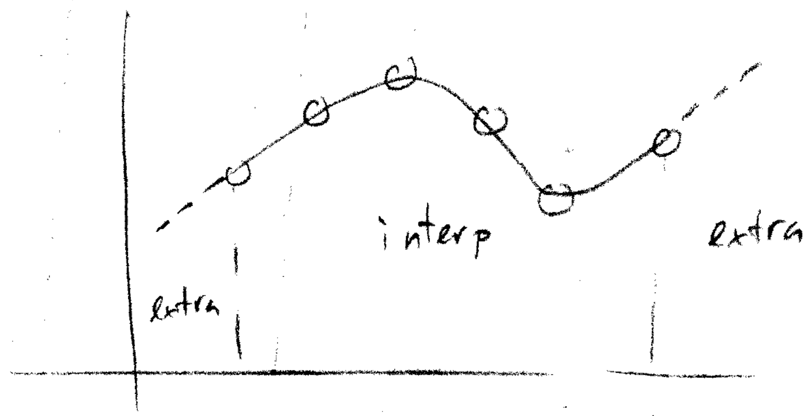
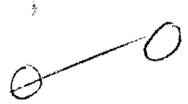
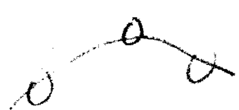
Interpolation & Extrapolation

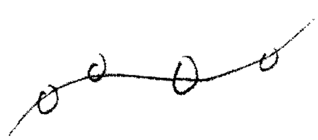
(A) Lagrange Interpolation (polynomials)



$(x_i, f(x_i))$
o – data pts
↑
from experiment
or
expensive
numerical
evaluations

 → unique straight line

 → quadratic

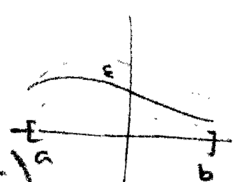
 → cubic

☆ → need in-between values!

→ by constructing $P_n(x)$ thru data pts.

Weierstrass Approximation Theorem

→ $f(x)$ in $[a,b]$ can always be approximated by $P_n(x)$
(continuous)



For a given ε
∃ n such that
$|f(x) - P_n(x)| < \varepsilon$
$\forall x \in [a,b]$.

Note: Interpolation ≠ curve fitting

Inter → good estimate of local $(P(x_i) = f(x_i))$
– $P(x)$ goes thru all pts

curve fit → global behavior
$(P(x_i) \neq f(x_i))$

– $\Sigma\Delta$ is min over ...

Polynomial Construction

Given : N pairs of points $(1, \ldots, N)$: $\{x_i, f(x_i)\}$

Goal : Construct a polynomial $P_{N-1}(x)$ of order $(N-1)$ such that

$$P_{N-1}(x_i) = f(x_i)$$

A simple example : 2 points – $(x_1, f(x_1))$
$(x_2, f(x_2))$

From taylor's series expansion, we have

$$\left.\begin{aligned} f(x_1) &= f(x) + (x_1 - x) f'(x) + O(\Delta^2) \\ f(x_2) &= f(x) + (x_2 - x) f'(x) + O(\Delta^2) \end{aligned}\right\} \text{expanding } f(x_1) \text{ \& } f(x_2) \text{ around } x.$$

★ Now, we want to define a 1st-order polynomial ($P_1(x)$ – linear in x [a straight line]) to approximate $f(x)$ by requiring that

$$f(x_i) = P_1(x_i)$$

Motived by T's expansion, try:

$$f(x_1) = P(x) + (x_1 - x)\, P'(x)$$

P(x) is linear so that $P'(x)$ is only a const.

And $f(x_2) = P(x) + (x_2 - x)\, P'(x)$

★ Note: by construction,

$$f(x_1) = P(x_1) \quad \& \quad f(x_2) = P(x_2)$$

Now, we can solve for $P(x)$ by multipling $(x_2 - x)$ to the top eq. & $(x_1 - x)$ to the bottom eq.

$$\begin{aligned} f(x_1)(x_2 - x) &= (x_2 - x)P(x) + (x_2 - x)(x_1 - x)P'(x) \\ \ominus\; f(x_2)(x_1 - x) &= (x_1 - x)P(x) + (x_1 - x)(x_2 - x)P'(x) \end{aligned}$$

$$f(x_1)(x_2 - x) - f(x_2)(x_1 - x) = P(x)\left[(x_2 - x) - (x_1 - x)\right]$$

$$P(x)(x_2 - x_1)$$

$$\boxed{P(x) = \frac{x - x_2}{x_1 - x_2} f(x_1) + \frac{x - x_1}{x_2 - x_1} f(x_2)}$$

↑

- a first order polynomial such that $P(x_1) = f(x_1)$ & $P(x_2) = f(x_2)$
- a linearly "weighted sum" of $f(x_1)$ & $f(x_2)$

For the General Case with $P_{N-1}(x)$ and N data points:

→ **Lagrange Formula** is a general way to construct $P_{N-1}(x)$ from N datapoints $\{(x_1, f(x_1)), \ldots, (x_N, f(x_N))\}$

$$P_{N-1}(x) = \sum_{K=1}^{N} L_{NK}(x)\, f(x_K)$$

where

$$L_{NK}(x) = \frac{(x-x_1)\cdots(x-x_{K-1})(x-x_K)\cdots(x-x_N)}{(x_K-x_1)\cdots(x_K-x_{K-1})(x_K-x_{K+1})\cdots(x_K-x_N)}$$

$(x-x_K)$ missing!

obviously, no (x_K-x_K) term!

---

example: we have already seen the 2 points example:

Here is one for three points: $\{(x_1, f_1), (x_2, f_2), (x_3, f_3)\}$

$$P_2(x) = \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)} f_1 + \frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x_3)} f_2 + \frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_3-x_2)} f_3$$

☆ Property of $L_{N,K}(x)$:

- $L_{N,K}(x)$ are polynomials of $(N-1)$ degree
- $\begin{cases} L_{N,K}(x_K) = 1 \\ L_{N,K}(x_i) = 0 \text{ if } i \neq K. \end{cases}$ (It acts like a kronecker delta function!)

☆ Property of $P_{N-1}(x)$:

① By construction, we have :

$$P_{N-1}(x_i) = f_i \quad !! \qquad \text{(as required!)}$$

② $L_{N,K}(x)$ are polynomials of N-degree, so that $P_{N-1}(x)$ is a polynomial of degree $(N-1)$!

☆ Main Theorem of Polynomial Interpolation :

let $\{(x_1, f_1), \ldots, (x_N, f_N)\}$ be $N+1$ points in the plane with distinct x_i. Then, there exists <u>one and only one</u> polynomial $P_{N-1}(x)$ of degree $(N-1)$ or less that satisfies $P_{N-1}(x_i) = f_i$ for all $i = 1, \ldots, N$.

Proof (on the uniqueness part):

(existence is given by explicit construction.)

- let $P_{N-1}(x)$ & $Q_{N-1}(x)$ be two different polynomials of degree $(N-1)$ or less that interpolate the given N data points.

- this means that

$$P_{N-1}(x_i) = Q_{N-1}(x_i) = f_i \quad \text{for all } i = 1, \ldots, N.$$

- Now, consider the different function

$$H_{N-1}(x) = P_{N-1}(x) - Q_{N-1}(x)$$

- $H_{N-1}(x)$ is at most of degree $(N-1)$ AND

$$H_{N-1}(x_i) = P_{N-1}(x_i) - Q_{N-1}(x_i) = 0 \quad \text{(at the datapts)!}$$

★ So that, $H_{N-1}(x)$ has N distinct zeros!

- Then, from the Fundamental Theorem of Algebra (a degree p polynomial can have at most p zeros unless it is a identically zero polynomial.)

$\Rightarrow$ $H_{N-1}(x) = 0$! and $P_{N-1}(x)$ is unique!

Neville's algorithm

- ~~recursive way~~ ~~computationally~~ iterative way ~~efficient way~~ to construct one continuous $P_N(x)$ thru $(N+1)$ pts

(4 pts example)

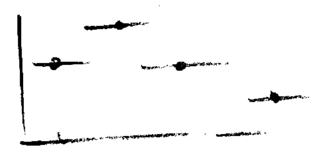
$$\left[\{x_1, x_2, x_3, x_4\} \ \& \ \{f_1, f_2, f_3, f_4\} \right]$$

(Not necessary evenly spaced)

① Define 4 0th order Polynomials (constant fns)

P_1, P_2, P_3, P_4 to appro f near x_1, x_2, x_3, x_4

such that $P_i = f_i$



② For each pairs of pts $[x_i, x_{i+1}]$ (3),

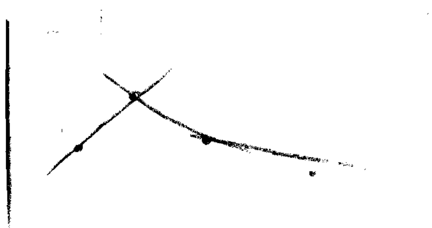
Define 3 1th order polynomials (straight lines)

P_{12}, P_{23}, P_{34} to appro f using P_i :

Note : for each pairs of pts $[x_i, x_{i+1}]$, $\{P_{12}, P_{23}, P_{34}\}$ are unique

↑

constructed from P_i



3 Similarly for high order polynomials with more pts.

— 2 $\begin{cases} x_1, x_2, x_3 \rightarrow P_{123} & \text{quadratic} \\ x_2, x_2, x_3 \rightarrow P_{234} & \text{quadratic} \end{cases}$

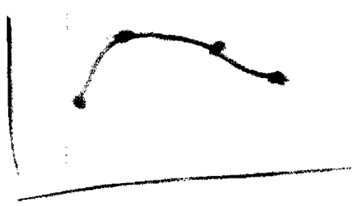
↑ * constructed from P_{ij}

4 Finally, the best we can do with 4 data pts is a cubic!

— 1 - $x_1, x_2, x_3, x_4 \rightarrow P_{1234}$ cubic

↑ * constructed from P_{ijk}

(this is the final goal)



We can put them in a table

0th order poly, 1st order poly, 2nd order poly, 3rd order poly

$$
\begin{array}{lllll}
x_1 : & P_1 & & & \\
 & & P_{12} & & \\
x_2 : & P_2 & & P_{123} & \\
 & & P_{23} & & P_{1234} \\
x_3 : & P_3 & & P_{234} & \\
 & & P_{34} & & \\
x_4 : & P_4 & & &
\end{array}
$$

parents → daughters

— Neville's algorithm is an efficient method to fill them in from left to right!

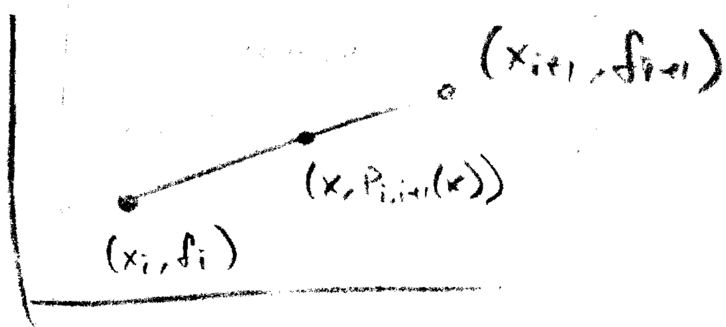
— P_i : Construction of 0th order is simple:

$$P_i = f_i \qquad \{P_1, P_2, P_3, P_4\}$$

— $P_{i,i+1}$: linear interpolation between P_i, P_{i+1}

$$P_{1,i+1}(x) = \frac{(x - x_{i+1}) \overset{(f_i)}{P_i} - (x - x_i) \overset{(f_{i+1})}{P_{i+1}}}{(x_i - x_{i+1})}$$

$\{P_{12}, P_{23}, P_{34}\}$



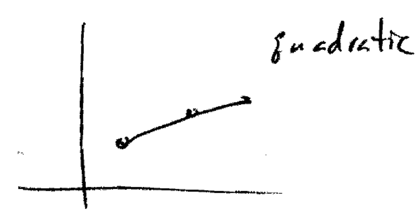
☆ Lagrange for two pts

— $P_{i,i+1,i+2}$: "again linear interpolation" between $P_{i,i+1}, P_{i+1,i+2}$

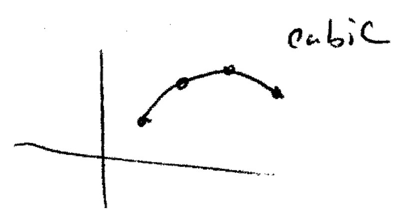
$\{P_{123}, P_{234}\}$

see (8')

$$P_{123}(x) = \frac{(x - x_3) P_{12} - (x - x_1) P_{23}}{(x_1 - x_3)}$$



— $$P_{1234}(x) = \frac{(x - x_4) P_{123} - (x - x_4) P_{234}}{(x_1 - x_4)}$$



To show $\swarrow$ "linear interpolation formula" (8')

$$P_{123}(x) = \frac{(x-x_3)P_{12} - (x-x_1)P_{23}}{(x_1-x_3)}$$

does give $P_{123}(x) \rightarrow$ Quadratic three pts Lagrange formula

---

$$\frac{(x-x_3)}{(x_1-x_3)}\left[\frac{(x-x_2)f_1}{(x_1-x_2)} - \frac{(x-x_1)f_2}{(x_1-x_2)}\right]$$

$$- \frac{(x-x_1)}{(x_1-x_3)}\left[\frac{(x-x_3)f_2}{(x_2-x_3)} - \frac{(x-x_2)f_3}{(x_2-x_3)}\right]$$

$$= \frac{(x-x_3)(x-x_2)}{(x_1-x_3)(x_1-x_2)}f_1$$

$$\left[-\frac{(x-x_3)(x-x_1)(x_2-x_3)}{(x_1-x_3)(x_1-x_2)(x_2-x_3)}f_2 - \frac{(x-x_1)(x-x_3)(x_1-x_2)}{(x_1-x_3)(x_2-x_3)(x_1-x_2)}f_2\right.$$

$$+ \frac{(x-x_1)(x-x_2)}{(x_1-x_3)(x_2-x_3)}f_3$$

$$- \left(\frac{(x-x_3)(x-x_1)f_2}{\cancel{(x_1-x_3)}(x_1-x_2)(x_2-x_3)}\left[\cancel{x_2} - \cancel{x_3} + \cancel{x_1} - \cancel{x_2}\right]\right)$$

$$= \frac{(x-x_3)(x-x_1)}{(x_2-x_1)(x_2-x_3)}f_2 \quad \checkmark$$

Advantage:

This can be put into an iterative form:

e.g.

two parents: $P_{\underbrace{i\,(i+1)\ldots(i+m-1)}_{m}}$ (P_{123})

$P_{\underbrace{(i+1)(i+2)\ldots(i+m)}_{m}}$ (P_{234})

one daughter: $P_{\underbrace{i\,(i+1)\ldots(i+m)}_{(m+1)}}$ (P_{1234})

(*)

$$\underbrace{P_{\overset{1st}{i}\,(i+1)\ldots\overset{last}{(i+m)}}}_{m+1} = \frac{(X-\overset{last}{X_{i+m}})\,\overset{P_{1st-2nd\,to\,last}}{P_{\underbrace{i\,(i+1)\ldots(i+m-1)}_{m}}} - (X-\overset{1st}{X_i})\,\overset{P_{2nd-last}}{P_{\underbrace{(i+1)(i+2)\ldots(i+m)}_{m}}}}{(\underset{1st}{X_i} - \underset{last}{X_{i+m}})}$$

Usage of this algorithm

★ input : x output : $P_{1234}(x)$

$$
\begin{tikzcd}[column sep=small, row sep=tiny]
 & X_1 : & Y_1 = P_1 \arrow[dr, dash] & & & \\
 & & & P_{12} \arrow[dr, dash] & & \\
x \arrow[r] & X_2 : & Y_2 = P_2 \arrow[ur, dash] \arrow[dr, dash] & & P_{123} \arrow[dr, dash] & \\
 & & & P_{23} \arrow[ur, dash] \arrow[dr, dash] & & P_{1234} \\
 & X_3 : & Y_3 = P_3 \arrow[ur, dash] \arrow[dr, dash] & & P_{234} \arrow[ur, dash] & \\
 & & & P_{34} \arrow[ur, dash] & & \\
 & X_4 : & Y_4 = P_4 \arrow[ur, dash] & & &
\end{tikzcd}
$$

best est at n stage ↓ ⟶ higher order

		fill in tree
0th order	$Y \simeq P_2 = Y_2$	P_1, P_3, P_4
1th order	$Y \simeq P_{23} = \dfrac{(x - x_3) P_2 - (x - x_2) P_3}{(x_2 - x_3)}$	P_{12}, P_{34}
2nd order	$Y \simeq P_{123} = \dfrac{(x - x_3) P_{12} - (x - x_1) P_{23}}{(x_1 - x_3)}$	P_{234}
3rd order	$Y \simeq P_{1234} = \dfrac{(x - x_4) P_{123} - (x - x_1) P_{234}}{(x_1 - x_4)}$	—

★ it is a way to calculate (estimate) the value $f(x) \simeq P_{1234}(x)$ by.

~~it is not~~

→ it does not give the coef. of the estimating function!

⇒ need to go thru the tree procedure for each x.