Interpolation Overview: https://www.youtube.com/playlist?list=PL-xEHa0Tmq2w0vLOjmjFkbTUArmHFzbQV

- You have a set of data points;
  - ex. Have a bunch x and y values and plot many points
  - **Interpolation**: find a curve that passes through all the points, then being able to predict other values within a **range**
    - Interpolating the output values within a **range of values** [a, b], to predict the x & y values

POLYNOMIAL INTERPOLATION ⇒ they are INF derivatable and can be of any shape

(1) Vandermonde Matrix

Ex. $\{x_i, y_i\},\ i = 0, 1, 2, ..., n \Rightarrow P_n(x)\ s.t.\ P_n(x_i) = y_i$

- Use polynomial approximations to the **data set** $\{x_i, y_i\}$
- We have n+1 data points: i = 0, 1, 2, … , n
- This data set should help us produce a polynomial of nth degree, and a polynomial such that $P_n(x_i) = y_i$

General Steps:

1. *Given our current polynomial:* $P_n(n) = a_0 + a_1x + a_2x^2 + ... + a_nx^n$
2. *Then this should happen:* $P_n(x_0) = y_0 = a_0 + a_1 x_0 + a_2x_0^2 + ... + a_nx_0^n$
3. *Substitute other x values into polynomial like in step 2: ex.* $P_n(x_1) = y_1 ....$
   a. *At the nth equation:* $P_n(x_n) = y_n : a_0 + a_1 x_n + a_2x_n^2 + ... + a_nx_n^n$
      i. *What values are known? i : the data points ==> so we have* $x_0,\ x_1,\ ...,\ x_n$
      ii. *We dont have any of the a value*
4. *Write the polynomials in matrix form: 'c' same thing as 'a'*

$$\Leftarrow \begin{bmatrix} 1 & x_0 & x_0^2 & \cdot & x_0^n \\ 1 & x_1 & x_1^2 & \cdot & x_1^n \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & x_n & x_n^2 & \cdot & x_n^n \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \cdot \\ c_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \cdot \\ y_n \end{bmatrix}$$ **VANDERMONDE MATRIX**

x_i	0	1	1/3
y_i	1	0	1/2

$n = 2$

$P_2(x) = a_0 + a_1 x + a_2 x^2$

$P_2(0) = 1 : \quad a_0 = 1$

$P_2(1) = 0 : \quad a_0 + a_1 + a_2 = 0$

$P_2(\frac{1}{3}) = \frac{1}{2} : \quad a_0 + \frac{1}{3}a_1 + \frac{1}{9}a_2 = \frac{1}{2}$

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & \frac{1}{3} & \frac{1}{9} & \frac{1}{2} \end{array}\right]$$

$a_0 = 1,\ a_1 = -\frac{1}{4},\ a_2 = -\frac{3}{4}$

$P_2(x) = -\frac{3}{4}x^2 - \frac{1}{4}x + 1$ Test $P_2(0) = 1$?

Example:

So the resulting equation represents the data points (i.e. x & y values)

We can find the range of data point values: [0, 1]

Problems:

- It gets more expensive for higher power polynomial
- Any tiny errors can diverge
- Large 'condition numbers'
- Approximation is just not accurate

(2) Lagrange Interpolation

Same situation: $\{x_i, y_i\},\ i = 0, 1, 2, ..., n :\ n+1$ *data points*

General Steps:

1. *Define cardinal functions:* $L_0, L_1, ..., L_n \in P_n$ *satisfying:* $L_i(x_j) = \delta_{ij} = \{1 \text{ where } i = j;\ 0 \text{ where } i \neq j\}$
   a. *These cardinal functions* (L_n) *belongs to* P_n*, so L can have power up to n.*
   b. δ_{ij} *= Kronecker Delta: so when* $i = j,\ \delta_{ij} = 1$*; when* $i \neq j,\ \delta_{ij} = 0$
      i. *So CARDINAL FUNCTIONS can only be either a 0 or 1!!!*
2. $L_i(n) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \Rightarrow \prod$ *is like* $\sum$ *but for multiplication*
3. $$l_i(x) = \frac{(x - x_0)\cdots(x - x_{i-1})(x - x_{i+1})\cdots(x - x_n)}{(x_i - x_0)\cdots(x_i - x_{i-1})(x_i - x_{i+1})\cdots(x_i - x_n)}, \qquad l_i(x_j) = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases}$$

- *we skipped over the ith multiplication since $j \neq i$, because then denominator $= 0$*

4. *Check if these cardinal functions follow property : i.e. have value of either 0 or 1*
   a. *Ex.* $L_i(x_i) = 1$

Lagrange form of the interpolating polynomial for $\{x_i, y_i\}$, $i = 0, 1, \ldots, n$ ⇒ $$p(x) = \sum_{i=0}^{n} l_i(x) y_i,$$

x	0	1	$3/2$
y	1	0	$1/2$

$n=2$ $\quad P_2(x) = \sum_{i=0}^{2} L_i(x) y_i = L_0(x)y_0 + L_1(x)y_1 + L_2(x)y_2$

$P_2(x) = L_0(x) + \frac{1}{2} L_2(x)$

$$L_0(x) = \prod_{j=1}^{2} \frac{x - x_j}{x_0 - x_j} = \left(\frac{x - x_1}{x_0 - x_1}\right)\left(\frac{x - x_2}{x_0 - x_2}\right) = \left(\frac{x-1}{-1}\right)\left(\frac{x - \frac{3}{2}}{-\frac{3}{2}}\right) = \frac{1}{2}(x-1)(3x-3)$$

$$L_2(x) = \prod_{j=0}^{1} \frac{x - x_j}{x_2 - x_j} = \left(\frac{x - x_0}{x_2 - x_0}\right)\left(\frac{x - x_1}{x_2 - x_1}\right) = \left(\frac{x}{3/2}\right)\left(\frac{x-1}{-3}\right) = -\frac{9}{2}x(x-1)$$

$$P_2(x) = \frac{1}{2}(3x^2 - 5x + 2) + \frac{1}{2}\left(-\frac{9}{2}\right)(x^2 - x)$$

$$= \frac{3}{2}x^2 - \frac{5}{2}x + 1 - \frac{9}{4}x^2 + \frac{9}{4}x$$

$$= -\frac{3}{4}x^2 - \frac{1}{4}x + 1$$

(2.b) Approximating Functions Using Lagrange Interpolation: https://youtu.be/c_5nwnvV2WU

ex. $f(x) = \frac{1}{x}$ want to appox over 3 points ⇒ $[2, 4]$

$x_0 = 2, x_1 = 2.75, x_2 = 4$

x	2	2.75	4
$f(x) = y$	$1/2$	$4/11$	$1/4$

* prev method, we have data pts (both x & y values)
↳ using polynomial to approx data.

* NOW ⇒ now we approx function by using data generated by f(x)
↳ ex. we want to approximate $f(3) = 1/3$

Solution: $\quad P(x) = \sum_{i=0}^{2} f(x_i) L_i(x)$ ⇐ lagrange.

$$= f(x_0)L_0(x) + f(x_1)L_1(x) + f(x_2)L_2(x)$$

$$L_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} = \frac{(x - 2.75)(x - 4)}{(2 - 2.75)(2 - 4)} = \frac{2}{3}(x - 2.75)(x - 4)$$

$$L_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} = \frac{(x-2)(x-4)}{(2.75 - 2)(2.75 - 4)} = -\frac{16}{15}(x-2)(x-4)$$

$$L_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} = \frac{(x-2)(x-2.75)}{(4-2)(4-2.75)} = \frac{2}{5}(x-2)(x-2.75)$$

$$\therefore P(x) = \frac{1}{3}(x - 2.75)(x-4) - \frac{64}{165}(x-2)(x-4) + \frac{1}{10}(x-2)(x-2.75)$$

$$= \frac{1}{22}x^2 - \frac{35}{88}x + \frac{49}{44}$$

⇒ $P(3) \approx 0.32955 \quad 1/3 \approx 0.333\ldots$

error is ok... ≈ 0.00378.

(2.c) Lagrange Polynomial Error (Error Bound for Function Approx): https://youtu.be/RG5mTXLB9iA

Lagrange Cons: if we have another set of points that we want to use to look at a higher order of interpolation, we have to start over

- SOLUTION: Newton's Divided Differences

NEWTON'S DIVIDED DIFFERENCES

Current data: $\{x_0, x_1 .... x_n\}$ & $\{y_0, y_1 .... Y_n\}$ Current polynomial: $P_n(x)$

New data: x_{n+1} & y_{n+1} (now there are n+1 data points) New polynomial: $P_{n+1}(x)$

- (through interpolation of the current polynomial)

Objective:

- We want a polynomial to interpolate the current data points to create 2 new data points: n+1st and n+2nd data point
  - i.e. x_{n+1} and y_{n+1}
- After we get this new data set, we want a method where the previous poly { $P_n(x)$ } can use the new points to create $P_{n+1}(x)$ without losing the work from computing the previous poly (as would happen if we use Lagrange method)
- This additional data point actually strengthens our polynomial: better approximation

How it works:

Newton's Divided Differences

Data: $\left\{\begin{array}{l|l} x_0, x_1, \dots x_n & x_{n+1} \\ y_0, y_1, \dots, y_n & y_{n+1} \end{array}\right.$ $\quad P_n(x) \Rightarrow P_{n+1}(x)$

Start with:

$$P_0(x) = y_0 \Rightarrow P_0(x_0) = y_0 \quad .. \quad \underline{P_0(\text{anything}) = y_0}$$

$$P_1(x) = P_0(x) + a_1(x - x_0)$$

$$\Rightarrow P_1(x_0) = P_0(x_0) + a_1(x_0 - x_0) = P_0(x_0) = y_0$$

$$\Rightarrow P_1(x_1) = P_0(x_1) + a_1(x_1 - x_0) = y_1$$

$$y_1 = y_0 + a_1(x_1 - x_0)$$

$$a_1 = \left(\frac{y_1 - y_0}{x_1 - x_0}\right) \approx f'(x)$$

$$P_2(x) = P_1(x) + a_2(x - x_0)(x - x_1)$$

$$\Rightarrow P_2(x_0) = P_1(x_0) + a_2(x_0 - x_0)(x_0 - x_1) = y_0$$

$$\Rightarrow P_2(x_1) = P_1(x_1) + a_2(x_1 - x_0)(x_1 - x_1) = P_1(x_1) = y_1$$

$$\Rightarrow P_2(x_2) = \underbrace{P_1(x_2)}_{\text{don't know this yet}} + a_2(x_2 - x_0)(x_2 - x_1) = y_2$$

$$a_2 = \frac{y_2 - P_1(x_2)}{(x_2 - x_0)(x_2 - x_1)}$$

$$P_1(x_2) = P_0(x_2) + a_1(x_2 - x_0)$$
$$= y_0 + a_1(x_2 - x_0)$$
$$= y_0 + \left(\frac{y_1 - y_0}{x_1 - x_0}\right)(x_2 - x_0)$$
$$= y_0 + \left(\frac{y_1 - y_0}{x_1 - x_0}\right)(x_2 - x_1 + x_1 - x_0)$$
$$= \cancel{y_0} + \left(\frac{y_1 - y_0}{x_1 - x_0}\right)(x_2 - x_1) + (y_1 - \cancel{y_0})$$
$$= \left(\frac{y_1 - y_0}{x_1 - x_0}\right)(x_2 - x_1) + y_1$$

★ sub back to get a_2.

So
$$a_2 = \frac{y_2 - y_1 - (y_1 - y_0)\left(\frac{x_2 - x_1}{x_1 - x_0}\right)}{(x_1 - x_0)(x_2 - x_1)}$$
$$= \frac{y_2 - y_1}{(x_2 - x_0)(x_2 - x_1)} - \frac{y_1 - y_0}{(x_1 - x_0)(x_2 - x_0)}$$

★ $(x_2 - x_0)$ is common in both.

$$= \left(\frac{\frac{y_2 - y_1}{x_2 - x_1} - \frac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_0}\right) \approx f''(x)$$

RECAP

Assume P_{K-1} (Polynomial) interpolates $\{x_i, y_i\}$ for $i = 0, 1, \dots, k-1$

$\Rightarrow$ w/ Divided differences, we want to find P_K that interpolates extra point (x_K, y_K)

$$★ \; P_K(x) = P_{K-1}(x) + a_K(x - x_0)(x - x_1) \dots (x - x_{K-1})$$

use $P_K(x_K) = y_K$ to find a_K

$$a_K = \frac{y_K - P_{K-1}(x_K)}{(x_K - x_0)(x_K - x_1) \dots (x_K - x_{K-1})}$$

$$★ \; P_n(n) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \dots + a_n(x - x_0)(x - x_1) \dots (x - x_{n-1})$$

Divided Differences

NOTATION:

Given $\{x_i, y_i\}$ $i = 0, 1, \ldots, n$.

① $f[x_i] = y_i$ $i = 0, 1, 2, \ldots, n$

★ ex. $f[x_0, x_1] = \frac{f[x_1] - f[x_0]}{x_1 - x_0}$ ("y_1" "y_0") ⇒ for every 2 adjacent pts.

$\boxed{\text{first divided diff}}$ ⇒ w/ only 2 vars

using ① $= \frac{y_1 - y_0}{x_1 - x_0}$

★ ex. 2nd Divided Diff:

$$f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}$$

⋮

★ $$f[x_0, x_1, \ldots, x_n] = \frac{f[x_1, x_2, \ldots, x_n] - f[x_0, x_1, \ldots x_{n-1}]}{x_n - x_0}$$

Previously: $P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \ldots$
$+ a_n(x - x_0)(x - x_1)\ldots(x - x_{n-1})$

where $a_0 = f[x_0]$ ... $a_n = f[x_0, \ldots, x_n]$

Divided Difference Practice Example:

Given:

x_i	0	2/3	1
y_i	1	1/2	0

⇒ $n = 2$.

x_i	$f[x_i]$	$f[x_i, x_{n+1}]$	
0	1		
		$\frac{1/2 - 1}{2/3 - 0} = -3/4$	
2/3	1/2		$\frac{-3/2 - (-3/4)}{1 - 0} = -\frac{3}{4}$
		$\frac{0 - 1/2}{1 - 2/3} = -3/2$	
1	0		

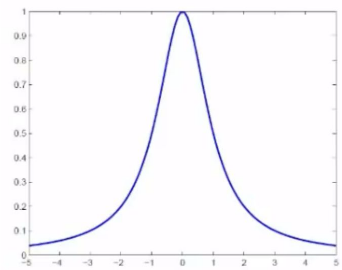
$P_2(x) = 1 + (-\frac{3}{4})(x - 0) + (-\frac{3}{4})(x - 0)(x - \frac{2}{3})$
$= 1 - \frac{3}{4}x - \frac{3}{4}x^2 + \frac{1}{2}x$
$= 1 - \frac{1}{4}x - \frac{3}{4}x^2$

INTERPOLATION - CUBIC SPLINES

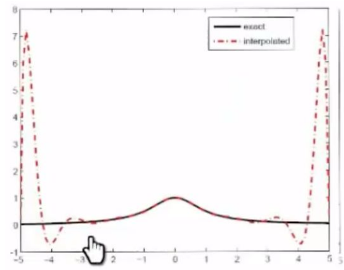
SPLINES OVERVIEW:

Why do we need another form of interpolation? From using divided difference and lagrange, it seems that for higher order polynomials, there are more oscillations (i.e. it has multiple turning points).

Ex. $f(x) = \frac{1}{1+x^2}$ and let's try to fit a 15 or 16th order polynomial into it...what happens



⇒



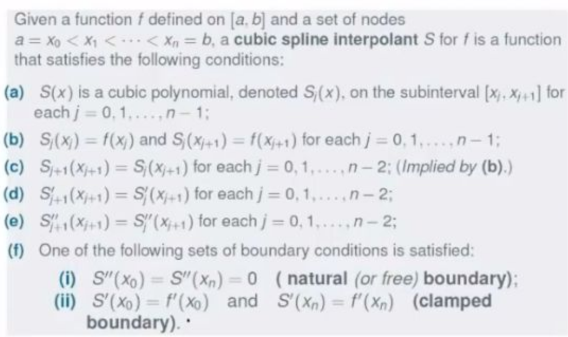
oscillations are in red,

Median approximations are okay, but the edges are way off

Splines Method Concept: connect the dots (i.e. using linear interpolants)

Take any data points that you have, rather than trying to fit ONE POLY in all given data points… you try to find an interpolant for two points at a time. ⇒ easiest is to connect them with **linear interpolants**

- CONS: there are many abrupt turns and jaggedness ⇒ then you can also use quadratic interpolants etc.
- Each of the interpolants that you use to connect between two points is called a **SPLINE**
  - Main thing is to use not too high order for splines ⇒ usually use within **cubic splines**



Given a function f defined on $[a, b]$ and a set of nodes $a = x_0 < x_1 < \cdots < x_n = b$, a **cubic spline interpolant** S for f is a function that satisfies the following conditions:

(a) $S(x)$ is a cubic polynomial, denoted $S_j(x)$, on the subinterval $[x_j, x_{j+1}]$ for each $j = 0, 1, \ldots, n-1$;

(b) $S_j(x_j) = f(x_j)$ and $S_j(x_{j+1}) = f(x_{j+1})$ for each $j = 0, 1, \ldots, n-1$;

(c) $S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ for each $j = 0, 1, \ldots, n-2$; *(Implied by* **(b)**.*)*

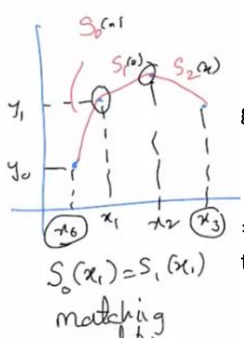
(d) $S'_{j+1}(x_{j+1}) = S'_j(x_{j+1})$ for each $j = 0, 1, \ldots, n-2$;

(e) $S''_{j+1}(x_{j+1}) = S''_j(x_{j+1})$ for each $j = 0, 1, \ldots, n-2$;

(f) One of the following sets of boundary conditions is satisfied:

(i) $S''(x_0) = S''(x_n) = 0$ (**natural** *(or free)* **boundary**);

(ii) $S'(x_0) = f'(x_0)$ and $S'(x_n) = f'(x_n)$ (**clamped boundary**).



- S0, S1, S2 are polynomials/splines that would fit out given data points...but what happens at the junctions?
- At junction x1 ⇒ we expect a <u>**matching condition**</u>:
  - $S_0(x_1) = S_1(x_1)$

⇒ to make it more smooth at the connecting points, we need to set more requirements:

- $S_0'(x_1) = S_1'(x_1)$ && $S_0''(x_1) = S_1''(x_1)$
- $S_0(x_0) = y_0$

⇒ All data points are called <u>**NODES**</u>

⇒ **Boundary conditions**: depends on how many points/nodes you have:

- <u>**Natural splines**</u>: $S''(x_0) = S''(x_n) = 0$ ⇐ use this condition instead
- Example: given 4 data points/nodes ⇒ then we will have 3 splines
  - $S_0''(x_0) = S_2''(x_3) = 0$
- <u>**Clamp splines**</u>: some issues about needing more info like derivatives on top of data pts idk

⇒ **Structures of Splines**

$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$ ⇒ a, b, c, d are unknown coefficients...

⇒ So as in our 4 nodes example, we have 3 splines of equations

- $S_0(x) = a_0 + b_0(x - x_0) + c_0(x - x_0)^2 + d_0(x - x_0)^3$… same thing for S_1 & S_2etc.
- So we have **12 unknown coefficients & 6 equations**
- Make sure to look at any overlaps for matching conditions

Natural Cubic Splines Example

Cubic Splines Example

★ Construct a natural cubic spline that passes through the pts. (1,2), (2,3) & (3,5)

$[1,2]$: $S_0(x) = a_0 + b_0(x-1) + c_0(x-1)^2 + d_0(x-1)^3$
$[2,3]$: $S_1(x) = a_1 + b_1(x-2) + c_1(x-2)^2 + d_1(x-2)^3$

} 8 unknowns, 2 equations

Find 8 unknowns:

$(1,2) \Rightarrow S_0(1) = 2$ sub 1 into S_0

$\Rightarrow a_0 = 2$ 7 more to go.

$(2,3) \Rightarrow S_0(2) = 3$

$\Rightarrow a_0 + b_0 + c_0 + d_0 = 3$

$b_0 + c_0 + d_0 = 1$ — (1)

$(2,3) \Rightarrow S_1(2) = 3$ $a_1 = 3$.

$(3,5) \Rightarrow S_1(3) = 5$

$\Rightarrow 3 + b_1 + c_1 + d_1 = 5$

$b_1 + c_1 + d_1 = 2$ — (2)

$S_0'(x) = b_0 + 2c_0(x-1) + 3d_0(x-1)^2$.
$S_0''(x) = 2c_0 + 6d_0(x-1)$
$S_1'(x) = b_1 + 2c_1(x-2) + 3d_1(x-2)^2$.
$S_1''(x) = 2c_1 + 6d_1(x-2)$

★ $S_0'(2) = S_1'(2)$ ← need to satisfy this condition

$\Rightarrow b_0 + 2c_0 + 3d_0 = b_1$ — (3)

★ $S_0''(2) = S_1''(2)$

$\Rightarrow 2c_0 + 6d_0 = 2c_1$ — (4)

$$S(x) = \begin{cases} 2 + \frac{3}{4}(x-1) + \frac{1}{4}(x-1)^3 & \text{for } x \in [1,2] \\ 3 + \frac{3}{2}(x-2) + \frac{3}{4}(x-2)^2 - \frac{1}{4}(x-2)^3 & \text{for } x \in [2,3] \end{cases}$$

Solution

B.C : $S_0''(1) = 0$; $S_1''(3) = 0$

$\Rightarrow c_0 = 0$ $\Rightarrow 2c_1 + 6d_1 = 0$ — (5)

Using the equations (5)

$$\Rightarrow \left[\begin{array}{ccccc|c} 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 2 \\ 1 & 3 & -1 & 0 & 0 & 0 \\ 0 & 3 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 6 & 0 \end{array}\right] \Rightarrow$$

$b_0 = 3/4$, $d_0 = 1/4$, $b_1 = 3/2$, $c_1 = 3/4$, $d_1 = -1/4$

$a_0 = 2$, $a_1 = 3$, $c_0 = 0$