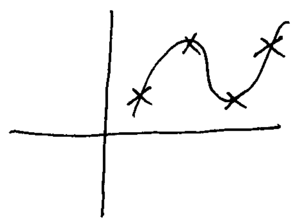


problem:

How do we find a function form to discrite the data?

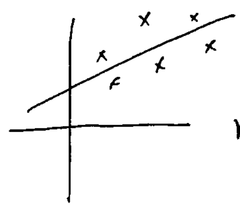
2.3.2 Polynomial Approximation two fundamental problems

(1) Data is assumed to be exact. The approximation must match data.



~~Inper~~ Interpolation

(2) Data is assumed to has errors. Try to find the best "approx." of a given form.



regression/least squares

we usually use polynomials

$$P_N(x) = \sum_{n=0}^{N} a_n x^n$$

Interpolation: $\quad Y_i = P_N(X_i) \quad x_i = 0, 1, \cdots, N$

least squares: $\quad$ minimize $\sum_{i=0}^{N} \left(Y_i - P_N(X_i) \right)^2$

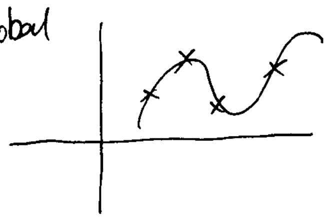
But we can also use other functional forms, eg for periodic problems

$$P_N(x) = \sum_{n=-N}^{N} a_n e^{inx}$$

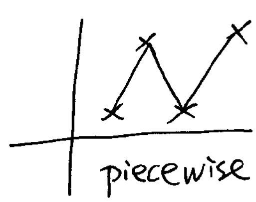
fourier series

The approx. can be global or local.

global



local



piecewise linear

Note: P.W approx.'s don't have to be smoothness constraints include splines.

smooth P.W approx's with additional

2.3.3 Gloabal Polynomial Interpolation

Given data $\{(X_i, Y_i)\}_{n=0}^{N}$

The Polynomial interpolant $P_N(X)$ satisfies

$$Y_i = P_N(X_i) \qquad i=0,1,\cdots,N$$

ie. with $P_N = \sum_{n=0}^{N} a_n X^n$, $Y_i = \sum_{n=0}^{N} a_n (X_i^n)$ $\quad i=0,1,\cdots,N$

$\uparrow$ min.

N+1 unknown

N+1 equations

or $\vec{Y} = M\vec{a}$

$$\vec{Y} = \begin{bmatrix} Y_0 \\ Y_1 \\ \vdots \\ Y_N \end{bmatrix} \quad \vec{a} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} \qquad M_{in} = X_i^n$$

so $\vec{a} = M^{-1}\vec{Y}$

unfortunately, (for X_i's uniformly spaced)

$$k_1(M) = 10^N \qquad N \sim 3 \text{ to } 4$$

Moral: Don't do this!

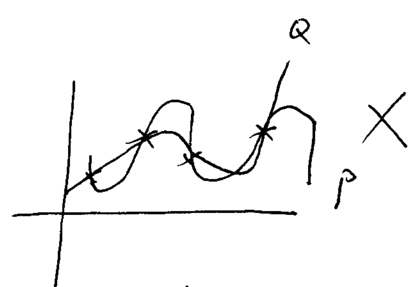
THM: Uniqueness.

Let P & Q be polynomials of degree $\le N$.

If the nodes $\{X_i\}_{i=0}^{N}$ are distinct

then $P(X_i) = Q(X_i)$ $\quad i=0,1,\cdots,N$

Then $P=Q$, $\forall X$



can't happen.

Proof: assume $P \neq 0$ and define $H(X) = P-Q$.

then $H(X)$ is polynomial of degree $\le N$ and

$H(X_i) = 0$ $\quad i=0,1,\cdots,N$

N+1 roots, polynomial degree N.

the only happen if $H(X)=0$, $P=Q$

We will use two forms of P_N to find polynomial interpolant

(1) Largrange

(2) Newton

$$\boxed{\text{Largrange : The Largrange is } \quad P_N(x) = \sum_{i=0}^{N} Y_i L_i(x)}$$

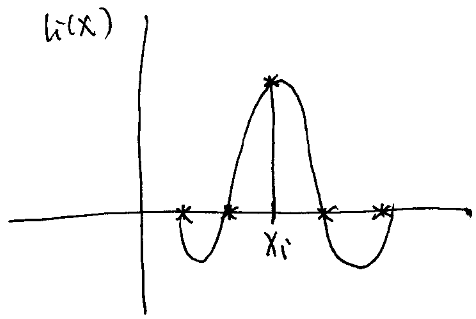
largrange interpolant polynomials

where $l_i(x)$ satisfies

(1) $l_i(x)$ polynomials of degree N.

(2) $L_i(X_j) = \begin{cases} 1 & i=j \\ 0 & i \neq j \end{cases}$

$= \delta_{ij}$



How to construct them?

$$l_i(x) = \underbrace{(x- \quad)(x- \quad)\cdots(x- \quad)}_{N \text{ factors,}} = (x-X_0)(x-X_1)\cdots(x-X_{i-1})(x-X_{i+1})\cdots(x-X$$

roots are all except $x = X_i$.

$$l_i(x) = \frac{(x-X_0)(x-X_1)\cdots(x-X_{i-1})(x-X_{i+1})\cdots(x-X_N)}{(X_i-X_0)(X_i-X_1)\cdots(X_i-X_{i-1})(X_i-X_{i+1})\cdots(X_i-X_N)}$$

$$L_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{N} \frac{x-X_j}{X_i-X_j} \qquad P_N(x) = \sum_{i=0}^{N} Y_i \left(\prod_{\substack{j=0 \\ j\neq i}}^{N} \frac{x-X_j}{X_i-X_j} \right)$$

EX: Find line through (1, 1) (3, 5)

$$P_1(x) = (1)\frac{(x-3)}{(1-3)} + (5)\frac{(x-1)}{(3-1)}$$

$\boxed{2.3}$

$$P_N = \sum_{n=0}^{N} a_n x^n$$

$$P_N = \sum_{j=0}^{N} Y_j L_j(x)$$

Indeed, $P_N(x_i) = Y_j$, $j = 0, 1, \cdots, N$

$$L_j(x) = \prod_{\substack{i=0 \\ i\neq j}}^{N} \frac{(x-x_i)}{(x_j-x_i)}$$

EX:

x_j	Y_j
2	6
3	11
4	18

$$P_2(x) = \frac{6(x-3)(x-4)}{(2-3)(2-4)} + \frac{11(x-2)(x-4)}{(3-2)(3-4)} + \frac{18(x-2)(x-3)}{(4-2)(4-3)}$$

Algorithm:

```
Interpolate(x)
  P=0
  For j=0 to N
      P1 = P + Yj * LIP(j,x)
  Next j
  return P.
```

work: $N\times$ to get P

$N\times$ to get each L

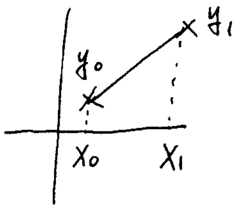
overall work $= O(N^2)$

$\propto N^2$ operations.

```
LIP(j,x)
L=1
For i=0 to j-1
   L = (x-xi)/(xj - xi) * L
Next i.
For i = j+1 to N
   L = (x-xi)/(xj-xi) * L
```

Newton:

point slope form



$$P_1(x) = y_0 + m(x - x_0)$$

$$y[x_0, x_1] = \frac{y_1 - y_0}{x_1 - x_0}$$

$$m = \frac{y_1 - y_0}{x_1 - x_0} \equiv y[x_0, x_1]$$

↑ called the first divided difference

$$P_1(x) = y_0 + y[x_0, x_1](x - x_0)$$

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

$$P_1(x) = y_0 + \frac{y_1 - y_0}{\cancel{y}\, x_1 - x_0}(x - x_0)$$

We get higher order interpolations recursively.

DEF: $\quad y[x_0, x_1, \cdots, x_N] \equiv \dfrac{y[x_1, x_2, \cdots, x_N] - y[x_0, x_1, x_2, \cdots, x_{N-1}]}{x_N - x_0}$

assume $\quad y[x_j] = y_j$

$$y[x_0, x_1, \cdots, x_N] = \frac{y[x_1, x_2, \cdots, x_N] - y[x_0, x_1, \cdots, x_{N-1}]}{x_N - x_0}$$

EX: $\quad x_0, x_1, x_2,$

$$y[x_0, x_1] = \frac{y_1 - y_0}{x_1 - x_0}$$

$$y[x_1, x_2] = \frac{y_2 - y_1}{x_2 - x_1}$$

$$y[x_0, x_1, x_2] = \frac{\frac{y_2 - y_1}{x_2 - x_1} - \frac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_0}$$

And we define the Newton form interpolant

$$P_N(x) = P_{N-1} + y[x_0, x_1, \cdots x_N] \prod_{i=0}^{N-1} (x - x_i)$$

$$P_N(x) = P_{N-1} + y[x_0, x_1, \cdots, x_N] \prod_{i=0}^{N-1} (x - x_i)$$

$$P_N(x) = P_{N-1} + y[x_0, \cdots, x_N] \prod_{i=1}^{N-1} (x - x_i)$$

Two steps:

(1) Compute divided differences

(2) Evaluate $P_N(x)$

EX:

j	x_j	y_j	$y[x_j, x_{j-1}]$	$y[x_0, x_1, x_2]$
0	2	6	$\frac{11-6}{3-2} = 5$	$\frac{7-5}{4-2} = 1$
1	3	11		
2	4	18	$\frac{18-11}{4-3} = 7$	

$$P_2(x) = 6 + 5(x-2) + 1\cdot(x-2)(x-3)$$

$P_2(2) = 6$

$P_2(3) = 11$

$P_2(4) = 6 + 10 + 2 = 18$

$$P_N(x) = y_0 + y[x_0, x_1](x-x_0) + y[x_0, x_1, x_2](x-x_0)(x-x_1) + \cdots + y[x_0, x_1, \cdots, x_N] \prod_{i=0}^{N-1} (x - x_i)$$

Remarks:

(1) Order of the points is inrelavant (irrelevant).

(2) Note:

$$P_N = y_0 + y[x_0, x_1](x-x_0) + y[x_0, x_1, x_2](x-x_0)(x-x_1) + \cdots + y[x_0, x_1, \cdots, x_N] \prod_{i=0}^{N-1} (x - x_i)$$

Taylor Polynomial

$$f_N(x) = y(x_0) + y'(x_0)(x-x_0) + \frac{y''(x_0)}{2!}(x-x_0)^2 + \cdots + \frac{y^{(N)}(x_0)}{N!}(x-x_0)^N$$

Thus

$$P_N(x) = y(x_0) + y'(x_0)(x-x_0) + \frac{y''(x_0)}{2!}(x-x_0)^2 + \cdots + \frac{y^{(N)}(x_0)}{N!}(x-x_0)^N$$

(3) Recall $y[x_0, x_1] \equiv \dfrac{y(x_1) - y(x_0)}{x_1 - x_0} = y'(\xi)$ ✓ MVT.

and so

($) $$y[x_0, x_1, \cdots, x_N] = \frac{y^{(N)}(\xi)}{N!}$$ ✓

To compute,

$$P_N(x) = \sum_{i=0}^{N} d_i \, w_i(x)$$

$$w_i(x) = \prod_{j=0}^{i-1} (x - x_j)$$

$d_i \rightarrow$ divided factor

differ ence table $(\vec{x}, \vec{y})$

```
{
  For i = 0 to N
      di = yi
  Next i
  For j = 1 to N
      For i = N to j step -1
          di = (di - di-1) / (xi - xi-j)
      Next i
  Next j
  Return d
}
```

Evaluate (x)

```
{
  P = ~~d[0]~~ d0
  W = 1
  For i = 1 to N
      w = w * (x - xi-1)
      P = P + di * W
  Next i
  Return P
}
```

$\begin{cases} \text{Newton} \\ \text{Lagrange} \end{cases}$

How well does P_N approximate $y(x)$?

THM: Let $\{x_i\}_{i=0}^{N}$ be distinct on an interval $[a,b]$ and let $y \in C^{N+1}[a,b]$,

for each $x \in [a,b]$, $\exists\, \xi \in [a,b] \Rightarrow y(x) = P_N(x) + \underbrace{\dfrac{y^{N+1}(\xi)}{(N+1)!} \prod_{j=0}^{N} (x - x_j)}_{E_N(x)}$

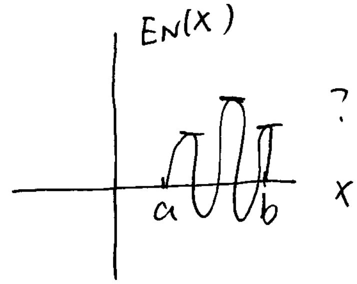
Note: The interpolation error $\quad y(x) = P_N(x) + \underbrace{\dfrac{y^{(N+1)}(\xi)}{(N+1)!} \prod_{j=0}^{N}(x-x_j)}_{E_N(x)}$

(1) $E_N(x_i) = 0$, $i = 0, 1, \cdots, N$

(2) If y is a polynomial of degree $\le N$, $E_N(x) = 0$.

(3) $E_N(x)$ is a polynomial of degree $N+1 \Rightarrow E_N$ has N extrema

$\Rightarrow E_N$ is very oscillatory.



Example:

$$y(x) = \frac{1}{1+25x^2}$$

E_N is oscillatory

$$\max_x |E_n| \approx \max_x |y| \qquad \max_x |E_n| \approx \max_x |y|$$

(gets worse as $N \to \infty$)

Note also max. error is near end points (commonly called "Runge PHENOMENA")

Consequences:

(1) best to evaluate only near the center of the interval.

(2) extrapolation is bad.

Fixes?

Yes, one thing we can do is to choose a better set of points.

we will use piecewise lower order polynomials.

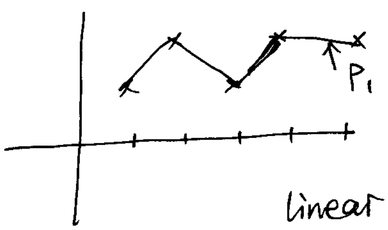
2.3.4. Spline Interpolation

Definition: Let $\{X_i\}_{i=0}^{N}$ be order and the elements distinct. The function S^k is a polynomial spline function of degree k if

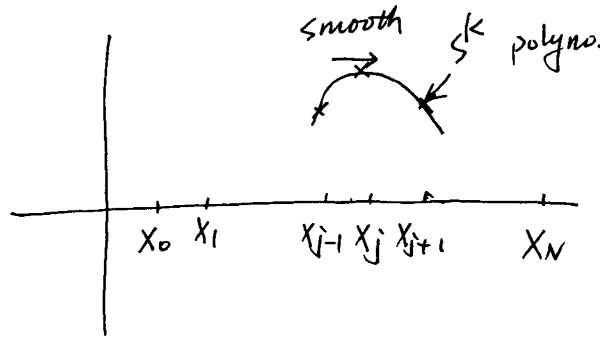
(1) on each subinterval $[X_i, X_{i+1}]$, S^k is a polynomial of degree $\le k$

(2) S has $(k-1)$ continuous derivatives on the interval $[X_0, X_N]$

EX: piecewise linear interpolation



linear spline



S^3 cubic spline

Most common spline is S^3 (cubic spline)

i.e. S^3 is cubic on $[X_i, X_{i+1}]$

S', S'' are continuous at the x_i's.

Derivation:

Note S'' is piecewise linear $\quad (X_i, M_i)\ (X_{i+1}, M_{i+1})$

$$S_i''(x) = M_i \frac{x - X_{i+1}}{X_i - X_{i+1}} + M_{i+1} \cdot \frac{x - X_i}{X_{i+1} - X_i}$$

S'' on $[X_i, X_{i+1}]$

Let $\Delta X_i = X_{i+1} - X_i$

So $S_i'' = \dfrac{M_i (X_{i+1} - X) + }{\Delta X_i} M_{i+1} \cdot \dfrac{X - X_i}{\Delta X_i} = M_i \dfrac{X_{i+1} - X}{\Delta X_i} + M_{i+1} \cdot \dfrac{X - X_i}{\Delta X_i}$

$$S_i'' = M_i \frac{X_{i+1} - X}{\Delta X_i} + M_{i+1} \frac{X - X_i}{\Delta X_i}$$

$\$ \quad M_i = S_i''(X_i) \approx y''(X_i)$

$M_i = S_i''(X_i) \approx y''(X_i)$

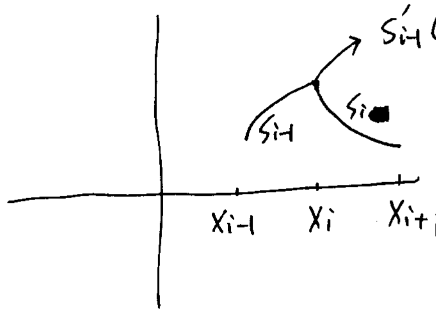
So, interpolate 2x.

$$S_i(x) = \frac{M_i}{6}\cdot\frac{(x_{i+1}-x)^3}{\Delta x_i} + \frac{M_{i+1}}{6\Delta x_i}(x-x_i)^3 + c(x-x_i) + D(x_{i+1}-x)$$

M_i, M_{i+1}, C, D. are unknown.

so to find M_i, M_{i+1}, C, D, from $\boxed{S_i(x_i) = y_i = y(x_i)}$ $i = 0, 1, \cdots, N-1$

smoothness

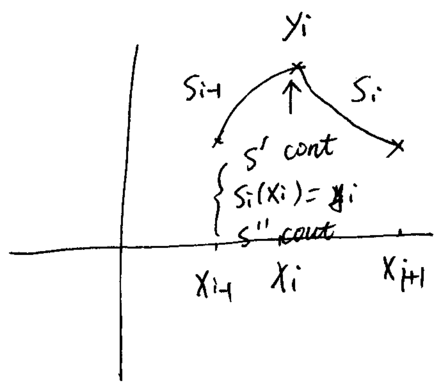


$S'_{i-1}(x_i) = S'_i(x_i)$

$S''_{i-1}(x_i) = S''_i(x_i)$

$$S_i(x) = \frac{M_i}{6}\cdot\frac{(x_{i+1}-x)^3}{\Delta x_i} + \frac{M_{i+1}}{6}\cdot\frac{(x-x_i)^3}{\Delta x_i} + c(x-x_i) + D(x_{i+1}-x)$$

2.8.



$$S_i(x) = \frac{M_i}{6\Delta x_i}(x_{i+1}-x)^3 + \frac{M_{i+1}}{6\Delta x_i}(x-x_i)^3 + C_i(x-x_i) + D(x_{i+1}-x)$$

$M_i \approx y''(x_i)$

$M_i \approx y''(x_i)$

$$S_i(x_{i+1}) = \boxed{y_{i+1} = \frac{M_{i+1}}{6\cancel{\Delta x_i}}\Delta x_i^2 + C\cdot\Delta x_i}$$

$$\Rightarrow C = \frac{1}{\Delta x_i}\left\{ y_{i+1} - \frac{M_{i+1}}{6}\Delta x_i^2 \right\}$$

$S_i(x_i) = y_i$ Gives D.

$$D = \frac{1}{\Delta x_i}\left\{ y_i - \frac{M_i}{6}\Delta x_i^2 \right\}$$

$$S_i(x_{i+1}) = y_{i+1} = \frac{M_{i+1}}{6}\Delta x_i^2 + C\cdot\Delta x_i$$

$$\Rightarrow C = \frac{1}{\Delta x_i}\left\{ y_{i+1} - \frac{M_{i+1}}{6}\Delta x_i^2 \right\}$$

$$S_i(x_i) = y_i = \frac{M_i}{6}\Delta x_i^2 + D\,\Delta x_i$$

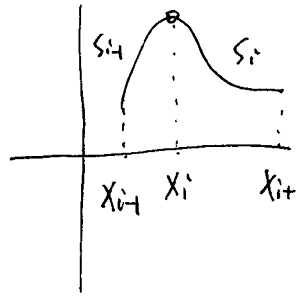
$$\Rightarrow D = \left\{ y_i - \frac{M_i}{6}\Delta x_i^2 \right\}\cdot\frac{1}{\Delta x_i}$$

$$S_i(x) = \frac{M_i}{6\Delta x_i}(x_{i+1}-x)^3 + \frac{M_{i+1}}{6\Delta x_i}(x-x_i)^3 + \frac{1}{\Delta x_i}\left(y_{i+1} - \frac{M_{i+1}}{6}\Delta x_i^2\right)(x-x_i)$$

$$+ \frac{1}{\Delta x_i}\left(y_i - \frac{M_i}{6}\Delta x_i^2\right)(x_{i+1}-x)$$

Need M_i $\quad i = 0, 1, \cdots, N$

To get those use ~~conditio~~ continuity of S'



$$\begin{cases} S_i'(x_i) = -\dfrac{\Delta x_i M_i}{3} - \dfrac{\Delta x_i}{6} M_{i+1} + y[x_i, x_{i+1}] \\ S_{i-1}'(x) = \dfrac{\Delta x_{i-1}}{3} M_i + \dfrac{\Delta x_{i-1}}{6} M_{i-1} + y[x_{i-1}, x_i] \end{cases}$$

Equate

$$\begin{cases} \Delta x_{i-1} M_{i-1} + 2(\Delta x_i + \Delta x_{i-1}) M_i + \Delta x_i M_{i+1} \\ = 6\{ y[x_i, x_{i+1}] - y[x_{i-1}, x_i] \} \end{cases} \qquad i = 1, 2, \cdots, N-1$$

$$\left(y[x_i, x_{i+1}] = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \right) \quad i = 1, 2, \cdots, N-1$$

$$y[x_0, x_1, \cdots, x_N] = \frac{y[x_1, x_2, \cdots, x_N] - y[x_0, x_1, \cdots, x_{N-1}]}{x_N - x_0}$$

Aside: Divide by $\Delta x_{i-1} + \Delta x_{i+1} = x_{i+1} - x_{i-1}$

The right side is

$$6\left(\frac{y[x_i, x_{i+1}] - y[x_{i-1}, x_i]}{x_{i+1} - x_{i-1}} \right) = 6\, y[x_{i-1}, x_i, x_{i+1}] = y''(\xi)$$

$$\xi \in [x_{i-1}, x_i]$$

we have $N+1$ unknowns, $N-1$ equations

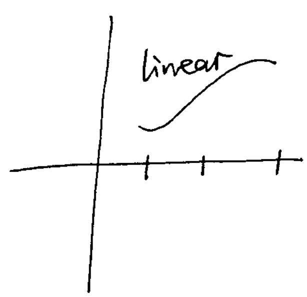
Need 2 more equations

Possibilities:

(1) $M_0 = y''(x_0)$, $M_N = y''(x_N)$ If we know y.

(2) $M_0 = 0$, $M_N = 0$

natural cubic spline.



$M_0 = 0$

$$\begin{bmatrix} 2(\Delta x_1 + \Delta x_0) & \Delta x_1 & 0 & \cdots & 0 \\ \Delta x_1 & 2(\Delta x_2 + \Delta x_1) & \Delta x_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \Delta x_{N-1} \\ 0 & 0 & 0 \cdots \Delta x_{N-2} & & 2(\Delta x_{N-1}, \Delta x_N) \end{bmatrix} \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{N-1} \end{bmatrix} = \begin{bmatrix} R_1 \\ R_2 \\ \vdots \\ R_{N-1} \end{bmatrix}$$

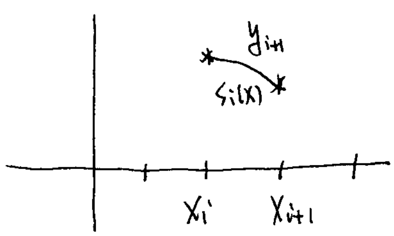
tridiatonal matrix

let $A_i = 2(\Delta x_i + \Delta x_{i-1})$

$R_i = 6\{ y[x_i, x_{i+1}] - y[x_{i-1}, x_i] \}$

$$\begin{bmatrix} A_1 & \Delta x_1 & \cdots & 0 \\ \Delta x_1 & A_2 & \Delta x_2 & 0 \\ \ddots & \vdots & \ddots & \Delta x_{N-2} \\ 0 & \Delta x_{N-2} & & A_{N-1} \end{bmatrix} \vec{M} = \vec{R}$$

2.10



$$S_i(x) = \frac{M_i}{6\Delta x_i}(x_{i+1} - x)^3 + \frac{M_{i+1}}{6\Delta x_i}(x - x_i)^3$$
$$+ \frac{1}{\Delta x_i}\left\{ y_{i+1} - \frac{M_{i+1}}{6}\Delta x_i^2 \right\}(x - x_i)$$
$$+ \frac{1}{\Delta x_i}\left\{ y_i - \frac{\cancel{M_i}}{6}\Delta x_i^2 \right\}(x_{i+1} - x) \qquad i = 0, 1, \cdots, N-$$

The M_i's are the solution of $T\vec{M} = \vec{R}$

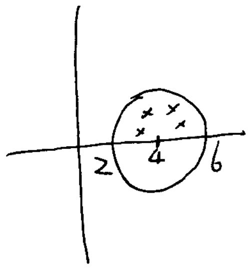
where $T \equiv \begin{bmatrix} A_1 & \Delta x_1 & 0 \\ \Delta x_1 & \ddots & \Delta x_{N-2} \\ 0 & \Delta x_{N-2} & A_{N-2} \end{bmatrix}$ $\quad A_i = 2(\Delta x_i + \Delta x_{i-1})$

$R_i = 6\{ y[x_i, x_{i+1}] - y[x_{i-1}, x_i] \}$

This is well-conditioned.

EX: $\Delta x_i = \Delta x = \text{constant}$ $\quad T = \Delta x \begin{bmatrix} 4 & 1 & 0 \\ 1 & \ddots & \end{bmatrix}$

EV's (evaluations) are ~~not~~ inside the circles with centers at $x=4$ and radii 2 gerschgorin



$$K(A) = \frac{\text{Max EV}}{\text{Min EV}} = \|A\|_2 \, \|A^{-1}\|_2$$

if A is symmetric

$$K(A) = \frac{\text{max EV}}{\text{min EV}} = \|A\|_2 \, \|A^{-1}\|_2 \quad \text{if A symmetric}$$

$2 \le EV \le 6$

$\Rightarrow K \le \frac{6}{2} = 3$

In S.P and D.P

This is well-conditioned

Solution of TRI-DIAGNAL Systems (diaganal)

$$T = \begin{bmatrix} d_1 & u_1 & 0 & \cdots & 0 \\ l_1 & d_2 & u_2 & \ddots & \\ \vdots & l_2 & \ddots & \ddots & u_{n-1} \\ 0 & 0 & \cdots & l_{n-1} & d_n \end{bmatrix}$$

Store 3 arrays only.

$d_i \Rightarrow i = 1, 2, \cdots, n$ store 0 to n.

$u_i \Rightarrow i = 1, 2, \cdots, n-1$

$l_i \Rightarrow i = 1, 2, \cdots, n-1$

$T\vec{X} = \vec{y}$

We use a variant of guess elimination called the <u>Thomas Algorithm</u>.

$$\begin{bmatrix} d_1 & u_1 & & 0 \\ l_1 & d_2 & u_2 & \\ & l_2 & \ddots & u_{n-1} \\ 0 & & l_{n-1} & d_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \qquad \begin{bmatrix} \hat{d}_1 & u_1 & & \\ & \hat{d}_2 & u_2 & \\ & & \ddots & u_{n-1} \\ & & & \hat{d}_n \end{bmatrix} \vec{X} = \hat{\vec{y}}$$

Forward elimination for $i = 2$ to n

$d_i = d_i - u_{i-1} * l_{i-1} / d_{i-1}$ ✡

$y_i = y_i - y_{i-1} * l_{i-1} / d_{i-1}$ ✡

next i

$$X_n = \frac{y_n}{d_n}$$

For $i = n-1$ to 1 step -1

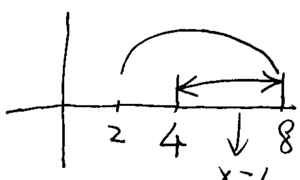
$X_i = (y_i - u_i * X_{i+1}) / d_i$

next i

backward substitution

Note: The diagnal and right-hand side are destoryed. Save before hand if need again.

To evaluate the spline, must fined the appropriate interval $[X_i, X_{i-1}]$



$K = \frac{N}{2}$

Is $X_K > X$?

Use binary (Bisection)

THM: Let $y \in C^4$ on interval $[a, b]$, partition $[a, b]$ into subintervals of width ΔX_i $\ni$(such that) $\underbrace{\Delta X \equiv \max_i \Delta X_i}$ and $\underbrace{\beta = \frac{\Delta X}{\Delta X_{min}}}$.

Let s be the cubic spline on $[a, b]$, with endpoints specified with the exactly values. Then

$$\| y^{(r)} - s^{(r)} \|_\infty \le C_r \, \Delta X^{4-r} \, \| y^{(4)} \|_\infty$$

$$C_0 = \frac{5}{384}, \quad C_1 = \frac{1}{24}, \quad C_2 = \frac{3}{8}, \quad C_3 = \frac{1}{2}\beta + \frac{2}{\beta}$$

$$s_i(x) = \frac{M_i}{6\Delta X_i}(X_{i+1} - X)^3 + \frac{M_{i+1}}{6\Delta X_i}(X - X_i)^3 + \frac{1}{\Delta X_i}\left(y_{i+1} - \frac{M_{i+1}}{6}\Delta X_i^2\right)(X - X_i) + \frac{1}{\Delta X_i}\left(y_i - \frac{M_i}{6}\Delta X_i^2\right)(X_{i+1} - X)$$

2.12 Comments on least squares approximations

A common approximation is linear least squares: for

$P_M(x) = \sum_{k=0}^{M} a_k \cdot x^k$ $\xrightarrow{}$ coefficiency $\qquad P_M(x) = \sum_{k=0}^{M} a^k x^k$

least squares: Find $\{a_n\}_{n=0}^{m} \ni \sum_{i=0}^{N} (y_i - p(x_i))^2$ is minimum.

for a set of data $\{(x_i y_i)\}_{i=0}^{N}$

EX: ~~linear~~ regression:

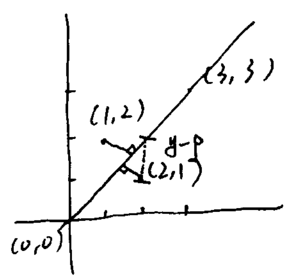
Given $\{(x_i, y_i)\}_{i=0}^{N}$ find a & b. $\ni$

$\sum_{i=0}^{N} (y_i - (a + bx_i))^2$ is minimum.

ie. $\nabla_{(a,b)} Q = 0$

Gives $\begin{bmatrix} N & \sum_{i=0}^{N} x_i \\ \sum_{i=0}^{N} x_i & \sum_{i=0}^{N} x_i^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{N} y_i \\ \sum_{i=0}^{N} x_i y_i \end{bmatrix}$ $\qquad \Phi \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \sum y_i \\ \sum x_i y_i \end{bmatrix}$

EX:



$N = 4$

$\sum x_i = 6$

$\sum x_i^2 = 14$

$\sum y_i = 6$

$\sum x_i y_i = 13$

$\begin{bmatrix} 4 & 6 \\ 6 & 14 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 6 \\ 13 \end{bmatrix}$

$\begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{4 \times 14 - 6 \times 6} \begin{bmatrix} 14 & -6 \\ -6 & 4 \end{bmatrix} \begin{bmatrix} 6 \\ 1 \end{bmatrix}$

$\Rightarrow P_1(x) = 0.3 + 0.8x$

General Case:

$p(x) = \sum_{k=0}^{M} a_k \phi_k(x)$ $\quad$ eg. $\Phi_k(x) = x^k$

Least Square: find $\vec{a} \ni \sum_i \left(y_i - \sum a_k \phi_k(x_i)\right)^2$ is minimum

$Q = \sum (y_i - \sum a_k \phi_k(x_i))^2$

$$\cancel{\nabla_{\vec{a}} Q = -2\sum_i}$$

$$\nabla_{\vec{a}} Q = \frac{\partial Q}{\partial a_L} = 2\sum_i (y_i - \sum a_K \phi_K(x_i))\, \phi_L = 0 \qquad \frac{\partial Q}{\partial a_L} = 2\left(\sum (y_i \phi_L - \phi_L \sum a_K \phi_K(x_i))\right) = 0$$

$$-\sum_i \sum_K a_K \phi_K(x_i) \phi_L(x_i) + \sum_i y_i \phi_L(x_i) = 0 \qquad -\sum_i \sum_K a_K \phi_K(x_i) + \sum_i y_i \phi_i(x_i) = 0$$

$$\Rightarrow \sum_{k=0}^{m} a_K \left(\sum_{i=0}^{N} \phi_K(x_i) \phi_L(x_i) \right) = \sum_{i=0}^{N} y_i \phi_L(x_i) \qquad \cancel{\frac{\partial Q}{\partial a_L} = \sum y_i}$$

$$\downarrow$$

$$\phi_{KL} \qquad \phi_{KL} = \sum_{i=0}^{N} \phi_K(x_i) \phi_L(x_i)$$

ie. $\Phi \vec{a} = \vec{R}$

$$\Phi_{KL} = \sum_{i=0}^{N} \phi_K(x_i) \phi_L(x_i)$$

$$\vec{R}_L = \sum_i y_i \phi_L(x_i)$$

$$\Phi \vec{a} = \vec{R}$$

$$\begin{cases} \Phi_{KL} = \sum \phi_K(x_i) \Sigma_L(x_i) \\ \vec{R}_L = \sum y_i \phi_L(x_i) \end{cases}$$

$\Phi_K(x) = x^k$

For $x \in [0,1]$, $\phi_k = x^k$, $m = 9$, $x_i = i \cdot \Delta x$, $\Delta x = \frac{1}{m}$, $K(\Phi) \approx 10^{12}$

Don't use $x^k = \Phi_k$. Instead, use orthogonal polynomials, eg. legecendre)

2.15

2.4 Quadrature

Most integers can only be approximated. EX: the pdf.

$$p(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-x^2} dx$$

Two basic classes of integration of approximations.

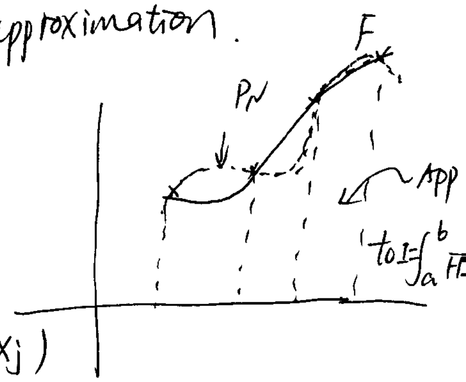
(1) ~~Interpol~~ Interpolatory quarature.

(2) monte Carlo

state w/ #1

11) Basic idea: approximate

The integrand by a polynomial & integrate the approximation.



formal derivation:

let $\{x_i\}_{i=0}^{N}$ be some points in $[a,b]$, then

$$F(x) = \underset{\downarrow \text{error}}{P_N(x) + E_N(x)} = \sum_{j=0}^{N} F(x_j)\, l_j(x) + \frac{F^{(N+1)}_{(\xi)}}{(N+1)!} \prod_{j=0}^{N}(x-x_j)$$

$F \to$ function

$P_N \to$ approx. to F.

$$I = \int_a^b F(x)\,dx = \underbrace{\int_a^b \sum_j F_j\, l_j(x)\,dx}_{\tilde{I}} + \underbrace{\int_a^b \frac{F^{(N+1)}_{(\xi)}}{(N+1)!} \prod_{j=0}^{N}(x-x_j)}_{\text{Error}}$$

$$\tilde{I} = \sum_{j=0}^{N} F_j \cdot \int_a^b l_j(x)\,dx = \sum_{j=0}^{N} F_j \cdot w_j$$

$w_j = \int_a^b l_j(x)\,dx =$ Quadrature weights

Different methods are found with different choices of $\{x_j\}_{j=0}^{N}$

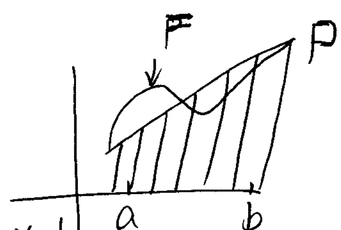
We'll study methods where the x_j's are uniformly spaced, called

Newton-Cotes Methods.

They are called closed if a & b are nodes, open if a & b aren't nodes.

Examples:

1) Trapezoidal Rule



write P_1 in newton form, $x_0=a$, $x_1=b$

$$P_1(x) = (x-a)F[a,b] + F(a) = F(a) + (x-a)\frac{F(b)-F(a)}{b-a}$$

$$\tilde{I} = \int_a^b P_1(x)\,dx = \int_a^b F(a)\,dx + F[a,b]\int_a^b (x-a)\cdot dx$$

$$= F(a)(b-a) + F[a,b]\ \frac{(x-a)^2}{2}\Big|_a^b = F(a)(b-a) + F[a,b]\cdot\frac{(b-a)^2}{2}$$

let $\Delta X = b-a$

$$\tilde{I} = F(a)\Delta X + \frac{\Delta X^2}{2}\cdot\frac{F(b)-F(a)}{b-a} = F(a)\Delta X + \frac{(F(b)-F(a))\Delta X}{2} = \frac{\Delta X}{2}\cdot\left(F(b)+F(a)\right)$$

$$\tilde{I}_{trap} = \Delta X\cdot\frac{F(b)+F(a)}{2}$$

ERROR?

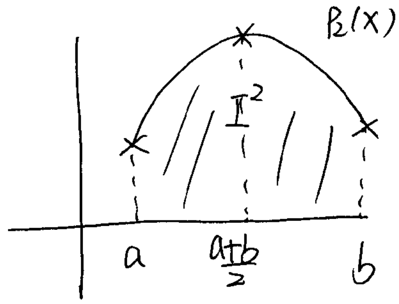
$$E = \underbrace{\int_a^b F''(x-a)(x-b)\,dx = F''(\xi)\int_a^b (x-a)(x-b)\,dx}$$

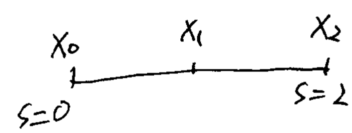
change variables: $\begin{cases} x = a + s\cdot\Delta X \qquad s=0 \Leftrightarrow x=a, \quad s=1 \Leftrightarrow x=b \\ x-a = s\cdot\Delta X \\ x-b = a+s\Delta X - b = -\Delta X + s\Delta X = (s-1)\Delta X \end{cases}$

$$E = \Delta X^3 F''(\xi)\int_0^1 s(s-1)\,ds = \frac{\Delta X^3 F''(\xi)}{2}\left[\underbrace{\frac{s^3}{3} - \frac{s^2}{2}}_{-\frac{1}{6}}\Big|_0^1\right]$$

$$E_{trap} = -\frac{\Delta X^3 F''(\xi)}{12}$$

2) Simpson's Rule





$x = a + s\cdot\Delta X$

$\Delta X = \frac{x-a}{2}$

$x - x_0 = x - a = s\cdot\Delta X$

$(x-x_0)(x-x_1) = s(s-1)\Delta X^2$

$$\tilde{I} = \int_0^2\left(F_0 + F[x_0,x_1]\cdot s\Delta X + F[x_0,x_1,x_2]\,s(s-1)\Delta X^2\right)ds\,\Delta X$$

$$= F_0\Delta X\int_0^2 ds + F[x_0,x_1]\Delta X^2\int_0^2 s\,ds + F[x_0,x_1,x_2]\Delta X^3\int_0^2 s(s-1)\,ds$$

??? $\begin{cases}\tilde{I} = 2F_0\Delta X + 2F[x_0,x_1]\Delta X^2 + F[x_0,x_1,x_2]\cancel{\ldots}(+\frac{2}{3}) \\ = \frac{2\Delta X}{6}\{F_0 + 4F_1 + F_2\} = \frac{b-a}{6}\{F(a) + 4F(\frac{a+b}{2}) + F(b)\}\end{cases}$

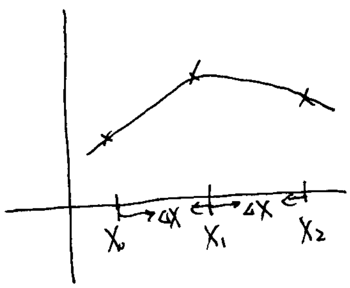
EX:

$$\tilde{I}_{simpson} = \frac{b-a}{6}\left\{F(a) + 4F\left(\frac{a+b}{2}\right) + F(b)\right\}$$

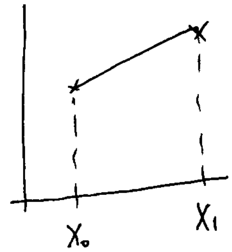
$$E = \frac{\Delta X^4}{3!}\int_0^2 F'' s(s-1)(s-2)\,ds \overset{?}{\underset{\downarrow}{=}} \frac{\Delta X^4}{3!}F''(\xi)\underbrace{\int_0^2 s(s-1)(s-2)\,ds} \quad (r = s-1)$$

Let $r = s-1$, $s = r+1$, $s-2 = r-1$, $\Rightarrow \int_{-1}^{1} (r+1)r(r-1)\,dr \equiv 0$

2.17



Simpson



TRAP

$$\int_{X_0}^{X_1} F(X)\,dX = \frac{\Delta X}{2}(F_0 + F_1) - \frac{\Delta X^3}{12} F''(\xi)$$

$$F_0 = F(X_0)$$

$$\int_{X_1}^{X_2} F(X)\,dX = \frac{\Delta X}{3}(F_0 + 4F_1 + F_2) + E$$

$$E = ?$$

Find Simpson Error.

$$\tilde{I} = \frac{\Delta X}{3}(F_0 + 4F_1 + F_2) = \frac{\Delta X}{3}\Big(F_1 + (-\Delta X F_1') + \frac{\Delta X^2}{2} F_1'' + \cdots$$

Tayler Series $\rightarrow F_0$ in terms of F_1

$$+ 4F_1$$

$\rightarrow 4F_1$

$$+ F_1 + \Delta X F_1' + \frac{\Delta X^2}{2} F_1'' + \cdots$$

$\rightarrow F_2$ in terms of F_1

$$\tilde{I} = \frac{\Delta X}{3}\left(6F_1 + \Delta X^2 F_1'' + 2 \cdot \frac{\Delta X^4}{4!} F_1^{(4)}(\xi)\right)$$

$$= 2\Delta X F_1 + \frac{\Delta X^3}{3} F_1'' + \frac{2\Delta X^5}{3 \times 4!} F_1^{(4)}(\xi)$$

$x_1 - x_0 = \Delta X = x_2 - x_1$

$$I = \int_{X_0}^{X_2} F(X)\,dX = \int_{X_0}^{X_2} \left(F_1 + (X - X_1)F_1' + \frac{(X-X_1)^2}{2} F_1'' + \cdots\right) dX$$

$$= 2\Delta X F_1 + \frac{(X-X_1)^2}{2} F_1' \Big|_{X_0}^{X_2} + \frac{(X-X_1)^3}{6} F_1'' \Big|_{X_0}^{X_2} + \cdots$$

$$= 2\Delta X F_1 + \frac{\Delta X^3}{3} F_1'' + \frac{\Delta X^5}{60} F_1^{(4)} + \cdots$$

so. $E = I - \tilde{I} = \Delta X^5 F_1^{(4)}\left(\frac{1}{60} - \frac{2}{3 \times 4!}\right) = -\frac{\Delta X^5}{90} F_1^{(4)}$

$$\int_{X_0}^{X_2} F(X)\,dX = \frac{\Delta X}{3}(F_0 + 4F_1 + F_2) - \frac{\Delta X^5}{90} F_1^{(4)}$$

Definition: (Precision)

A quadrature has precision P if it is exact for all polynomials of DEG. $\leq P$

Examples:

$$I=\int_0^1 e^x dx = e-1 = 1.7782818 = 1.7182818$$

TRAP Rule: $\tilde{I} = \frac{e^1+e^0}{2} = \frac{e+1}{2} = 1.8591409$

$$|E| = \left|\frac{\Delta x}{12} e^{x''}(\xi)\right| = \frac{\Delta x}{12} e^{\xi} \qquad \xi \in [0,1]$$

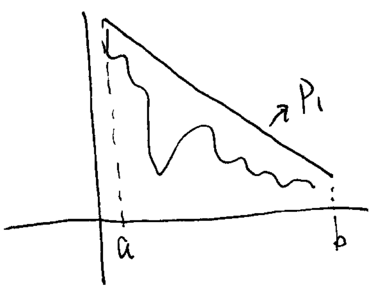
$$\begin{cases} \frac{1}{12}e^0 \le |E| \le \frac{1}{12}e^1 \\ 0.0833 \le |E| \le 0.2265 \end{cases} \qquad |E| = 0.140859$$

Simpson Rule:

$$\tilde{I} = \frac{1}{6}(e^0 + 4e^{0.5} + e^1) = 1.7188612$$

$|E| = 5.8\times 10^{-4}$ $\qquad (\frac{1}{2})^5 \frac{1}{90} \le |E| \le (\frac{1}{2})^5 \frac{e}{90}$

$3.5 < 10^{-4} < 5.8\times 10^{-4} < 9.4\times 10^{-4}$ ✓



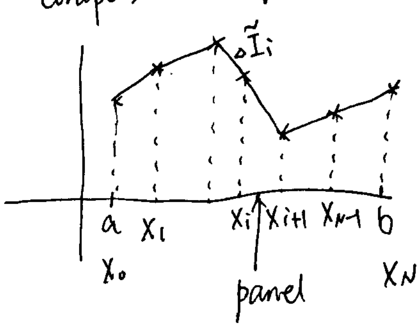
Bad error if
(1) $b-a$ is large
(2) $P^{(2)}(\xi)$ is large $\qquad$ p=presion

2.4.2 Composite Rules

If $[a,b]$ is large, and/or $F^{(p+1)}(\xi)$ is large. Newton-Cotes Rule are not accurat

fix: use piecewise polynomial interpolations.

Composite trapezoidal.



piecewise linear $\qquad \tilde{I} = \sum_{i=0}^{N-1} \tilde{I}_i$

Assume $x_i = a + i\Delta x \qquad \Delta x = \frac{b-a}{N}$ $\qquad F_i = F(x_i)$

$$\tilde{I}_i = \frac{\Delta x}{2}\left(F(x_i)+F(x_{i+1})\right)$$

$$\tilde{I} = \sum_{i=0}^{N-1} \tilde{I}_i = \frac{\Delta x}{2}\sum_{i=0}^{N-1}\left(F(x_i)+F(x_{i+1})\right)$$

$$\tilde{I} = \frac{\Delta x}{2}\left(F_0 + 2\sum_{i=1}^{N-1} F_i + F_N\right) \to \text{Composite Trap Rule}$$

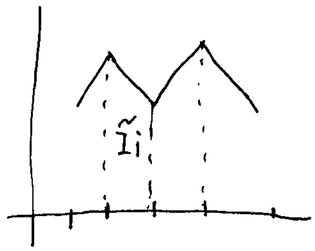
$$E = \sum_{i=0}^{N-1} E_i$$

$$E = -\frac{\Delta x^3}{12} \sum_{i=0}^{N-1} F''(\xi_i)$$

2.19

$$I = \int_a^b F(x)\,dx$$

$$\tilde{I} = \sum_{i=0}^{N-1} \tilde{I}_i$$



if the spacing Δx = constant

$$\tilde{I}_{trap} = \frac{\Delta x}{2} \left\{ F_0 + 2\sum_{j=1}^{N-1} F_j + F_N \right\}$$

Algorithm:

$S = 0$

$\Delta x = \frac{b-a}{N}$

For $j = 1$ to $N-1$

  $x = a + j\Delta x$

  $S = S + F(x)$

Next j

$\tilde{I} = \frac{\Delta x}{2} * (F(a) + 2*S + F(b))$

Error: $E = \sum_{i=0}^{N-1} E_i$ $\quad E_i = -\frac{\Delta x^3}{12} F''(\xi_i) \quad \xi_i \in [x_i, x_{i+1}]$

$$E = -\frac{\Delta x^3}{12} \sum_{i=0}^{N-1} F''(\xi_i)$$

Note: If F'' is continuous, $\min_{x\in[a,b]} F''(x) \leq F''(\xi_i) \leq \max_{x\in[a,b]} F''(x)$

Sum over i's

$$N \min F''(x) \leq \sum_{i=0}^{N-1} F''(\xi_i) \leq N \cdot \max F''(x)$$

or $\min F'' \leq \left(\frac{1}{N} \sum_{i=0}^{N-1} F''(\xi_i) \right) \leq \max F''$

$\Downarrow$

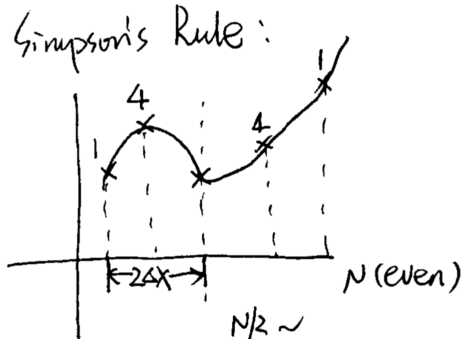
By intermediate value theorem, $\exists u \in [a,b]$, $\ni$ such that
($\exists$ → exist; $\ni$ → such that)

$$\frac{1}{N}\sum_{i=0}^{N-1} F''(\xi_i) = F''(u)$$

therefore, $E = -\frac{\Delta x^3}{12} N F''(u)$ $\qquad$ Δx → spacing between points.

but $N = \frac{b-a}{\Delta x}$ $\Rightarrow$ $E = -\frac{b-a}{12}\Delta x^2 F''(u)$

Simpson's Rule:



$$\tilde{I}_{simp} = \sum_{j=0}^{N/2} \tilde{I}_i$$

two intervals each panel

$$\tilde{I}_{simp} = \frac{\Delta x}{3}\left\{ F(a) + \sum_{j=0}^{N/2-1} F_{2j} + 4\sum_{j=1}^{N/2} F_{2j-1} + F(b) \right\}$$

Algorithm:

se = 0 $\qquad$ se → sum even

so = 0 $\qquad$ so → sum odd

$\Delta X = \frac{b-a}{N}$

For j = 1 to N-1 step +2
  X = a + j * ΔX
  so = so + F(X)
Next j
For j = 2 to N-N step 2
  X = a + j * ΔX
  se = se + F(X)
Next j.

$$\tilde{I} = \frac{\Delta X}{3}(F(a) + F(b) + 2*se + 4*so)$$

Error:

$$E_i = -\frac{\Delta x^{4+1}}{90}F^{(4)}(\xi_i) = -\frac{\Delta x^5}{90}F^{(4)}(\xi_i)$$

$$E = -\frac{b-a}{180}\Delta x^4 F^{(4)}(u)$$

2.4.3 Error Estimation & Adaptive Quadrature

$I = \tilde{I} + E$

$E = \underbrace{C}_{\text{constant}} \Delta x^r F^{(m)}(u)$ $\qquad r=2, m=2$ Composite trap.

$\qquad r=4, m=4$, Composite simpson.

For Δx (small), $\quad F^{(m)}(u) \approx$ constant.

so, as $\Delta x \to 0$, $\quad E \to \underbrace{c}_{\text{Generic}} \Delta x^r$

EX: Trap. Rule

$\frac{1}{N}\sum_{i=0}^{N-1} F''(\xi) = F''(u)$ $\qquad \Delta x = \frac{b-a}{N}$ $\qquad \Delta x \to 0$, $N \to \infty$

$\frac{1}{N} = \frac{\Delta x}{b-a}$, $\quad \frac{1}{N}\sum F''(\xi_i) = \frac{1}{b-a} \underbrace{\sum_{i=0}^{N-1} F''(\xi_i)\Delta x}$ $\qquad$ riemann sum

$$\lim_{N\to 0} \frac{1}{b-a}\sum_{i=0}^{N-1} F''(\xi_i)\Delta x = \int_a^b F''(\xi_i)\,d\xi \cdot \frac{1}{b-a}$$

$$= \frac{F'(b)-F'(a)}{b-a} = \text{constant}$$

As $\Delta x \to 0$ $\quad F''(u) \to$ constant.

~~Error ≈~~ $\quad I \approx \tilde{I} + \underbrace{C\Delta x^r}_{E\Delta x}$ $\qquad r, I$ unknow

One equation

$I \approx \tilde{I}_{\Delta x} + E\Delta x$

Redo with $\frac{\Delta x}{2}$.

$I \approx \tilde{I}_{\frac{\Delta x}{2}} + C\cdot\left(\frac{\Delta x}{2}\right)^r$

2.22. Error Estimation (Ricardson)

$$I \approx \tilde{I}_{\Delta x} + \underbrace{C\Delta x^n}_{E}$$

$n \to r$

$$I \approx \tilde{I}_{\frac{\Delta x}{2}} + C\left(\frac{\Delta x}{2}\right)^n$$

$$0 \approx (\tilde{I}_{\Delta x} - \tilde{I}_{\frac{\Delta x}{2}}) + \underbrace{C\Delta x^n}_{E_{\Delta x}}\left(1 - \frac{1}{2^n}\right)$$

$$E_{\Delta x} \doteq \frac{2^n}{2^n-1}(\tilde{I}_{\frac{\Delta x}{2}} - \tilde{I}_{\Delta x})$$

$F^{(m)}(u)$ doesn't change with Δx

$$E_{\frac{\Delta x}{2}} \doteq \frac{1}{2^n-1}(\tilde{I}_{\frac{\Delta x}{2}} - \tilde{I}_{\Delta x})$$

EX: $\int_0^1 \frac{1}{1+x}dx = \ln 2$ $n=8,\ n=16$

Trap. Rule: $I_8 = 0.694122$

$I_{16} = 0.693391$

$$|E_8| \doteq \left(\frac{2^2}{2^2-1}\right)|(0.693391 - 0.694122)| = 9.7467 \times 10^{-4}$$

Actual Error: find in the table 9.74670×10^{-4}

$$|E_{16}| = \frac{1}{4} \times 9.7467 \times 10^{-4} = 2.4366 \times 10^{-4}$$

Also, since $E \approx C\Delta x^r$, we can estimate Δx needed to get a desired error.

$$E_n = C\Delta x_n^r \qquad \Rightarrow \qquad \frac{E}{E_n} = \left(\frac{\Delta x}{\Delta x_n}\right)^r$$

$$E = C \cdot \Delta x^r = Tol$$

$$\left(\frac{Tol}{E_n}\right)^{\frac{1}{r}} \cdot \Delta x_n = \Delta x$$

so the new # of panels $N = \frac{b-a}{\Delta x}$

EX: with 32 intervals simpson error was 2.78×10^{-8}.

Find N so trap. rule matches.

$n=32$, $E_{32}(\text{trap}) = \frac{1}{32}\sqrt{\frac{2.78\times10^{-8}}{6.10\times10^{-5}}}$

$\Delta x = 6.7\times10^{-4}$

$N = 1499$

A simpson error control strategy.

1. compute $\tilde{I}_{\Delta x}$ with Δx
2. compute $\tilde{I}_{\frac{\Delta x}{2}}$ with $\frac{\Delta x}{2}$
3. estimate

$$E_{\frac{\Delta x}{2}} = \frac{1}{2^r-1}\left(\tilde{I}_{\frac{\Delta x}{2}} - \tilde{I}_{\Delta x}\right)$$

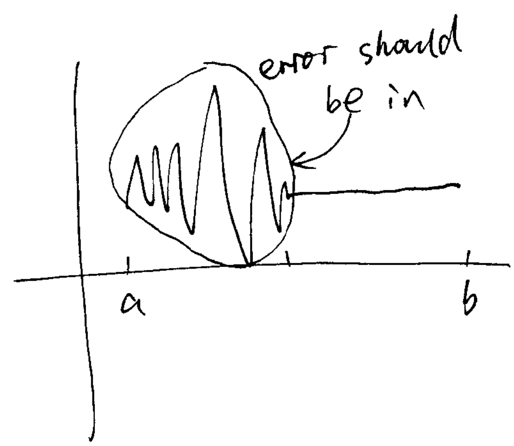
4. if $|E_{\frac{\Delta x}{2}}| \le abstol + |\tilde{I}_{\frac{\Delta x}{2}}| \cdot reltol$

   return $\tilde{I}_{\frac{\Delta x}{2}} + E_{\frac{\Delta x}{2}}$

5. repeat 1-5.

Suppose we have a function composite rule (F, a, b, N) → $\tilde{I}_N$; N → # of intervals

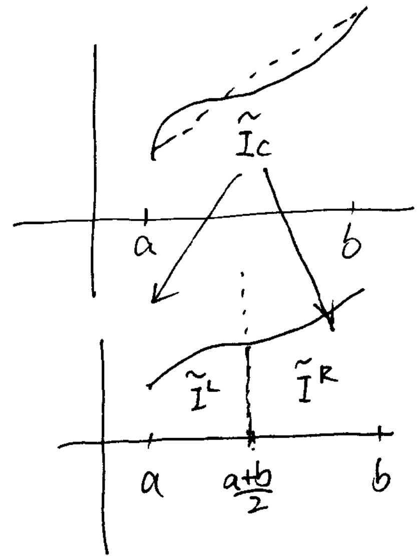
```
N=2
for k=1 to maxk.
    Ic = composite rule (F,a,b,N)
    If = compositerule (F,a,b,2N)
    E = (If - Ic)/(2^r - 1)
    if |E| ≤ abstol + |If| · reltol
        return If + E
    else
        N=2N
    endif
next k.
```



Adaptive Quadrature

$$E_c \equiv I - \tilde{I}_c = \frac{2^r}{2^r-1}(\tilde{I}_f - \tilde{I}_c)$$

if $|E_c| \le tol$?

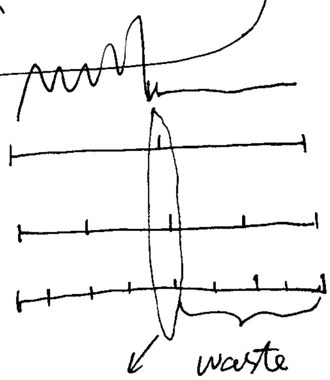


$\tilde{I}_f = \tilde{I}^L + \tilde{I}^R$

(2.24)

$$\max_{x\in[a,b]} |\underbrace{y^{(r)}(x) - s^{(r)}(x)}_{\text{error}}| \le C_r(\Delta x^{4-r})\,\|y^{(4)}\|_\infty$$
$$\le C\Delta x^{4-r}$$
$$\Delta x^3$$
$$E_s \sim \Delta x^4$$

$$I = \int_a^b F(x)\,dx$$



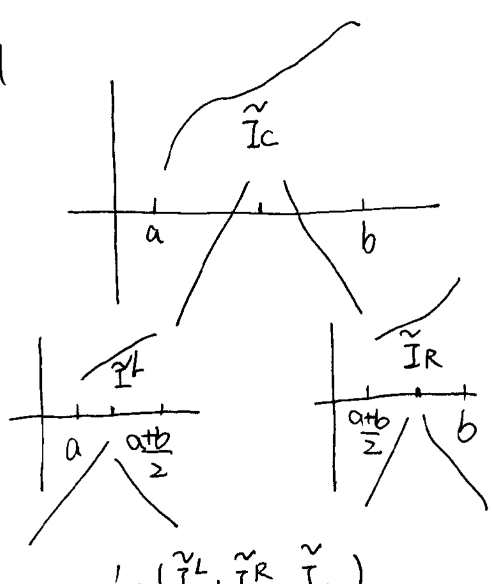
$n=2$, $\tilde{I}_2$

$n=4$, $\tilde{I}_4$

$E_4 \doteq \frac{1}{2^n-1}(\tilde{I}_4 - \tilde{I}_2)$ $\quad n=8$, $\tilde{I}_8$ & repeat

recalculate same point over and over.

Instead



$E_f \approx \frac{1}{2^r-1}(\tilde{I}^L + \tilde{I}^R - \tilde{I}_c)$

repeat

(top)
at each level, we test

$$\frac{1}{2^r}(\tilde{I}^L + \tilde{I}^R - \tilde{I}_c) \overset{?}{\le} Tol.$$

if yes, return $\tilde{I}^L + \tilde{I}^R + E_f$

if no, we split each half & redo.

on each half we allow $\frac{tol}{2}$ error.

so total is approximately

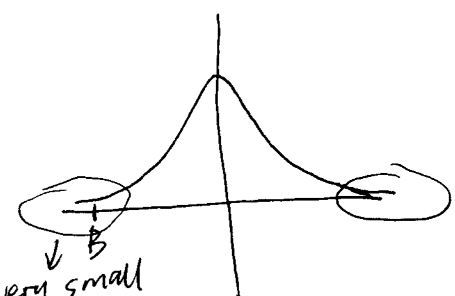
$$\frac{tol}{2} + \frac{tol}{2} = tol.$$

Assume we have a quardture function $Q(F, a, b)$

```
Adaptive_Q (F, a, b, abstol)
    Ic = Q(F, a, b)
    If = Q(F, a, (a+b)/2) + Q(F, (a+b)/2, b)
    E = (If - Ic)/(2^r - 1)
  if |E| > abstol
    IL = adaptive_Q(F, a, (a+b)/2, abstol/2)
    IR = adaptive_Q(F, (a+b)/2, b, abstol/2)
    return IL + IR
  else
    return If + E
  endif
```

2.4.4 Infinite intervals

E(X): $p(X) = \frac{1}{\sqrt{2\pi}} \int_0^X e^{-\frac{t^2}{2}} dt$



suppose $X > 0$.

$p(X) \geqslant \frac{1}{2}$

Need $\int_{-\infty}^{B} e^{-\frac{x^2}{2}} dx = \epsilon$

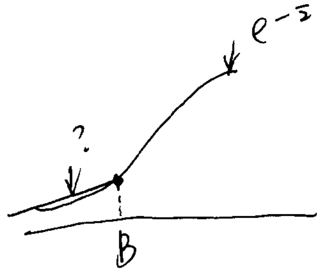
find B, so $\int_{-\infty}^{B} e^{-\frac{t^2}{2}} dt \ll \int_B^X e^{-\frac{t^2}{2}} dt$

$\downarrow$

can be ignored. B?

find B, so $\int_{-\infty}^{B} e^{-\frac{x^2}{2}} dx < \frac{\varepsilon}{2}$ ← machine epsilon

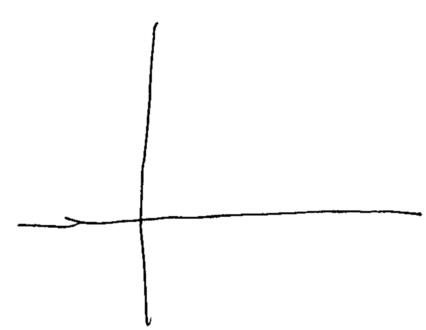
find an integral $\Rightarrow \int_{-\infty}^{B} e^{-\frac{t^2}{2}} dt \le \int_{-\infty}^{B} (\;) dt < \frac{\epsilon}{2}$



$e^{-\frac{B^2}{2}} e^{-\frac{x-B}{2}}$

$e^{-\frac{B^2}{2}} \int_{-\infty}^{B} e^{-\frac{(x-B)^2}{2}} dx = \frac{\epsilon}{2}$

$\Rightarrow 2e^{-B^2/2} = \frac{\epsilon}{2} \Rightarrow e^{-\frac{B^2}{2}} = \frac{\epsilon}{4}$



$$B = -\sqrt{-2\log\frac{\epsilon}{4}}$$

$B \approx -6$ for single precision.

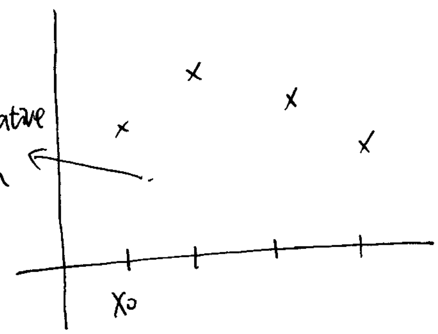
But: $\int_{-\infty}^{0} e^{-\frac{t^2}{2}}\,dt = \frac{1}{2}$

$$\frac{1}{\sqrt{2\pi}}\left(\frac{1}{2} + \int_0^x e^{-\frac{t^2}{2}}\,dt\right) = p(x)$$

2.26

2.6 Differentiation

approximate derivative at a point given discrete data.



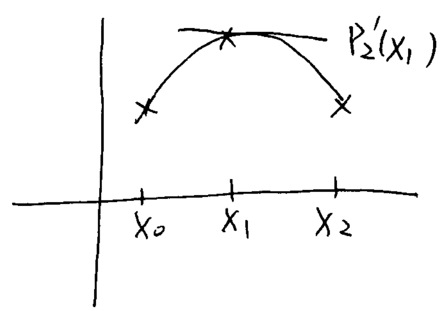
Idea. Approximate the function by an interpolant $M_0 \approx y''(x_0)$

and differentiate the interpolant.

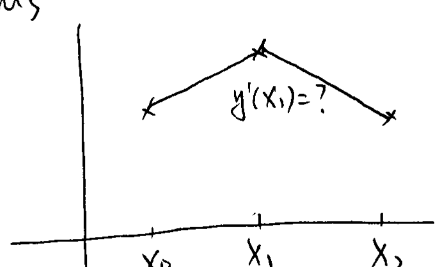
Given a set of point $\{y_i\}_{i=0}^{N}$ approx $y \approx P_N(x)$

$P_N(x_i) = y_i$, $y \approx P_N(x)$

$$\left.\frac{d^k y}{dx^k}\right|_{x_j} \doteq \left.\frac{d^k}{dx^k} P_N(x)\right|_{x_j}$$



EX: Three points



$$P_1(x) = y_0 + y[x_0, x_1](x - x_0)$$

$$P_1'(x_1) = y[x_0, x_1] = \frac{y_1 - y_0}{x_1 - x_0}$$

$$P_1(x) = y_1 + y[x_1, x_2](x - x_1)$$

$$P_1'(x) = y[x_1, x_2] = \frac{y_2 - y_1}{x_2 - x_1}$$

call $\delta^+_{y_j} = \dfrac{y_{j+1} - y_j}{\Delta x_j}$ first order forward derivative approximation

$\delta^-_{y_j} = \dfrac{y_j - y_{j-1}}{\Delta x_{j-1}}$ backward difference.

Quadratic

$$P_2(x) = y_0 + (x - x_0) F[x_0, x_1] + (x - x_0)(x - x_1) F[x_0, x_1, x_2]$$

$$P_2'(x) = F[x_0, x_1] + [(x - x_1) + (x - x_0)] F[x_0, x_1, x_2]$$

$$P_2'(x_1) = F[x_0, x_1] + \Delta x_0 F[x_0, x_1, x_2]$$

Assume: The x_j's are uniformly spaced.

$\Delta x_j = \Delta x =$ constant

$$P_2'(x_1) = \frac{y_1 - y_0}{\Delta x} + \Delta x \left[\frac{\frac{y_2 - y_1}{\Delta x} - \frac{y_1 - y_0}{\Delta x}}{2\Delta x} \right]$$

$$= \frac{y_1 - y_0}{\Delta x} + \frac{\Delta x}{2} \cdot \frac{y_2 - 2y_1 + y_0}{\Delta x^2}$$

$$= \frac{y_1 - y_0}{\Delta x} + \frac{y_2 - 2y_1 + y_0}{2\Delta x}$$

$$= \frac{2y_1 - 2y_0 + y_2 - 2y_1 + y_0}{2\Delta x} = \frac{y_2 - y_0}{2\Delta x}$$

Define:
$$\delta^{\circ} y_j = \frac{y_{j+1} - y_j}{2\Delta X}$$

centered operator and note $\overset{\circ}{\delta y_j} = (\delta y_j^{+} + \delta y_j^{-})/2$

Error ?
$$y(x) = P_N(X) + y^{(N+1)}_{(\xi)} / (N+1)! \prod_{i=0}^{N} (X - X_i)$$

$$y(X) = P_N(x) + \underbrace{\frac{y^{(N+1)}_{(\xi)}}{(N+1)!} \prod_{i=0}^{N} (X - X_i)}_{E_N(X)}$$

$$\frac{d^k y}{dx^k} = \frac{d^k P_N(x)}{dx^k} + \underbrace{\frac{d^k E_N(X)}{dX^k}}_{E_N^{(k)}(X)}$$

$$E_N^{(k)} = \frac{y^{(N+1)}_{(\xi)}}{(N+1)!} \frac{d^k}{dX^k} \prod_{i=0}^{N} (X - X_i)$$

for uniform spaced points, X_i ordered

Let $X = X_0 + S * \Delta X$ note : $\frac{d^k}{dx^k} = \frac{d^k}{ds^k} \cdot \frac{1}{\Delta X^k}$

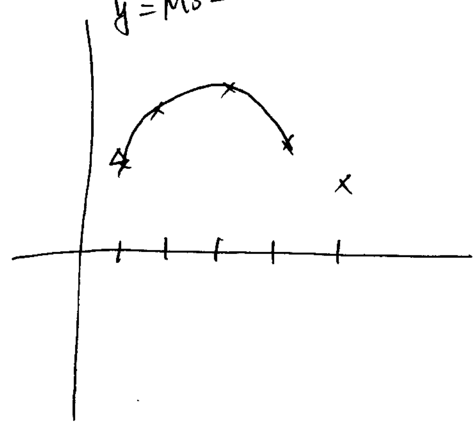
$X_i = X_0 + i * \Delta X$

$X - X_i = (S - i)\Delta X$

$$E_N^{(k)} = \frac{y^{(N+1)}_{(\xi)}}{(N+1)!} \cdot \frac{1}{\Delta X^k} \left(\frac{d^k}{ds^k} \prod_{i=0}^{N} (\cancel{s} - i) \right) \Delta X^{N+1}$$

$$= \left[\frac{y^{(N+1)}_{(\xi)}}{(N+1)!} \frac{d^k}{ds^k} \prod_{i=0}^{N} (S - i) \right] \Delta X^{N-K+1}$$

$y'' = M_0 = ?$



n derivative $\Rightarrow N - K + 1 = n$.

$N - K + 1 = 2$

$K = 2$

$N = 3$

Definition: $E_N^{(k)}$ is of order r if $E_N^{(k)} = O(\Delta x^n)$

In general, we use one of two ways to find error

(1) A-priori

use $E_N^{(k)} = \left[\frac{y^{(N+1)}(\xi)}{(N+1)!} \frac{d^k}{s^k} \prod_{i=0}^{N} (s-i) \right] \Delta x^{N-k+1}$

(2) A-posteriori

(3) Use Tayler's theorem.

EX. $\delta \overset{\circ}{y_j} = \frac{y_{j+1} - y_{j-1}}{2\Delta x} \approx y'(x_j)$

$$y_{j+1} = y_j + \Delta x y_j' + \frac{\Delta x^2}{2} y_j'' + \cdots$$

$$y_{j-1} = y_j - \Delta x y_j' + \frac{\Delta x^2}{2} y_j'' + \cdots$$

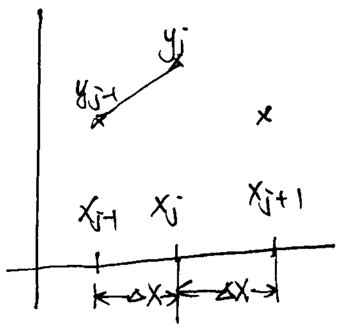
$$y_{j+1} - y_{j-1} = \left(2\Delta x y_j^{(1)} + \frac{2\Delta x^3 y_j^{(3)}}{3!} + \cdots \right) = 2\Delta x y_j' + 2\Delta x^3 \frac{y_j'''}{3!} + \cdots$$

$$\delta \overset{\circ}{y_j} = y_j' + \frac{\Delta x^2}{2\cdot 6} y'''(\xi)$$

So δ° is order 2.

3.1

Finite differentiates



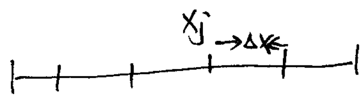
$$\delta \bar{y}_i \equiv \frac{y_i - y_{i-1}}{\Delta x}$$

and $y_j' \equiv y'(x_j) = \delta \bar{y}_j + O(\Delta x)$

$$\delta \overset{+}{y}_i \equiv \frac{y_{j+1} - y_j}{\Delta x}$$

δ^+, δ^- are operators

on a grid



Define the shift operator $S y_j = y_{j+1}$

so $S^2 y_j = S(S y_j) = y_{j+2}$

$S^{-1} y_j = y_{j-1}$

Then $S S^{-1} y_j = y_j$

define $I y_j = y_j$

$\Rightarrow \; S S^{-1} = I$

$$\delta \bar{y}_j = \frac{y_j - y_{j-1}}{\Delta x} = \frac{I y_j - S^{-1} y_j}{\Delta x} = \frac{I - S^{-1}}{\Delta x} = y_j$$

$$\delta^- = \frac{I - S^{-1}}{\Delta x}$$

$$\delta^+ = \frac{S - I}{\Delta x}$$

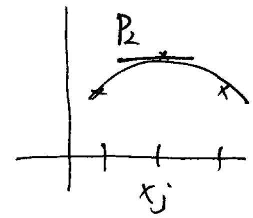
Note: $\dfrac{\delta^+ + \delta^-}{2} = \left(\dfrac{(S-I)}{\Delta x} + \dfrac{(I - S^{-1})}{\Delta x}\right)\dfrac{1}{2} = \dfrac{1}{2} \cdot \dfrac{S - S^{-1}}{\Delta x} \equiv \delta^\circ$

Also, $\dfrac{\delta^\circ y_j - \delta^\circ y_{j+1}}{\Delta x} = \delta^- \delta^\circ y_j$

$\dfrac{d^2}{dx^2} \approx \delta^- \delta^\circ$, or $(\delta^\circ)^2$, or $(\delta^+ \delta^-)$

$$\delta^+\delta^- = \frac{(S-I)}{\Delta x} * \frac{(I-S^-)}{\Delta x} = \frac{S-2I+S^{-1}}{\Delta x^2} = \frac{y_{j+1} - 2y_j + y_{j-1}}{\Delta x^2}$$

$y_{j+1} = y_j + \Delta x\, y_j' + \frac{\Delta x^2}{2} y_j'' + \dots$



call $D = \dfrac{d}{dx}$

$y_{j+1} = I y_j + \Delta x\, D y_j + \Delta x^2 D^2 y_j + \dots$

$$S y_j = \underbrace{(I + \Delta x D + \tfrac{1}{2}\Delta x^2 D^2 + \dots)}_{e^{\Delta x D} = e^{\Delta x \frac{d}{dx}}} y_j$$

$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots$

So $S = e^{\Delta x D}$

$\delta^+ = D + E_{\Delta\,\text{error}}$

$$\frac{S-I}{\Delta x} = \frac{e^{\Delta x D} - I}{\Delta x} = \frac{(I + \Delta x D + \frac{\Delta x^2}{2} D^2 + \dots) - I}{\Delta x}$$

$\delta^+ = D + \left(\frac{\Delta x}{2} D + \dots\right) = D + O(\Delta x)$

1st derivative

$y_j' = \delta^+ y_j + O(\Delta x)$

$y_j' = \delta^- y_j + O(\Delta x)$

$y_j = \delta^\circ y_j + O(\Delta x^2)$ $\qquad \delta^\circ y_j \rightarrow$ centered

$$y_j' = \frac{-y_{j+2} - 4y_{j+1} - 3y_j}{2\Delta x} + O(\Delta x^2)$$

$\uparrow$

$\times\ y_j' = -y_{j+2} + 8y_{j+1} + 8y_j \dots$ one-sided

2^{nd} derivative

$$y_j'' = \delta^+ \delta^- y_j = \frac{y_{j+1} - 2y_j + y_{j-1}}{\Delta x^2}$$

$$y_j'' = \frac{-y_{j+2} + 16 y_{j+1} - 30 y_j + 16 y_{j-1} - y_{j-2}}{12 \Delta x^2} + O(\Delta x^4)$$

3.3.

2.7 Integration of odes

Initial Value Problem IVP

$$\begin{cases} \vec{y}' = \dfrac{d\vec{y}}{dt} = \vec{F}(t, \vec{y}) & t \geq 0 \\ \vec{y}(0) = \vec{y}_0 \end{cases}$$

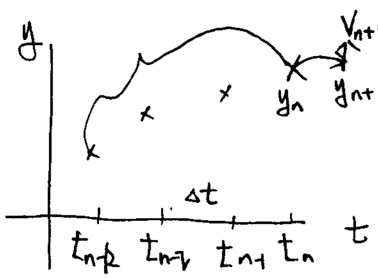
EX: Continuous bond price

$r(t)$ the interest rate

$R(t)$ coupon payment.

$$\frac{dV}{dt} = r(t)\, V - R(t) \equiv F(t, V)$$

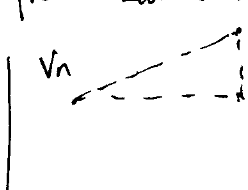
We will look at a class of step-by-step methods $t_n = n \cdot \Delta t$



if $k > 0$, the method is called multistep.

if $k = 0$, the method is called single step.

Example: Euler's method [Forward Euler]



$$\Delta y = \frac{dy}{dt} \cdot \Delta t = F \Delta t$$

Derivative I:

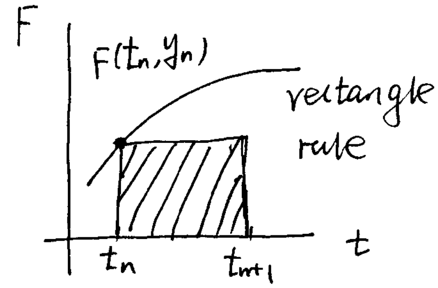
$$y' = F(t, y)$$

$$\int_{t_n}^{t_{n+1}} y'\,dt = \int_{t_n}^{t_{n+1}} F(t,y)\,dt$$

$$y_{n+1} - y_n = \int_{t_n}^{t_{n+1}} F(t,y)\,dt$$

$$y_n \equiv y(t_n)$$

$$y_{n+1} = y_n + \underbrace{\int_{t_n}^{t_{n+1}} F(t,y)\,dt}_{\text{approx. by Quadrature.}}$$



$$F(t, y(t)) = \underbrace{F(t_n, y_n)}_{P_0} + \frac{F'(\xi)}{1!}(t-t_n)$$

$$F(t, y(t)) = \underbrace{F(t_n, y_n)}_{P_0} + \frac{F'(\xi)}{1!}(t-t_n)$$

$$F'(\xi) = y''(\xi)$$

$$y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} F(t_n, y_n)\,dt + y''(\xi)\int_{t_n}^{t_{n+1}} (t-t_n)\,dt$$

$$y_{n+1} = y_n + F(t_n, y_n)\Delta t + y''(\xi)\cdot\frac{\Delta t^2}{2}$$

Drop $o\Delta t^2$ term and let Y_{n+1} satisfy

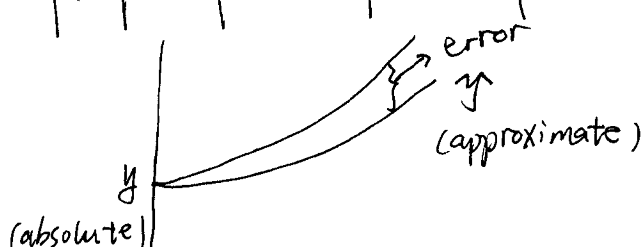
$$Y_{n+1} = Y_n + \Delta t\, F(t_n, Y_n)$$

Example: $\begin{cases} y' = y \\ y(0) = 1 \end{cases}$ $\quad y = e^t$

Euler: $Y_{n+1} = Y_n + \Delta t\, Y_n = (1+\Delta t)\, Y_n$

choose $\Delta t = 0.1$, $Y_{n+1} = 1.1\, Y_n$

n	t_n	Y_n (approx. of y_n)	y_n
0	0	1	1
1	0.1	1.1	1.105
2	0.2	1.210	1.221
3	0.3	1.331	1.350
4	0.4	1.464	1.492



Derivation 2.

$$y_{n+1} = y_n + y'\Delta t + y''(\xi)\cdot\frac{\Delta t^2}{2}$$

$$y' = F(t_n, y_n)$$

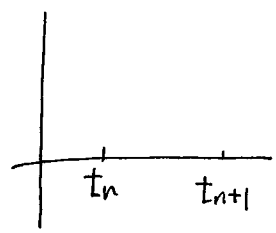
$$y_{n+1} = y_n + F(t_n, y_n)\Delta t + \frac{y''(\xi)}{2}\Delta t^2$$

$$Y_{n+1} = Y_n + F(t_n, Y_n)\Delta t \qquad \text{FE.}$$

Derivation 3.

$$y' = F(t, y)$$

$$\delta_t^+ y_n = F(t_n, y_n) + \text{Error}$$



$$\frac{y_{n+1} - y_n}{\Delta t} = F(t_n, y_n) + E$$

$$y_{n+1} = y_n + \Delta t F(t_n, y_n) + \Delta t E \rightarrow \text{flow away}$$

$$Y_{n+1} = Y_n + \Delta t F(t_n, Y_n)$$

we can find the error over one step a-posteriori

$$Y_{n+1} = Y_n + (\Delta t) F(t_n, Y_n)$$

$$Y_{n+1} = Y_n + \Delta t F(t_n, Y_n)$$

substitute exact solution.

$$y_{n+1} = y_n + \Delta t F(t_n, y_n) + \tau$$

$$y_{n+1} = y_n + y_n' \Delta t + y''(\xi)\cdot\frac{\Delta t^2}{2}$$

$$= y_n + \Delta t F(t_n, y_n) + \tau$$

$$\Rightarrow \tau = y''(\xi)\cdot\frac{\Delta t^2}{2}$$

$\downarrow$

trancation error

Definition: The local truncation error is the error created over one time step.

Definition: A step by step method is of order r if $\tau = O(\Delta t^{n+1})$

Forward Euler is order 1. $\tau \approx O(\Delta t^2)$

Definition: A method is consistent if it is at least order 1.

Remark: Euler's Method

$$y_{n+1} = y_n + \Delta t F(t_n, y_n) + y''(\xi) \cdot \frac{\Delta t^2}{2}$$

$$\frac{y_{n+1} - y_n}{\Delta t} = F(t_n, y_n) + \frac{1}{\Delta t}\tau \qquad \tau = y''(\xi) \cdot \frac{\Delta t^2}{2}$$

and limit, $\lim_{\Delta t \to 0} y' = F + 0$

Want to know.

$$e_n = y_n - Y_n$$

THM: let $\Delta t = \frac{T}{N}$, $t_n = n\Delta t$

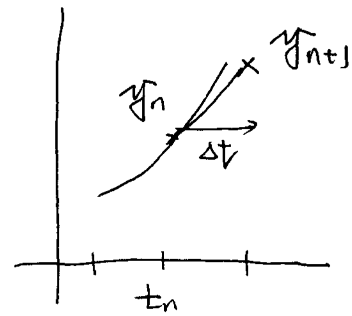
let $Y_{n+1} = Y_n + \Delta t F(t_n, y_n)$

suppose $\left|\frac{\partial F}{\partial y}\right| \le L \left|\frac{\partial F}{\partial t}\right| \le R$, $|F| \le Z$.

then $Y_N \to y(T)$ as $N \to \infty$ ($\Delta t \to 0$)

3.5. $y' = F(t, y)$

Forward E. $\tilde{y}_{n+1} = \tilde{y}_n + \Delta t\, F(t_n, \tilde{y}_n)$



Proof of THM:

$$y_{n+1} = y_n + \Delta t\, F(t_n, y_n) + \tau_n$$
$$-\ \tilde{y}_{n+1} = \tilde{y}_n + \Delta t\, F(t_n, \tilde{y}_n)$$

$\cancel{e_{n+1} = \tilde{y}_{n+1}}$ $\quad e_{n+1} = y_{n+1} - \tilde{y}_{n+1}$

$$= e_n + \Delta t\,[F(t_n, y_n) - F(t_n, \tilde{y}_n)] + \tau_n$$

$$|e_{n+1}| \le |e_n| + \Delta t\, |F(t_n, y_n) - F(t_n, \tilde{y}_n)| + |\tau_n|$$

By MVT. $\quad |F(t_n, y_n) - F(t_n, \tilde{y}_n)| \le \underbrace{\frac{\partial F}{\partial y}}_{\le L} \overbrace{|y_n - \tilde{y}_n|}^{e_n}$

$\tau \sim$ trancation error

$$|e_{n+1}| \le \underbrace{|e_n| + \Delta t L\, |e_n|}_{(1+\Delta t L)|e_n|} + |\tau_n| \qquad \tau = \max_n |\tau_n|$$

$$|e_{n+1}| \le (1+\Delta t L)|e_n| + \tau$$
$$\le (1+\Delta t L)\{(1+\Delta t L)(e_{n-1}) + \tau\} + \tau$$
$$\le (1+\Delta t L)^2 |e_{n-1}| + (1+\Delta t L)\tau + \tau$$
$$\cdots \le \underbrace{(1+\Delta t L)^{n+1} |e_0|}_{0} + \tau \sum_{j=0}^{n-1} (1+\Delta t L)^j$$

$$|e_N| \le \tau \sum_{j=0}^{N-1} (1+\Delta t L)^j$$

Recall $\sum_{j=0}^{N-1} \xi^j = \frac{\xi^N - 1}{\xi - 1}$

So $|e_N| \leq \tau \frac{(1+\Delta t L)^N - 1}{\Delta t L - 1}$

$\cancel{1+\Delta t L} \quad 1+\Delta t L \leq e^{\Delta t L}$

$|e_N| \leq \frac{\tau}{\Delta t}(e^{LT} - 1)/L \Rightarrow |e_N| \leq \frac{\tau}{\Delta t} \frac{e^{LT} - 1}{L}$

But $\tau = \max \frac{1}{2} |y''(\xi)| \Delta t^2$

$|e_N| \leq \frac{1}{2} \max |y''(\xi)| \Delta t \cdot \frac{e^{LT} - 1}{L}$

But $y'' = F' = \frac{dF}{dt} = \frac{\partial F}{\partial t} + \frac{\partial F}{\partial y} \cdot \frac{dy}{dt} \quad (F = \frac{dy}{dt})$

$= \frac{\partial F}{\partial t} + \frac{\partial F}{\partial y} \cdot F$

So, $y'' = \frac{\partial F}{\partial t} + \frac{\partial F}{\partial y} F \leq K + LZ$ just a number

So. $|e_N| \leq \Delta t \left(\frac{1}{2}(K + LZ) \frac{e^{LT} - 1}{L} \right)$

$\lim_{\substack{N \to \infty \\ \Delta t \to \frac{T}{N}}} |e_N| = 0$

Also, $|e| = O(\Delta t)$

$(\tau = O(\Delta t^2))$

RMK: consistency is a necessary condition for convergence.

Except: we assume that

$(1+\Delta t L)^N \cdot |e_0| \equiv 0$

so, really, $|e_N| \leq e^{LT} |e_0| + O(\Delta t)$

if LT is large, this doesn't tell us much.
need condition on growth of errors.

Test problem:

$$y' = \lambda y \qquad \lambda \in \mathbb{C}$$

$$(\vec{y}' = \vec{F}(t, \vec{y}))$$

$\downarrow$ linearize

$$\vec{y} = A(t) * \vec{y}$$

$\downarrow$ matrix

$\downarrow$

freeze t $\quad \vec{y}' = A\vec{y}$

$\downarrow$ diagnolize $\quad \cancel{A = S \Lambda S^{-1}}$ $\downarrow$ ~~matrix~~

$$A = S \Lambda S^{-1}$$

$$\vec{y}' = S \Lambda S^{-1} \vec{y}$$

$$S^{-1} \vec{y}' = \Lambda S^{-1} \vec{y}$$

Define $\vec{w} = S^{-1} \vec{y} \Rightarrow \vec{w} = \Lambda \vec{w}$

$$w_i' = \lambda w_i$$

Now let $\quad u =$ solution with no error initial

$\quad v =$ solution with initial error

$$u' = \lambda u$$
$$v' = \lambda v$$

$$\hat{y}' = (u - v)' = \lambda \hat{y}$$

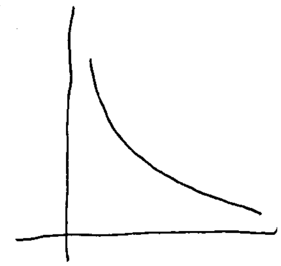
$\uparrow$

$u - v$ error

$$\hat{y}(0) = e_0$$

and scale $y = \dfrac{\hat{y}}{e_0}$

$$\begin{cases} y' = \lambda y \\ y(0) = 1 \end{cases} \rightarrow \text{the problem we have to test}$$



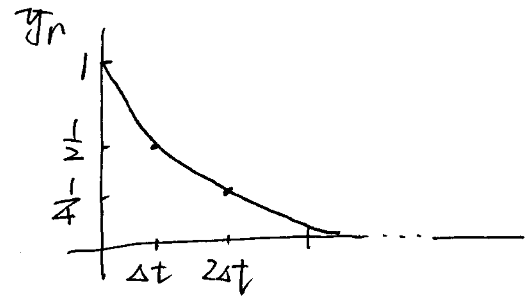
Apply Euler's Method

$$y_{n+1} = y_n + \Delta t(\lambda y_n)$$
$$= (1+\lambda\Delta t)\, y_n$$

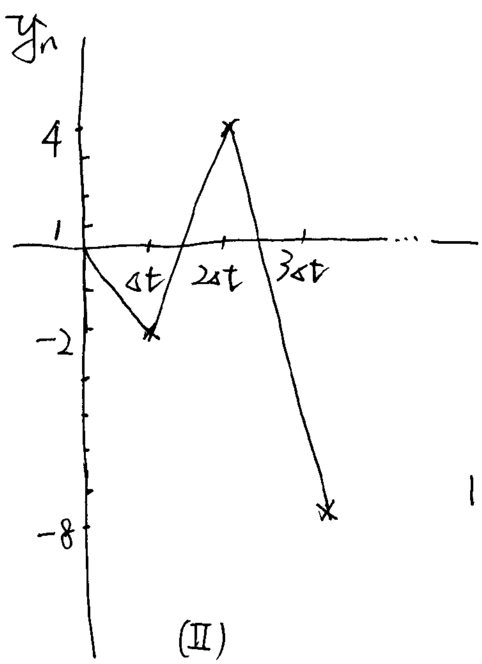
Assume λ is real and negative ($\lambda < 0$)

CASE I: choose $\lambda\Delta t$, such that $1+\lambda\Delta t = \frac{1}{2}$

CASE II: choose Δt, $\Rightarrow$ $1+\lambda\Delta t = -2$



(I)



(II)

$n \to \infty$
$|y_n| \to \infty$ blows up.

To Not blow up,

$$|1+\lambda\Delta t| \le 1$$

Definition: A step-by-step method is absolutely stable if

$$|y_{n+1}| \le |y_n|$$

Forward Euler is absolutely stable if $|1+\lambda\Delta t| \le 1$