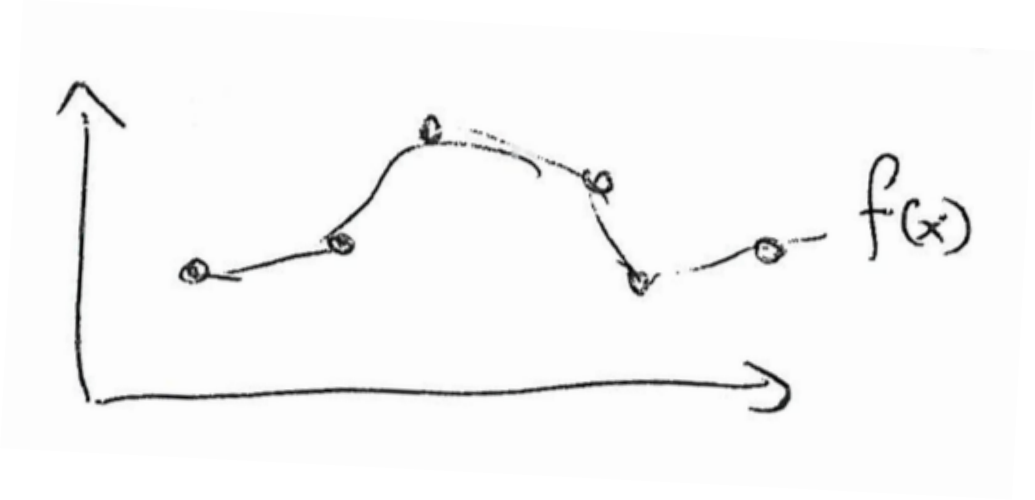
Numerical Methods II: interpolation

The basic problem is well known: given the values ($f_1,f_2,\ldots,f_N$) of a function $f=f(x)$ at the points ($x_1,x_2,\ldots,x_N$), where $f_i=f(x_i)$, find:

1) $f(\bar{a})$, where $\bar{a}$ inside $[x_1,x_N]$: **interpolation**
2) $f(\bar{a})$, where $\bar{a}$ outside $[x_1,x_N]$: **extrapolation**

Both interpolation and extrapolation must model a function among or beyond the assigned set of points. For this we need model functions that are sufficiently general to accommodate (e.g., to approximate) a large class of functions.



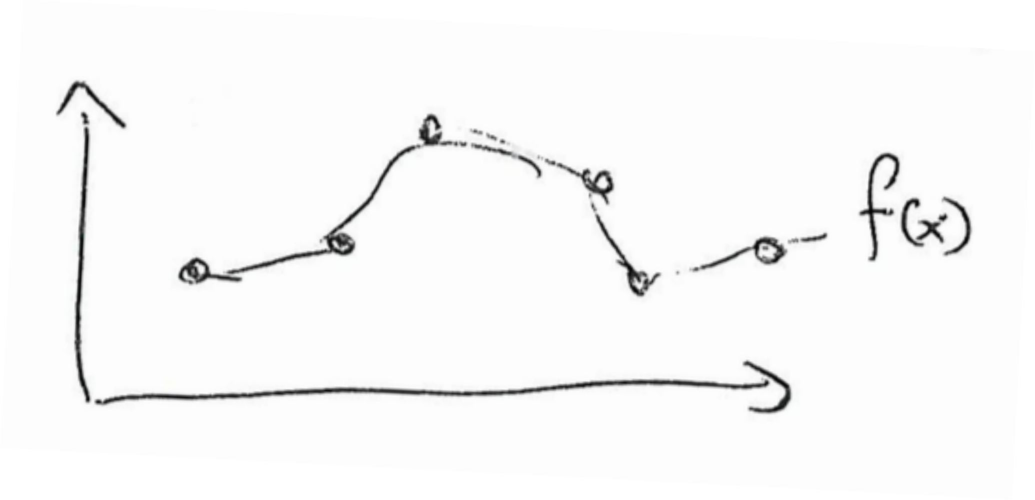
Exemples:

- polynomials
- rational functions
- trigonometric functions

The basic problem is well known: given the values $(f_1,f_2,\ldots,f_N)$ of a function $f=f(x)$ at the points $(x_1,x_2,\ldots,x_N)$, where $f_i=f(x_i)$, find:

1) $f(\bar{a})$, where $\bar{a}$ inside $[x_1,x_N]$: **interpolation**
2) $f(\bar{a})$, where $\bar{a}$ outside $[x_1,x_N]$: **extrapolation**

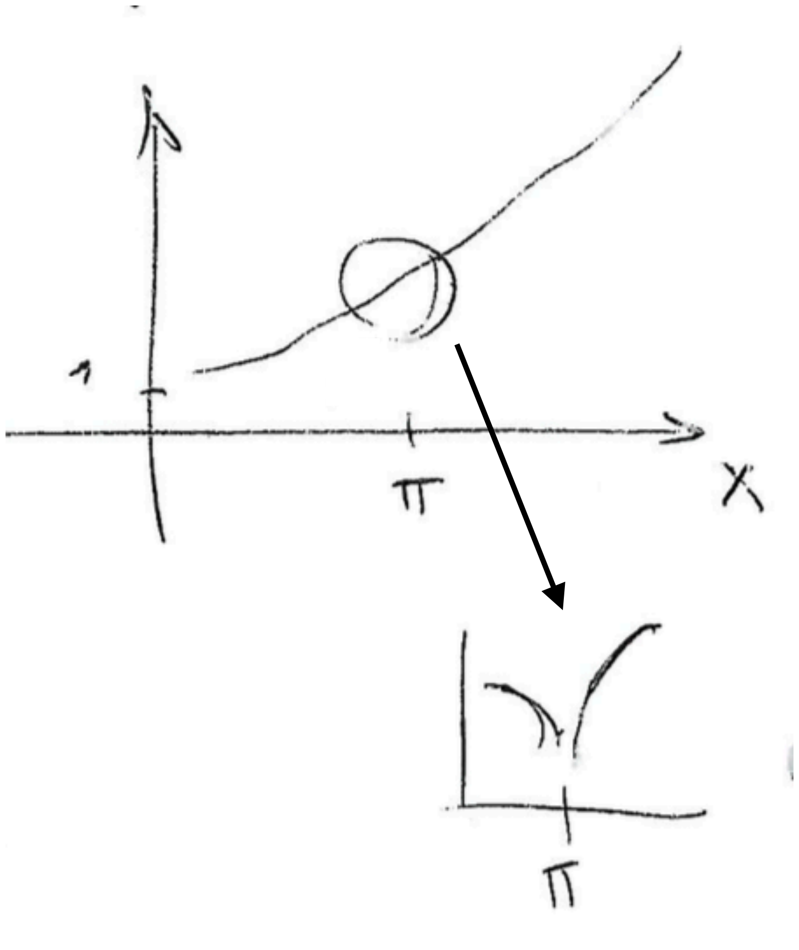
Both interpolation and extrapolation must model a function among or beyond the assigned set of points. For this we need model functions that are sufficiently general to accommodate (e.g., to approximate) a large class of functions.



Exemples:
- polynomials
- rational functions
- trigonometric functions

Good model functions cannot solve pathological problems, e.g.,

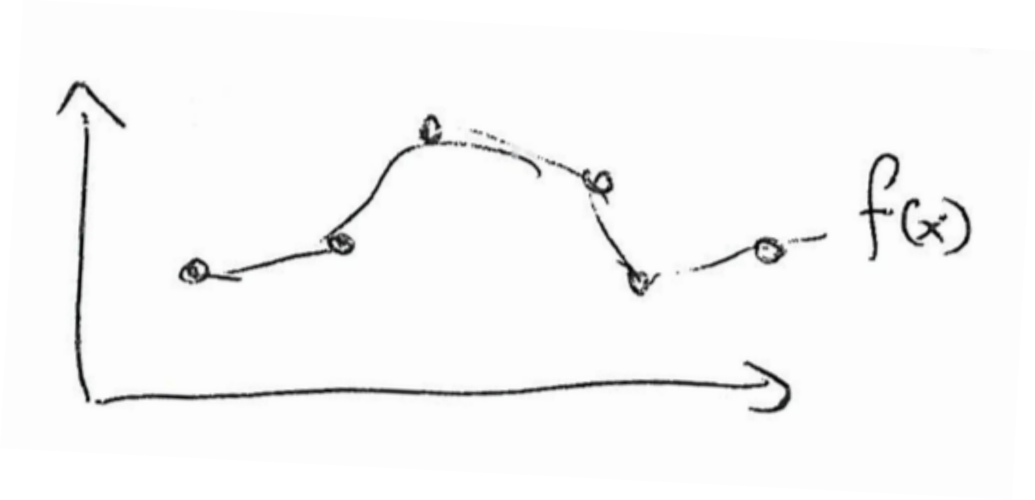
$$f(x) = 3x^2 + \frac{1}{\pi^4}\ln\left[(\pi - x)^2\right] + 1$$



The basic problem is well known: given the values $(f_1,f_2,\ldots,f_N)$ of a function $f=f(x)$ at the points $(x_1,x_2,\ldots,x_N)$, where $f_i=f(x_i)$, find:

1) $f(\bar{a})$, where $\bar{a}$ inside $[x_1,x_N]$: **interpolation**
2) $f(\bar{a})$, where $\bar{a}$ outside $[x_1,x_N]$: **extrapolation**

Both interpolation and extrapolation must model a function among or beyond the assigned set of points. For this we need model functions that are sufficiently general to accommodate (e.g., to approximate) a large class of functions.

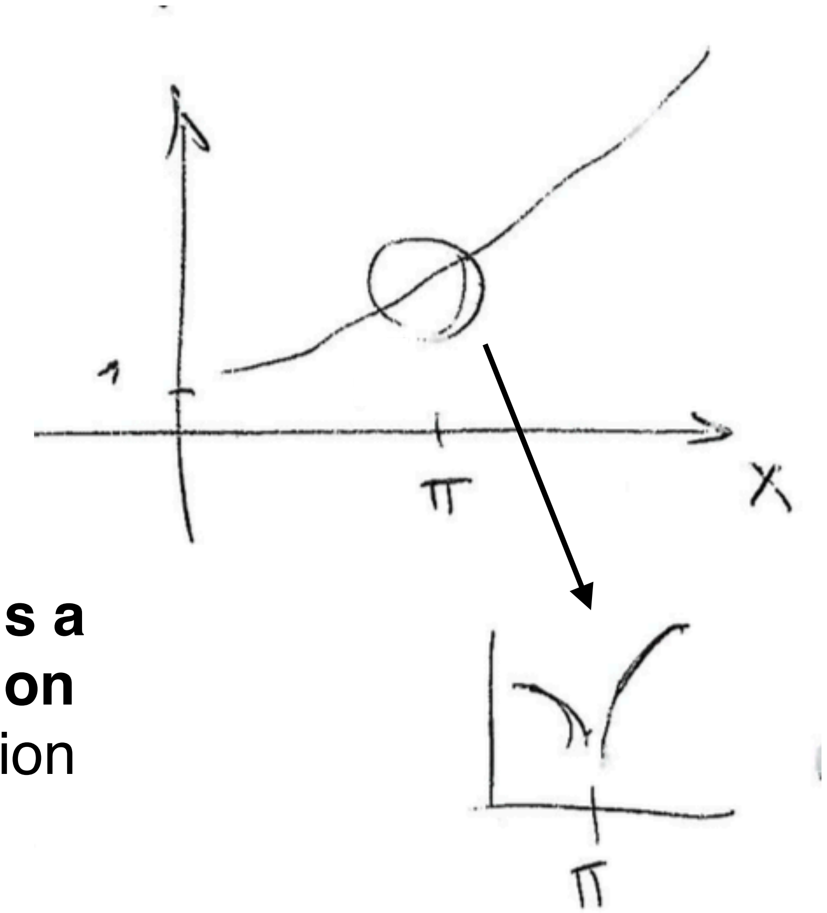


Exemples:

- polynomials
- rational functions
- trigonometric functions

Good model functions cannot solve pathological problems, e.g.,

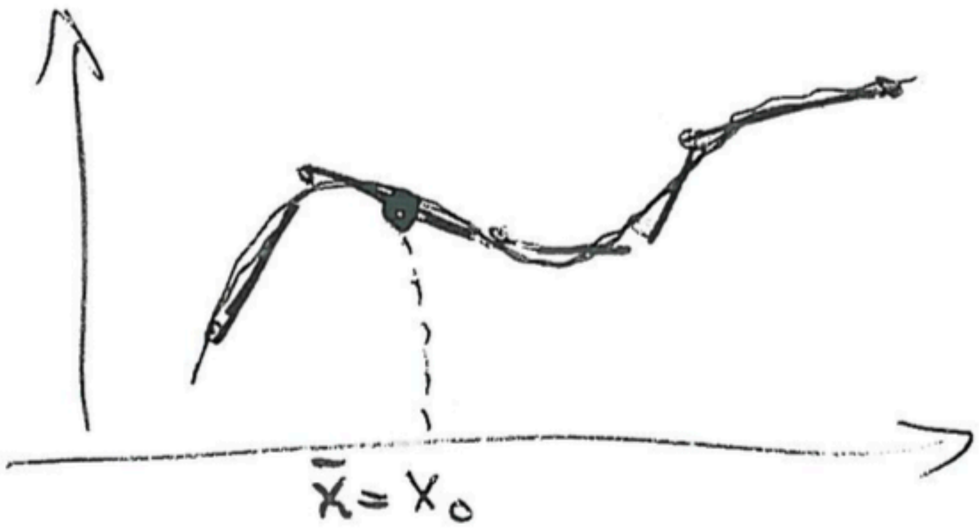
$$f(x) = 3x^2 + \frac{1}{\pi^4}\ln\left[(\pi - x)^2\right] + 1$$



In other words, **numerical interpolation and extrapolation is a well-posed mathematical problem if the underlying function is smooth**. If this is not the case, extrapolation and interpolation are not reliable.

Theoretically, there are two steps:

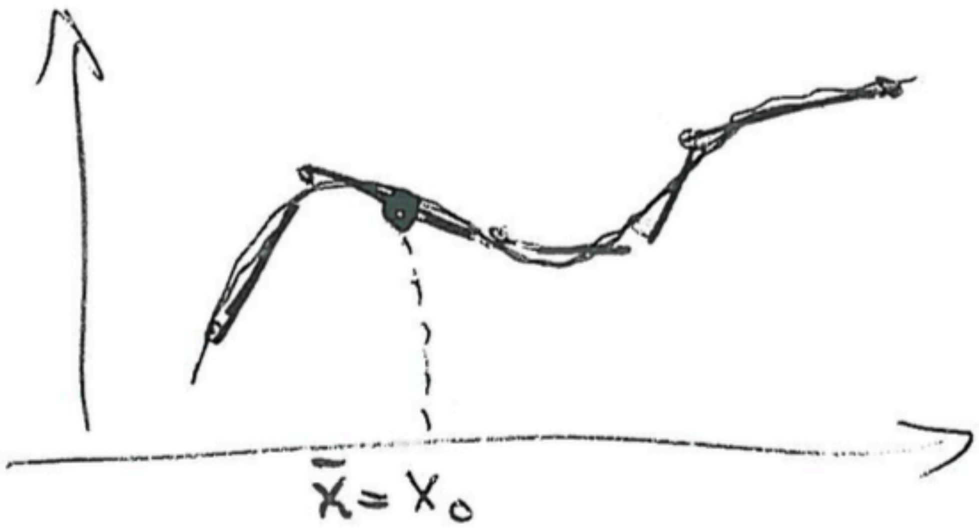
1) find an interpolating function at the assigned points
2) evaluate this function at the desired point x_0



In practice, it is preferable to combine step 1) and 2):
$f(x_0)$ is evaluate directly from $(f_1,f_2,\ldots,f_N)$ and $(x_1,x_2,\ldots,x_N)$.
In general, this takes something like $O(N^2)$ operations.

Theoretically, there are two steps:

1) find an interpolating function at the assigned points
2) evaluate this function at the desired point x_0



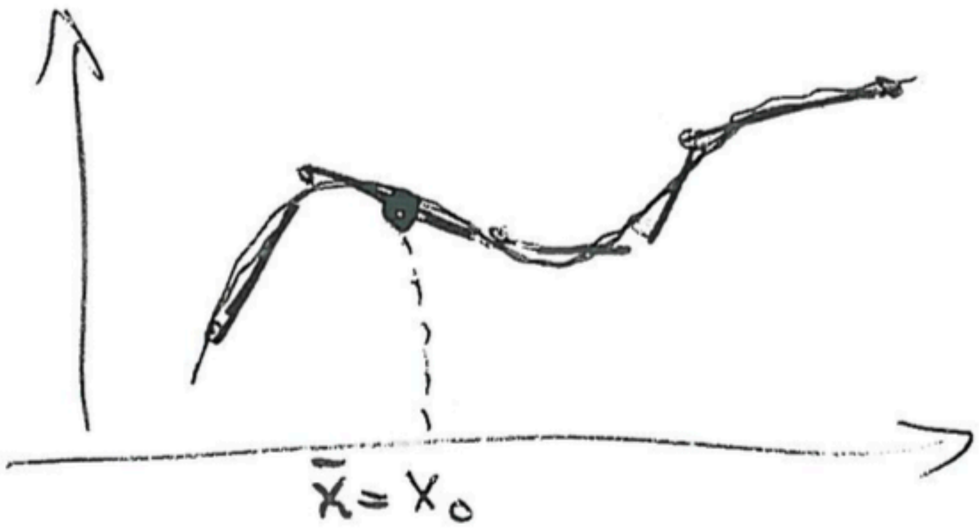
In practice, it is preferable to combine step 1) and 2):
$f(x_0)$ is evaluate directly from $(f_1,f_2,\ldots,f_N)$ and $(x_1,x_2,\ldots,x_N)$.
In general, this takes something like $O(N^2)$ operations.

There are two ways to do an interpolation:

A) LOCAL: coefficients are calculated only through the neighboring points to x_0
B) GLOBAL: coefficients are calculated globally (e.g., spline fit)

Theoretically, there are two steps:

1) find an interpolating function at the assigned points
2) evaluate this function at the desired point x_0



In practice, it is preferable to combine step 1) and 2):
$f(x_0)$ is evaluate directly from $(f_1,f_2,\ldots,f_N)$ and $(x_1,x_2,\ldots,x_N)$.
In general, this takes something like $O(N^2)$ operations.

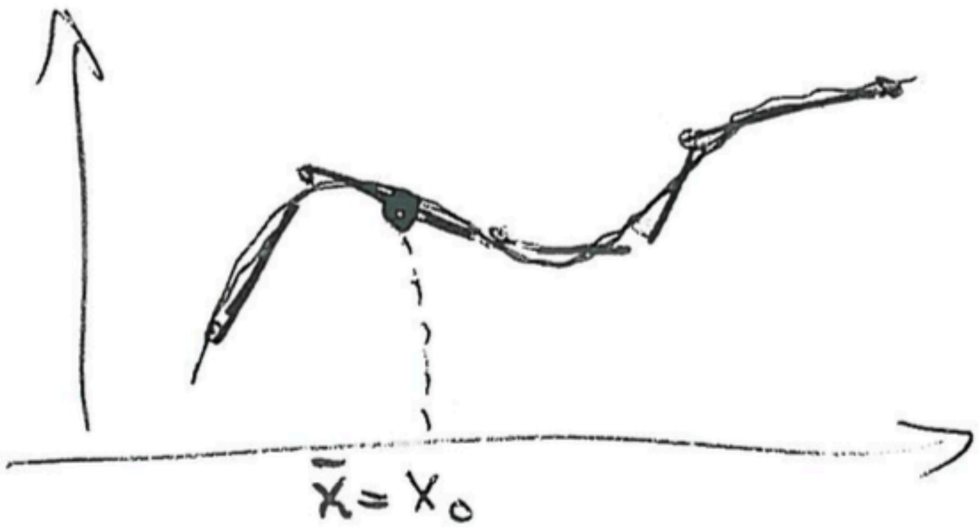
There are two ways to do an interpolation:

A) LOCAL: coefficients are calculated only through the neighboring points to x_0
B) GLOBAL: coefficients are calculated globally (e.g., spline fit)

A) PROs: very simple and efficient; CONs: might introduce discontinuities in the derivatives
B) PROs: there are going to be no discontinuities in the derivatives (by construction); CONs: more complicated and computationally expensive

Theoretically, there are two steps:

1) find an interpolating function at the assigned points
2) evaluate this function at the desired point x_0



In practice, it is preferable to combine step 1) and 2): $f(x_0)$ is evaluate directly from $(f_1,f_2,\ldots,f_N)$ and $(x_1,x_2,\ldots,x_N)$. In general, this takes something like $O(N^2)$ operations.

There are two ways to do an interpolation:

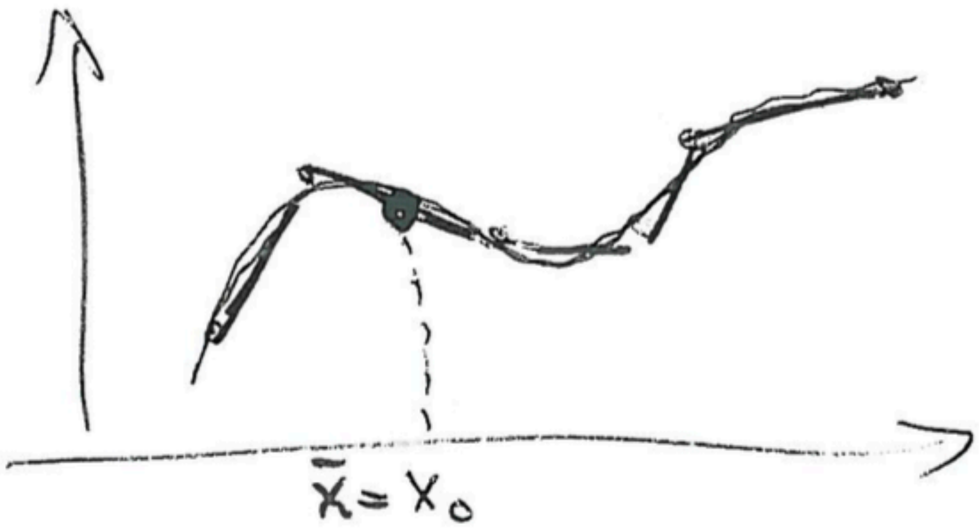
A) LOCAL: coefficients are calculated only through the neighboring points to x_0
B) GLOBAL: coefficients are calculated globally (e.g., spline fit)

A) PROs: very simple and efficient; CONs: might introduce discontinuities in the derivatives
B) PROs: there are going to be no discontinuities in the derivatives (by construction); CONs: more complicated and computationally expensive

Let's consider **polynomials** as modeling functions.
Q: what is a good order for polynomials?

Theoretically, there are two steps:

1) find an interpolating function at the assigned points
2) evaluate this function at the desired point x_0



In practice, it is preferable to combine step 1) and 2):
$f(x_0)$ is evaluate directly from $(f_1, f_2, \ldots, f_N)$ and $(x_1, x_2, \ldots, x_N)$.
In general, this takes something like $O(N^2)$ operations.

There are two ways to do an interpolation:

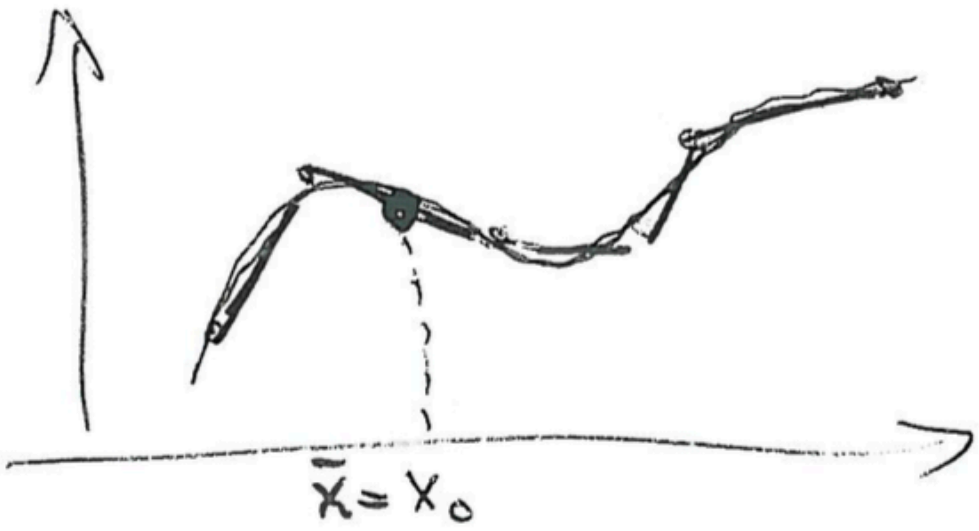
A) LOCAL: coefficients are calculated only through the neighboring points to x_0
B) GLOBAL: coefficients are calculated globally (e.g., spline fit)

A) PROs: very simple and efficient; CONs: might introduce discontinuities in the derivatives
B) PROs: there are going to be no discontinuities in the derivatives (by construction); CONs: more complicated and computationally expensive

Let's consider **polynomials** as modeling functions.
Q: what is a good order for polynomials?
A: it depends on the function

Theoretically, there are two steps:

1) find an interpolating function at the assigned points
2) evaluate this function at the desired point x_0



In practice, it is preferable to combine step 1) and 2): $f(x_0)$ is evaluate directly from $(f_1,f_2,\dots,f_N)$ and $(x_1,x_2,\dots,x_N)$. In general, this takes something like $O(N^2)$ operations.

There are two ways to do an interpolation:

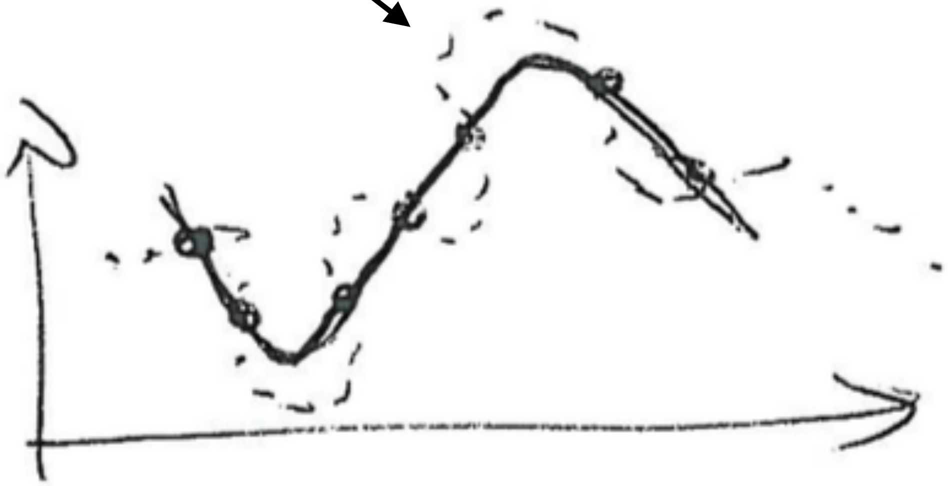
A) LOCAL: coefficients are calculated only through the neighboring points to x_0
B) GLOBAL: coefficients are calculated globally (e.g., spline fit)

A) PROs: very simple and efficient; CONs: might introduce discontinuities in the derivatives
B) PROs: there are going to be no discontinuities in the derivatives (by construction); CONs: more complicated and computationally expensive

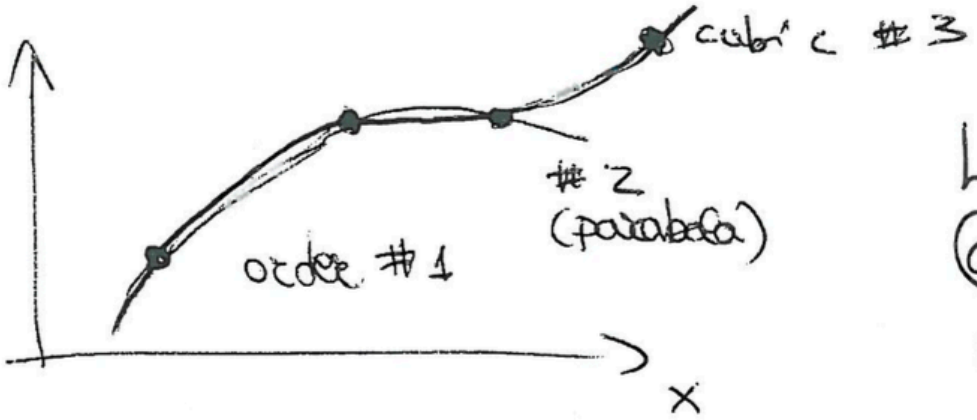
Let’s consider **polynomials** as modeling functions.
Q: what is a good order for polynomials?
A: it depends on the function

Ex. i) function with sharp corners (i.e., large gradients)
—> low order polynomial is a good idea

Ex. ii) function that is smooth —> use high order polynomial


with a high polynomial

For high order polynomials, never go past 5th order. If I have N points at which the function is tabulated, then the maximum order of the polynomial I can use is $N_{max} = N-1$

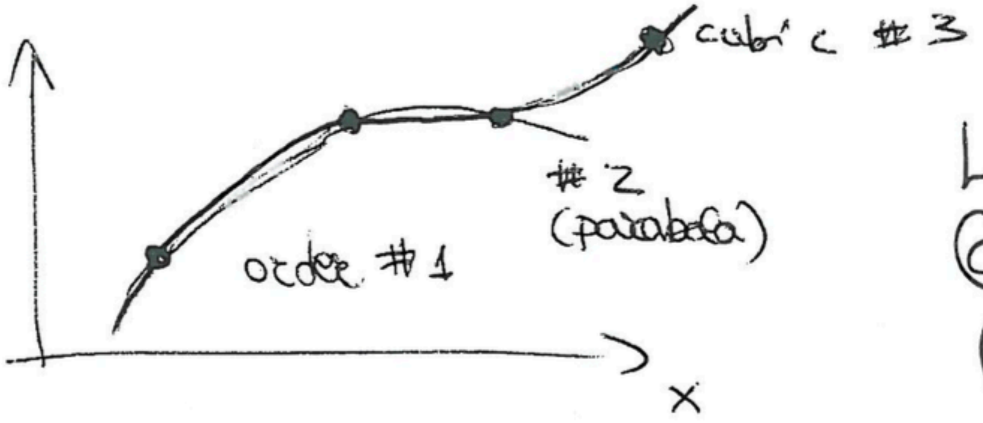


Linear $N_{max}=1$

Quadratic $N_{max}=2$

Cubic $N_{max}=3$

For high order polynomials, never go past 5th order. If I have N points at which the function is tabulated, then the maximum order of the polynomial I can use is $N_{max} = N-1$



Linear $N_{max}=1$

Quadratic $N_{max}=2$

Cubic $N_{max}=3$

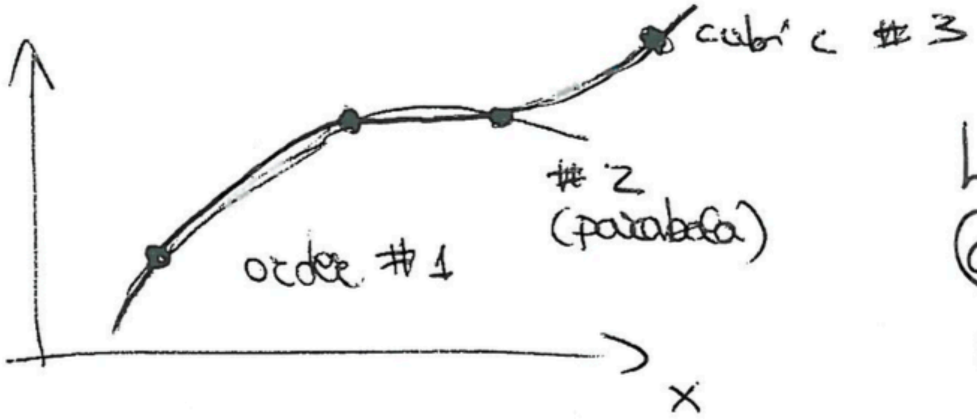
Lagrange's Formula:

Given $(x_1,x_2,\ldots,x_N)$; $(f_1,f_2,\ldots,f_N) = (y_1,y_2,\ldots,y_N)$

$$y(x) = \frac{(x-x_2)(x-x_3)...(x-x_N)}{(x_1-x_2)(x_1-x_3)...(x_1-x_N)}y_1 + ... + \frac{(x-x_1)(x-x_2)...(x-x_{N-1})}{(x_N-x_1)(x_N-x_2)...(x_N-x_{N-1})}y_N$$

point at which I want to interpolate

For high order polynomials, never go past 5th order. If I have N points at which the function is tabulated, then the maximum order of the polynomial I can use is $N_{max} = N-1$



Linear $N_{max}=1$

Quadratic $N_{max}=2$

Cubic $N_{max}=3$

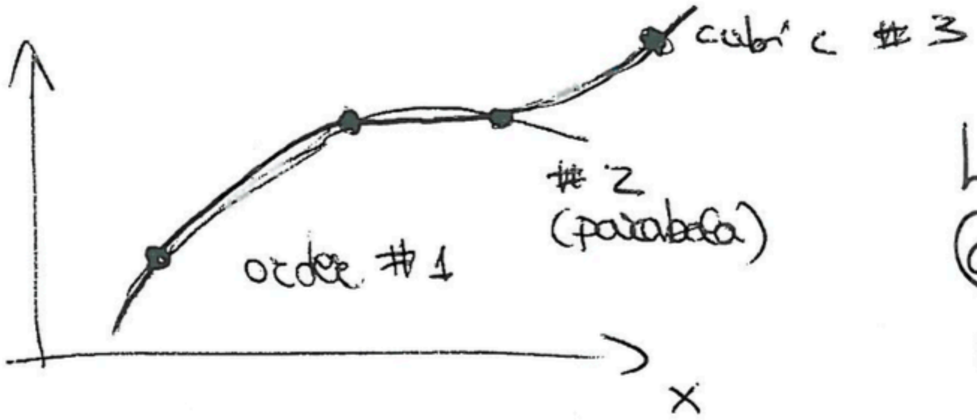
Lagrange's Formula:

Given $(x_1,x_2,\dots,x_N)$; $(f_1,f_2,\dots,f_N) = (y_1,y_2,\dots,y_N)$

$$y(x) = \frac{(x-x_2)(x-x_3)...(x-x_N)}{(x_1-x_2)(x_1-x_3)...(x_1-x_N)}y_1 + ... + \frac{(x-x_1)(x-x_2)...(x-x_{N-1})}{(x_N-x_1)(x_N-x_2)...(x_N-x_{N-1})}y_N$$

point at which I want to interpolate

For high order polynomials, never go past 5th order. If I have N points at which the function is tabulated, then the maximum order of the polynomial I can use is $N_{max} = N-1$



Linear $N_{max}=1$
Quadratic $N_{max}=2$
Cubic $N_{max}=3$

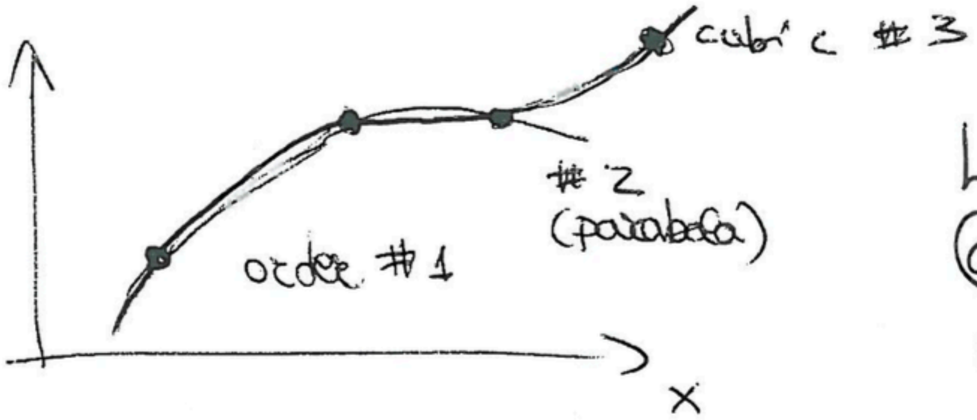
Lagrange's Formula:

Given $(x_1,x_2,\ldots,x_N)$; $(f_1,f_2,\ldots,f_N) = (y_1,y_2,\ldots,y_N)$

$$y(x) = \frac{(x-x_2)(x-x_3)\ldots(x-x_N)}{(x_1-x_2)(x_1-x_3)\ldots(x_1-x_N)}y_1 + \ldots + \frac{(x-x_1)(x-x_2)\ldots(x-x_{N-1})}{(x_N-x_1)(x_N-x_2)\ldots(x_N-x_{N-1})}y_N$$

point at which I want to interpolate

For high order polynomials, never go past 5th order. If I have N points at which the function is tabulated, then the maximum order of the polynomial I can use is N_{max} = N-1



Linear $N_{max}=1$
Quadratic $N_{max}=2$
Cubic $N_{max}=3$

Lagrange's Formula:

Given (x_1,x_2,…,x_N); (f_1,f_2,…,f_N) = (y_1,y_2,…,y_N)

$$y(x) = \frac{(x-x_2)(x-x_3)...(x-x_N)}{(x_1-x_2)(x_1-x_3)...(x_1-x_N)}y_1 + ... + \frac{(x-x_1)(x-x_2)...(x-x_{N-1})}{(x_N-x_1)(x_N-x_2)...(x_N-x_{N-1})}y_N$$

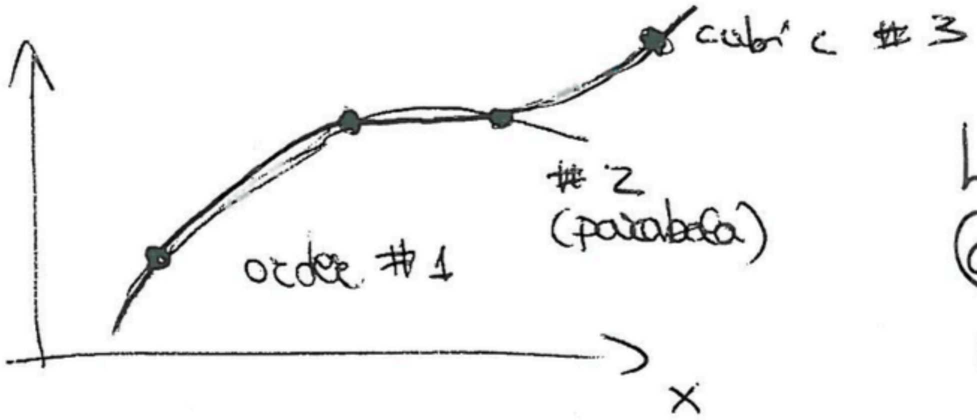
point at which I want to interpolate

EXAMPLE: N=2 (straight line)

$$y(x) = \frac{(x-x_2)}{(x_1-x_2)}y_1 - \frac{(x-x_1)}{(x_1-x_2)}y_2 = ty_1 + uy_2$$

$$where \quad t = \frac{x-x_2}{x_1-x_2}, \quad u = 1-t = -\frac{x-x_1}{x_1-x_2}$$

For high order polynomials, never go past 5th order. If I have N points at which the function is tabulated, then the maximum order of the polynomial I can use is $N_{max} = N-1$



Linear $N_{max}=1$
Quadratic $N_{max}=2$
Cubic $N_{max}=3$

Lagrange's Formula:

Given $(x_1,x_2,\ldots,x_N)$; $(f_1,f_2,\ldots,f_N) = (y_1,y_2,\ldots,y_N)$

$$y(x) = \frac{(x-x_2)(x-x_3)\ldots(x-x_N)}{(x_1-x_2)(x_1-x_3)\ldots(x_1-x_N)}y_1 + \ldots + \frac{(x-x_1)(x-x_2)\ldots(x-x_{N-1})}{(x_N-x_1)(x_N-x_2)\ldots(x_N-x_{N-1})}y_N$$

point at which I want to interpolate

EXAMPLE: N=2 (straight line)

$$y(x) = \frac{(x-x_2)}{(x_1-x_2)}y_1 - \frac{(x-x_1)}{(x_1-x_2)}y_2 = ty_1 + uy_2$$

$$where \quad t = \frac{x-x_2}{x_1-x_2}, \quad u = 1-t = -\frac{x-x_1}{x_1-x_2}$$

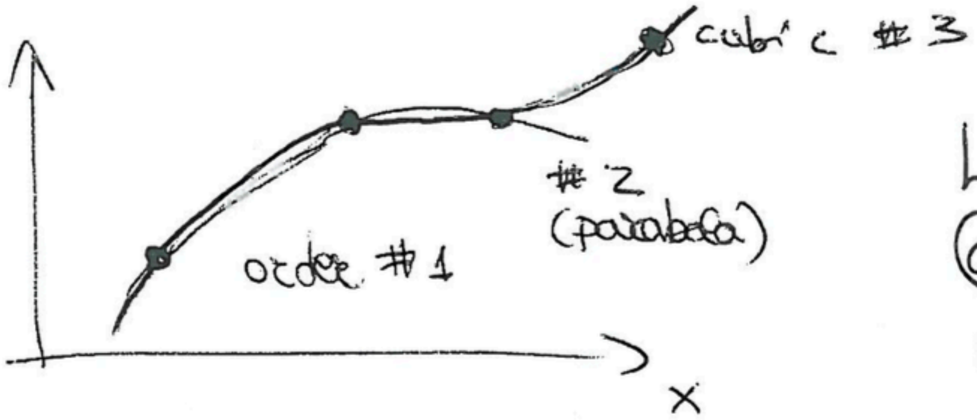
$$(x_1,y_1),(x_2,y_2)$$

$$y = \alpha x + \beta$$

$$\alpha = \frac{y_2-y_1}{x_2-x_1}$$

$$\beta = y_1 - \alpha x_1$$

For high order polynomials, never go past 5th order. If I have N points at which the function is tabulated, then the maximum order of the polynomial I can use is $N_{max} = N-1$



Linear $N_{max}=1$
Quadratic $N_{max}=2$
Cubic $N_{max}=3$

Lagrange's Formula:

Given $(x_1,x_2,\ldots,x_N)$; $(f_1,f_2,\ldots,f_N) = (y_1,y_2,\ldots,y_N)$

$$y(x) = \frac{(x-x_2)(x-x_3)\ldots(x-x_N)}{(x_1-x_2)(x_1-x_3)\ldots(x_1-x_N)}y_1 + \ldots + \frac{(x-x_1)(x-x_2)\ldots(x-x_{N-1})}{(x_N-x_1)(x_N-x_2)\ldots(x_N-x_{N-1})}y_N$$

point at which I want to interpolate

EXAMPLE: N=2 (straight line)

$$y(x) = \frac{(x-x_2)}{(x_1-x_2)}y_1 - \frac{(x-x_1)}{(x_1-x_2)}y_2 = ty_1 + uy_2$$

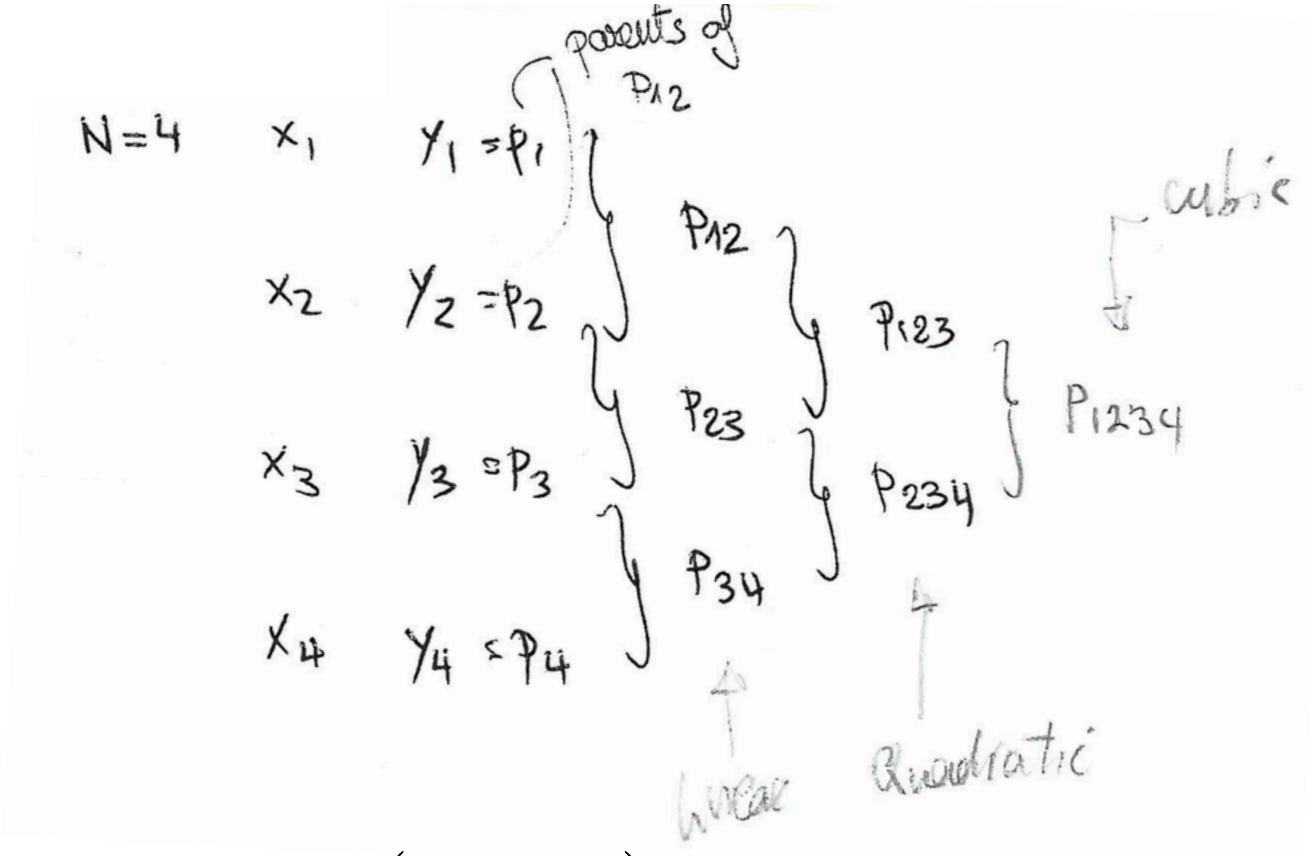
$$where \quad t = \frac{x-x_2}{x_1-x_2}, \quad u = 1-t = -\frac{x-x_1}{x_1-x_2}$$

$$\boxed{\begin{aligned} &(x_1,y_1),(x_2,y_2) \\ &y = \alpha x + \beta \\ &\qquad \alpha = \frac{y_2-y_1}{x_2-x_1} \\ &\qquad \beta = y_1 - \alpha x_1 \end{aligned}}$$

Lagrange's formula is fine mathematically, but it is not easy to implement numerically

Neville's Algorithm:

a smart implementation of Lagrange's formula

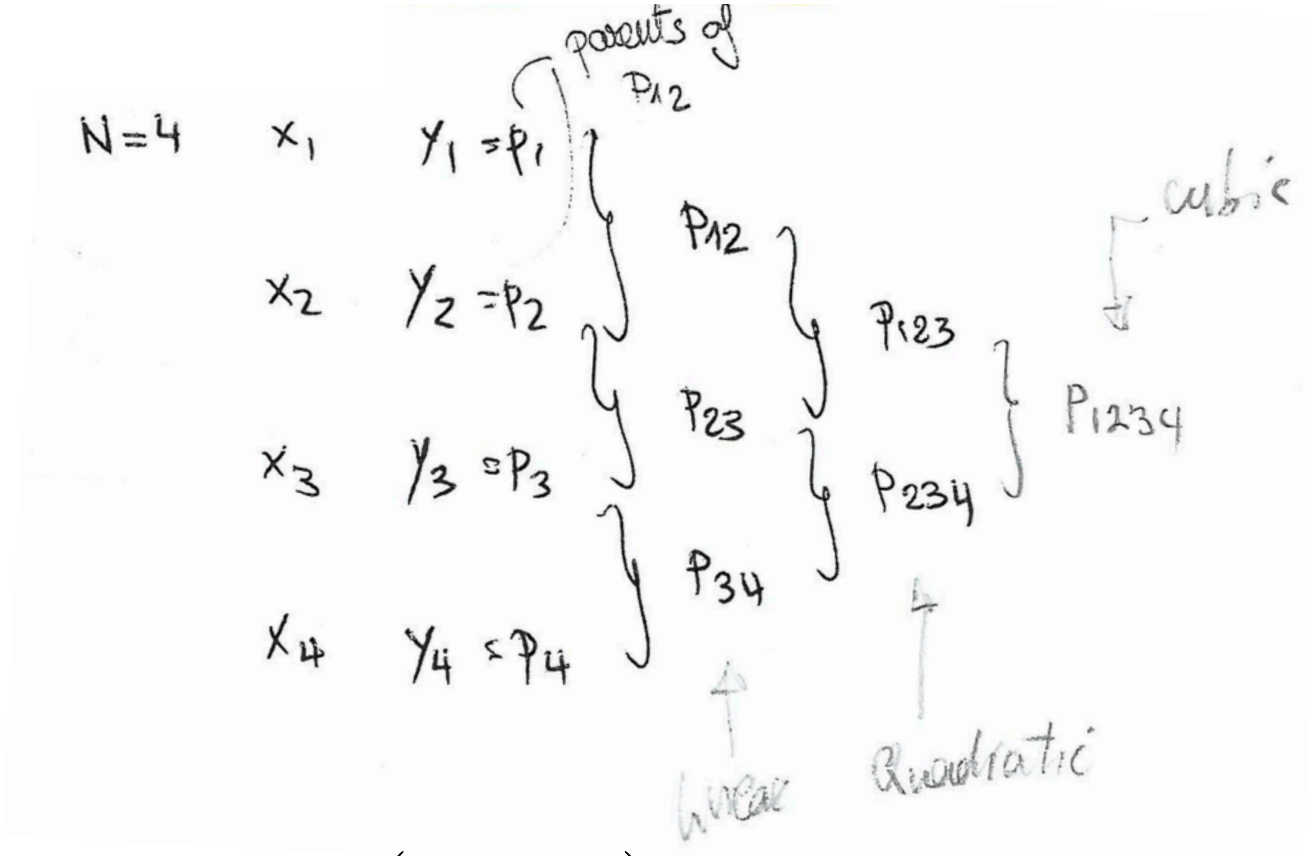


$$p_{i(i+1)\ldots(1+m)}(x) = \frac{(x - x_{i+m})p_{i(i+1)\ldots(i+m-1)} + (x_i - x)p_{(i+1)(i+2)\ldots(i+m)}}{x_i - x_{i+m}}$$

where m=N-1

Neville’s Algorithm:

a smart implementation of Lagrange’s formula



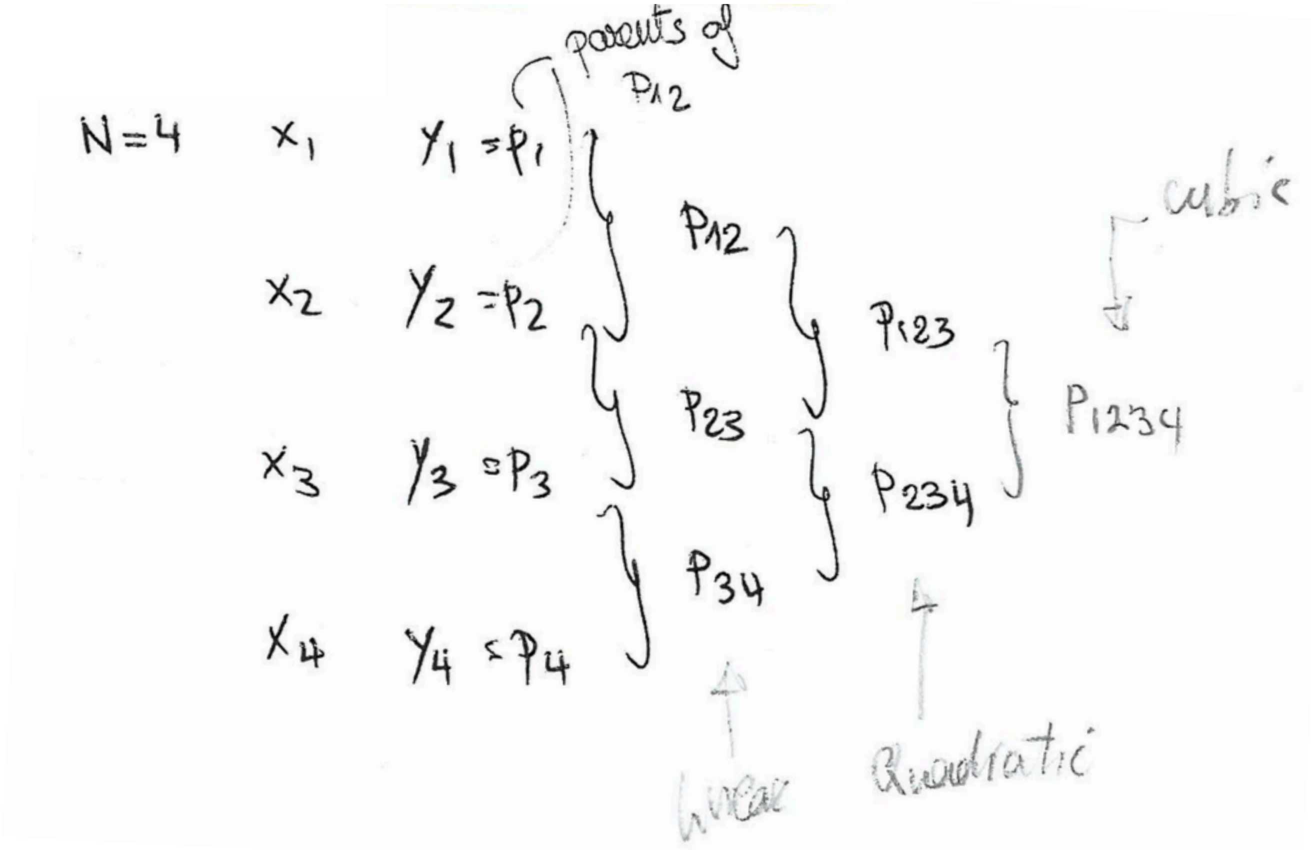
$$p_{i(i+1)\ldots(1+m)}(x) = \frac{(x - x_{i+m})p_{i(i+1)\ldots(i+m-1)} + (x_i - x)p_{(i+1)(i+2)\ldots(i+m)}}{x_i - x_{i+m}}$$

where m=N-1

In other words, the final polynomial is just a linear combination of lower order polynomials:

Neville's Algorithm:

a smart implementation of Lagrange's formula



$$p_{i(i+1)\dots(1+m)}(x) = \frac{(x-x_{i+m})p_{i(i+1)\dots(i+m-1)} + (x_i - x)p_{(i+1)(i+2)\dots(i+m)}}{x_i - x_{i+m}}$$

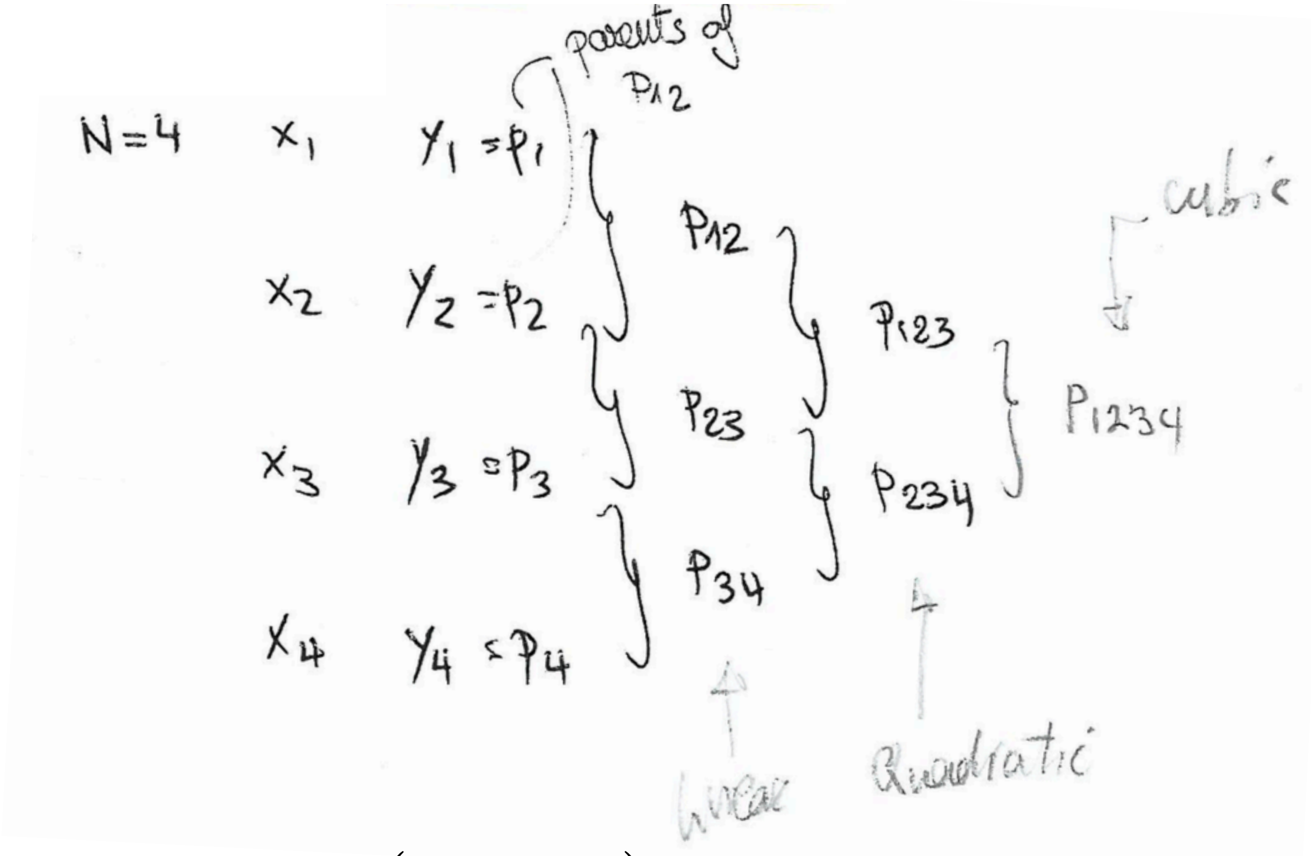
where m=N-1

In other words, the final polynomial is just a linear combination of lower order polynomials:

$$p_{1234}(x) = \frac{(x-x_4)p_{123} + (x_1 - x)p_{234}}{x_1 - x_4}$$ cubic

Neville's Algorithm:

a smart implementation of Lagrange's formula



$$p_{i(i+1)\ldots(1+m)}(x) = \frac{(x - x_{i+m})p_{i(i+1)\ldots(i+m-1)} + (x_i - x)p_{(i+1)(i+2)\ldots(i+m)}}{x_i - x_{i+m}}$$

where m=N-1

In other words, the final polynomial is just a linear combination of lower order polynomials:

$$p_{1234}(x) = \frac{(x - x_4)p_{123} + (x_1 - x)p_{234}}{x_1 - x_4}$$

cubic

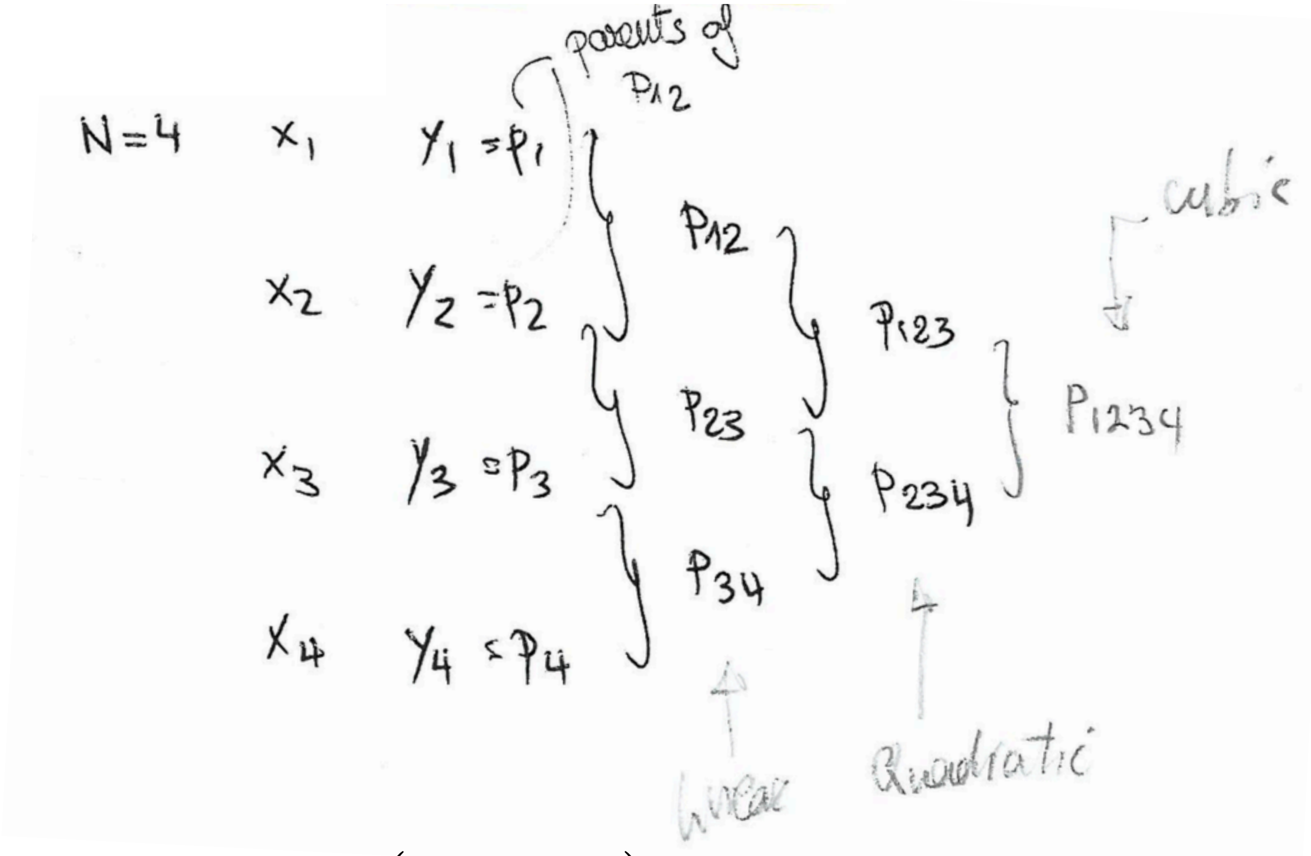
$$p_{123}(x) = \frac{(x - x_3)p_{12} + (x_1 - x)p_{23}}{x_1 - x_3}$$

$$p_{234}(x) = \frac{(x - x_4)p_{23} + (x_2 - x)p_{34}}{x_2 - x_4}$$

parabola

Neville's Algorithm:

a smart implementation of Lagrange's formula



$$p_{i(i+1)\dots(1+m)}(x) = \frac{(x - x_{i+m})p_{i(i+1)\dots(i+m-1)} + (x_i - x)p_{(i+1)(i+2)\dots(i+m)}}{x_i - x_{i+m}}$$

where m=N-1

In other words, the final polynomial is just a linear combination of lower order polynomials:

$$p_{1234}(x) = \frac{(x - x_4)p_{123} + (x_1 - x)p_{234}}{x_1 - x_4}$$

cubic

$$p_{123}(x) = \frac{(x - x_3)p_{12} + (x_1 - x)p_{23}}{x_1 - x_3} \qquad p_{234}(x) = \frac{(x - x_4)p_{23} + (x_2 - x)p_{34}}{x_2 - x_4}$$

parabola

$$p_{12}(x) = \frac{(x - x_2)p_1 + (x_1 - x)p_2}{x_1 - x_2} \qquad p_{34}(x) = \frac{(x - x_4)p_3 + (x_3 - x)p_4}{x_3 - x_4}$$

straight line

$$p_{23}(x) = \frac{(x - x_3)p_2 + (x_2 - x)p_3}{x_2 - x_3}$$

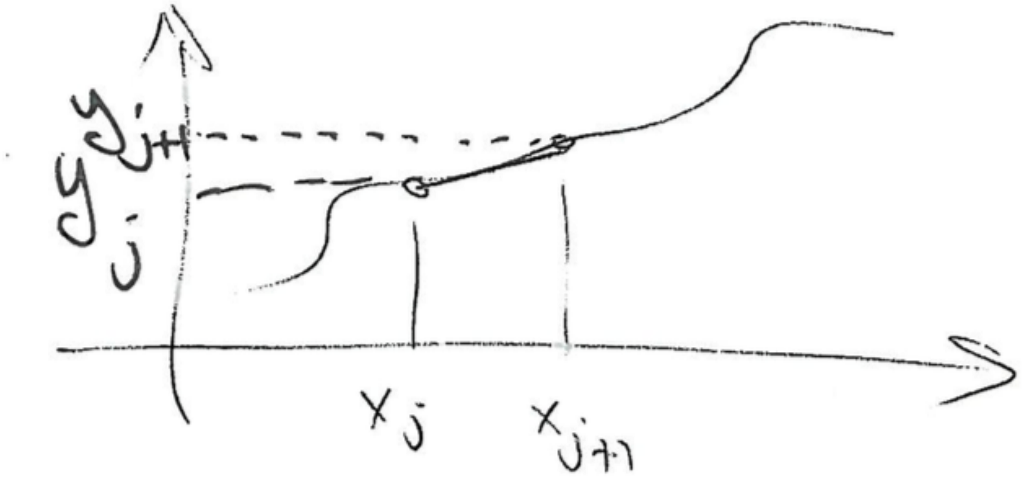
So far, we’ve been talking about local interpolation methods.

Global interpolation methods: cubic spline [NOTE: you can call this function/routine when coding]

So far, we've been talking about local interpolation methods.

Global interpolation methods: cubic spline [NOTE: you can call this function/routine when coding]

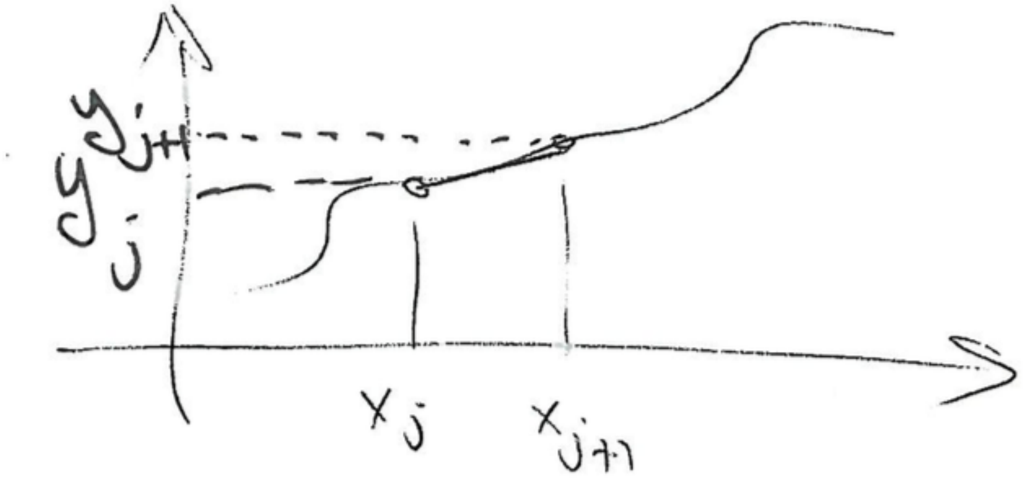
Given (x_1,x_2,…,x_N) and (f_1,f_2,…,f_N) = (y_1,y_2,…,y_N), consider two points x_j,x_{j+1} and write a linear interpolation formula for there two points (this is a special case of the general Lagrange interpolation formula):



So far, we've been talking about local interpolation methods.

Global interpolation methods: cubic spline [NOTE: you can call this function/routine when coding]

Given (x_1,x_2,…,x_N) and (f_1,f_2,…,f_N) = (y_1,y_2,…,y_N), consider two points x_j,x_{j+1} and write a linear interpolation formula for there two points (this is a special case of the general Lagrange interpolation formula):

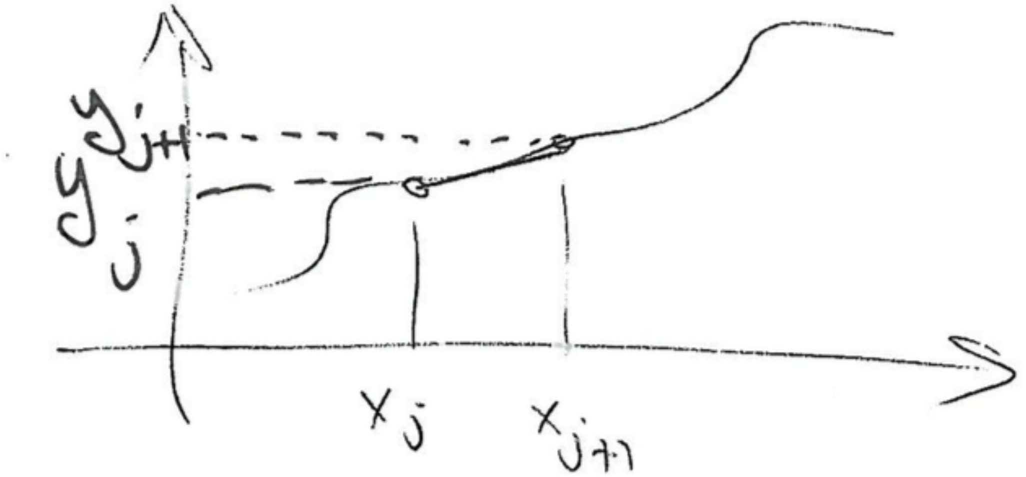


$$y(x) = Ay_j + By_{j+1}$$

$$A = A(x) = \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B = 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$

So far, we've been talking about local interpolation methods.

Global interpolation methods: cubic spline [NOTE: you can call this function/routine when coding]



Given (x_1,x_2,…,x_N) and (f_1,f_2,…,f_N) = (y_1,y_2,…,y_N), consider two points x_j,x_{j+1} and write a linear interpolation formula for there two points (this is a special case of the general Lagrange interpolation formula):

$$y(x) = Ay_j + By_{j+1}$$

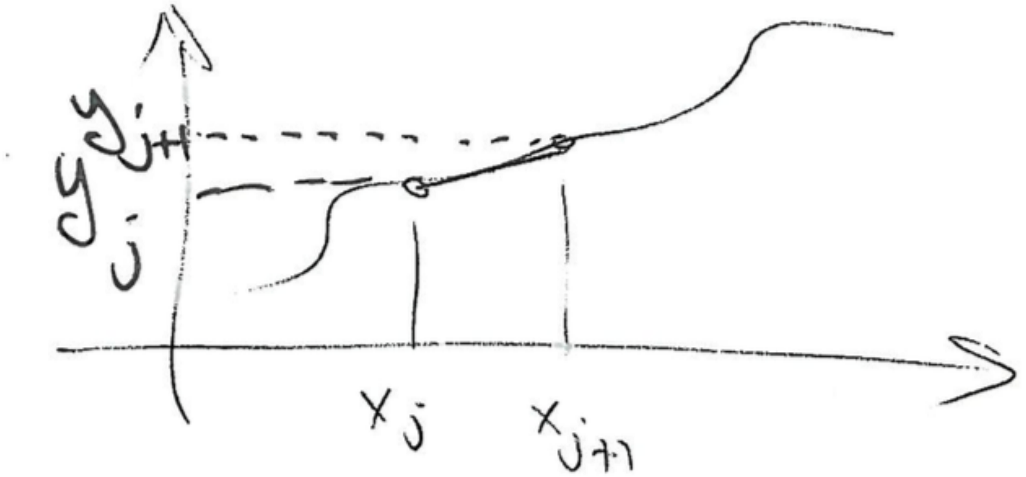
$$A = A(x) = \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B = 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$

This has y"=0 in the interior of each interval (x_j,x_{j+1}), and undefined or infinite y" at x=x_j. However, <u>we want an interpolating formula that is smooth in y', and continuous in y", both within and at the boundaries</u>.

So far, we've been talking about local interpolation methods.

Global interpolation methods: cubic spline [NOTE: you can call this function/routine when coding]

Given (x_1,x_2,…,x_N) and (f_1,f_2,…,f_N) = (y_1,y_2,…,y_N), consider two points x_j,x_{j+1} and write a linear interpolation formula for there two points (this is a special case of the general Lagrange interpolation formula):



$$y(x) = Ay_j + By_{j+1}$$

$$A = A(x) = \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B = 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$

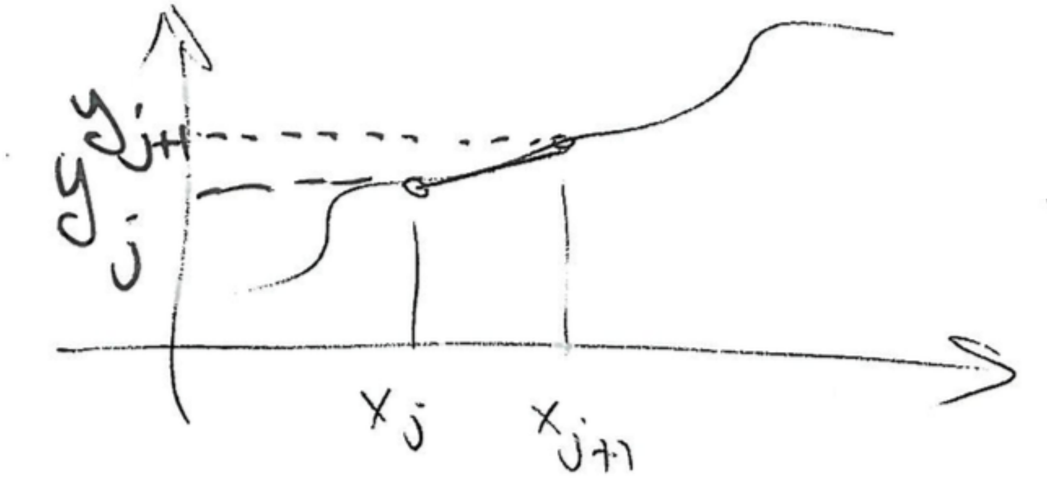
This has y"=0 in the interior of each interval (x_j,x_{j+1}), and undefined or infinite y" at x=x_j. However, we want an interpolating formula that is smooth in y', and continuous in y", both within and at the boundaries.

Suppose now we want to write an interpolating cubic using only the points (x_j,x_{j+1}):

So far, we've been talking about local interpolation methods.

Global interpolation methods: cubic spline [NOTE: you can call this function/routine when coding]

Given ($x_1,x_2,\ldots,x_N$) and ($f_1,f_2,\ldots,f_N$) = ($y_1,y_2,\ldots,y_N$), consider two points x_j,x_{j+1} and write a linear interpolation formula for there two points (this is a special case of the general Lagrange interpolation formula):



$$y(x) = Ay_j + By_{j+1}$$

$$A = A(x) = \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B = 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$

This has y"=0 in the interior of each interval (x_j,x_{j+1}), and undefined or infinite y" at $x=x_j$. However, we want an interpolating formula that is smooth in y', and continuous in y", both within and at the boundaries.

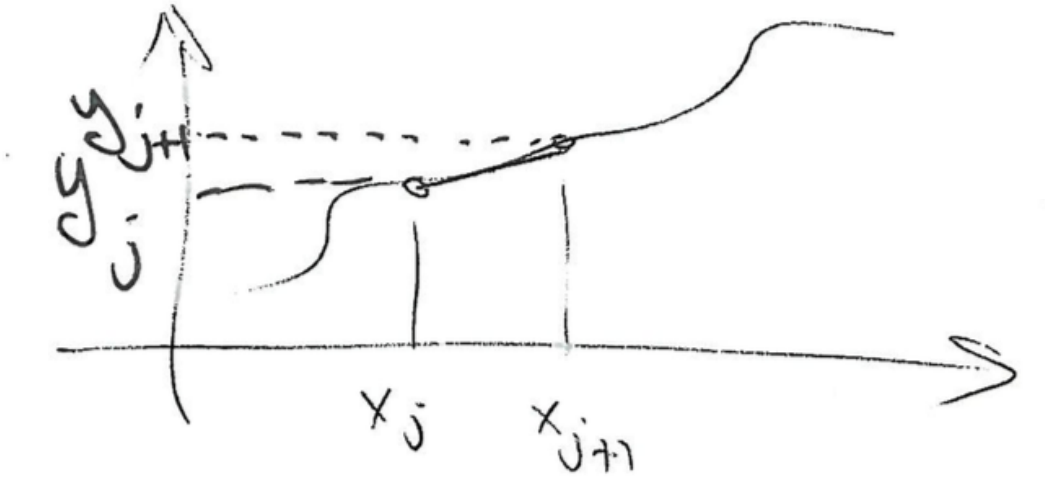
Suppose now we want to write an interpolating cubic using only the points (x_j,x_{j+1}):

$$y(x) = Ay_j + By_{j+1} + Cy''_j + Dy''_{j+1}$$ ★

So far, we've been talking about local interpolation methods.

Global interpolation methods: cubic spline [NOTE: you can call this function/routine when coding]

Given (x_1,x_2,…,x_N) and (f_1,f_2,…,f_N) = (y_1,y_2,…,y_N), consider two points x_j,x_{j+1} and write a linear interpolation formula for there two points (this is a special case of the general Lagrange interpolation formula):



$$y(x) = Ay_j + By_{j+1}$$

$$A = A(x) = \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B = 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$

This has y"=0 in the interior of each interval (x_j,x_{j+1}), and undefined or infinite y" at x=x_j. However, we want an interpolating formula that is smooth in y', and continuous in y", both within and at the boundaries.

Suppose now we want to write an interpolating cubic using only the points (x_j,x_{j+1}):

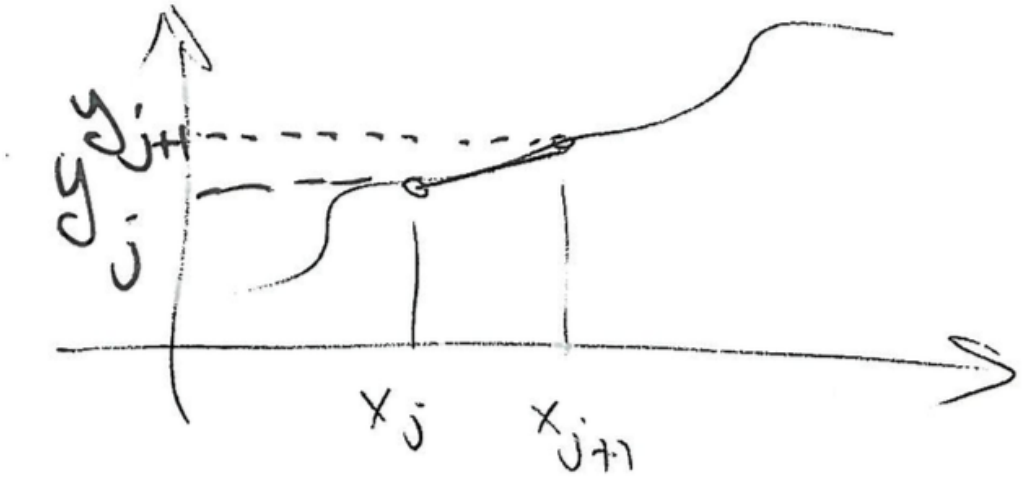
$$y(x) = Ay_j + By_{j+1} + Cy_j'' + Dy_{j+1}'' \quad ★$$

imagine these are tabulated (given)

So far, we've been talking about local interpolation methods.

Global interpolation methods: cubic spline [NOTE: you can call this function/routine when coding]

Given (x_1,x_2,…,x_N) and (f_1,f_2,…,f_N) = (y_1,y_2,…,y_N), consider two points x_j,x_{j+1} and write a linear interpolation formula for there two points (this is a special case of the general Lagrange interpolation formula):



$$y(x) = Ay_j + By_{j+1}$$

$$A = A(x) = \frac{x_{j+1} - x}{x_{j+1} - x_j} \qquad B = 1 - A = \frac{x - x_j}{x_{j+1} - x_j}$$

This has y"=0 in the interior of each interval (x_j,x_{j+1}), and undefined or infinite y" at x=x_j. However, we want an interpolating formula that is smooth in y', and continuous in y", both within and at the boundaries.

Suppose now we want to write an interpolating cubic using only the points (x_j,x_{j+1}):

$$y(x) = Ay_j + By_{j+1} + Cy''_j + Dy''_{j+1}$$ ★

imagine these are tabulated (given)

where:
A=A(x), B=B(x),
C=C(x^3), D=D(x^3),
hence the name
cubic spline

We choose C and D so that $y(x_j)=y_j$ and $y(x_{j+1})=y_{j+1}$, and the cubic polynomial has zero values when calculated at x_j and x_{j+1}.

We choose C and D so that $y(x_j)=y_j$ and $y(x_{j+1})=y_{j+1}$, and the cubic polynomial has zero values when calculated at x_j and x_{j+1}.

Then, one can demonstrate that:

$$C = \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2$$

$$D = \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2$$

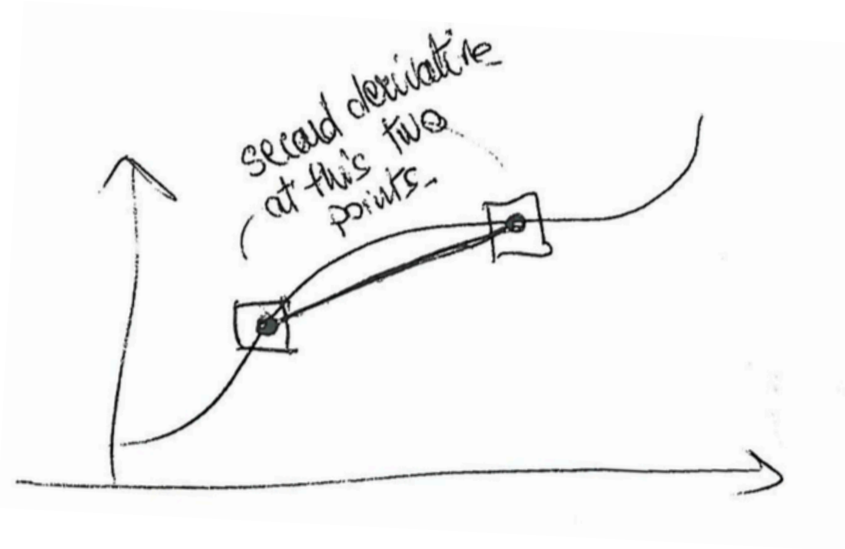
We choose C and D so that $y(x_j)=y_j$ and $y(x_{j+1})=y_{j+1}$, and the cubic polynomial has zero values when calculated at x_j and x_{j+1}.

Then, one can demonstrate that:

$$C = \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2$$

$$D = \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2$$

Q: what are $y_j'' = y''(x_j)$ and $y_{j+1}'' = y''(x_{j+1})$?



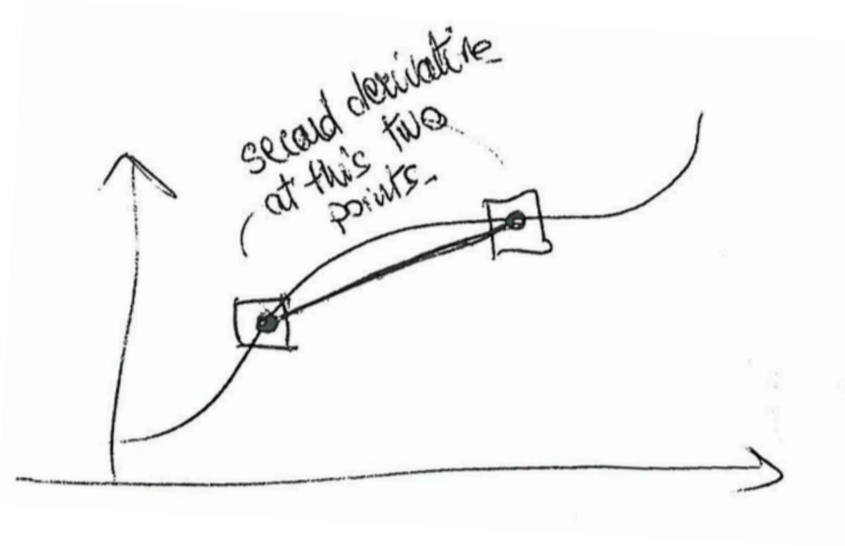
We choose C and D so that $y(x_j)=y_j$ and $y(x_{j+1})=y_{j+1}$, and the cubic polynomial has zero values when calculated at x_j and x_{j+1}.

Then, one can demonstrate that:

$$C = \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2$$

$$D = \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2$$

Q: what are $y''_j = y''(x_j)$ and $y''_{j+1} = y''(x_{j+1})$?



A: to calculate them, let's take the derivatives of the expression ★

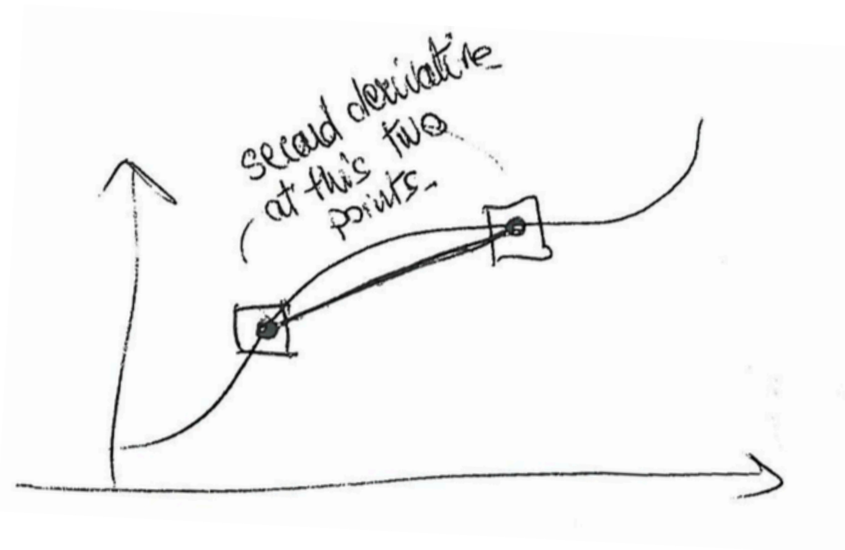
We choose C and D so that $y(x_j)=y_j$ and $y(x_{j+1})=y_{j+1}$, and the cubic polynomial has zero values when calculated at x_j and x_{j+1}.

Then, one can demonstrate that:

$$C = \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2$$

$$D = \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2$$

Q: what are $y_j'' = y''(x_j)$ and $y_{j+1}'' = y''(x_{j+1})$?



A: to calculate them, let's take the derivatives of the expression ★

🟦 $$\frac{dy}{dx} = y' = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{3A^2 - 1}{6}(x_{j+1} - x_j)y_j'' + \frac{3B^2 - 1}{6}(x_{j+1} - x_j)y_{j+1}''$$

since $A' = \frac{dA}{dx} = \frac{1}{x_{j+1} - x_j}$ $\quad B' = \frac{dB}{dx} = A'$ $\quad C' = \frac{dC}{dx} = \frac{3A^2A' - A'}{6}(x_{j+1} - x_j)^2$

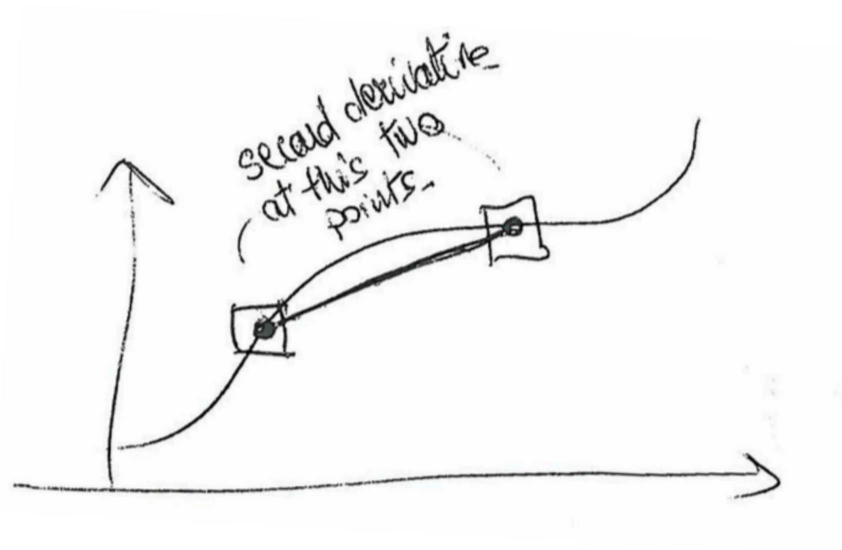
We choose C and D so that $y(x_j)=y_j$ and $y(x_{j+1})=y_{j+1}$, and the cubic polynomial has zero values when calculated at x_j and x_{j+1}.

Then, one can demonstrate that:

$$C = \frac{1}{6}(A^3 - A)(x_{j+1} - x_j)^2$$

$$D = \frac{1}{6}(B^3 - B)(x_{j+1} - x_j)^2$$

Q: what are $y''_j = y''(x_j)$ and $y''_{j+1} = y''(x_{j+1})$?



A: to calculate them, let's take the derivatives of the expression ★

■

$$\frac{dy}{dx} = y' = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{3A^2 - 1}{6}(x_{j+1} - x_j)y''_j + \frac{3B^2 - 1}{6}(x_{j+1} - x_j)y''_{j+1}$$

since $A' = \frac{dA}{dx} = \frac{1}{x_{j+1} - x_j}$ $\quad B' = \frac{dB}{dx} = A'$ $\quad C' = \frac{dC}{dx} = \frac{3A^2A' - A'}{6}(x_{j+1} - x_j)^2$

$$\frac{d^2y}{dx^2} = y'' = -\frac{6AA'}{6}(x_{j+1} - x_j)y''_j + \frac{6BB'}{6}(x_{j+1} - x_j)y''_{j+1} = Ay''_j + By''_{j+1}$$

since $A(x_j) = 1 \quad A(x_{j+1}) = 0 \quad B(x_j) = 0 \quad B(x_{j+1}) = 1$

→ $C(x_j)=0$, $C(x_{j+1})=0$, $D(x_j)=0$, $D(x_{j+1})=0$, i.e., we can give any values we want to y_j'' and y_{j+1}''

In other words, we don’t need to give any specific value to them, because they are not needed to ensure that the interpolating function passes through y_j and y_{j+1}.

➡ C(x_j)=0, C(x_{j+1})=0, D(x_j)=0, D(x_{j+1})=0, i.e., we can give any values we want to y''_j and y''_{j+1}

In other words, we don't need to give any specific value to them, because they are not needed to ensure that the interpolating function passes through y_j and y_{j+1}.

However, there is a condition we can enforce: continuity of the first derivatives, i.e.,

$$\frac{dy}{dx} = y'|_{x_j^-} = y'|_{x_j^+}$$

GLOBAL condition

Computed using x_{j-1} and x_j

Computed using x_{j+1} and x_j

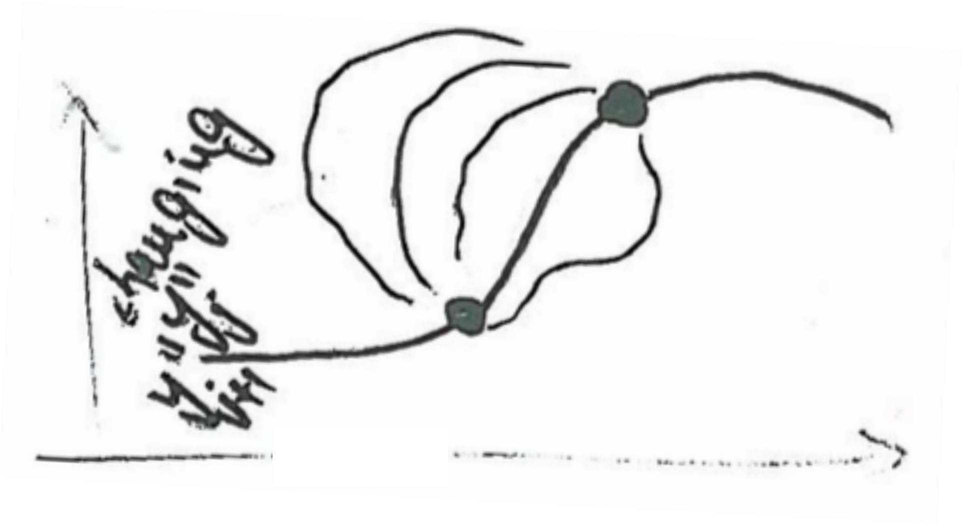
→ $C(x_j)=0$, $C(x_{j+1})=0$, $D(x_j)=0$, $D(x_{j+1})=0$, i.e., we can give any values we want to y''_j and y''_{j+1}

In other words, we don't need to give any specific value to them, because they are not needed to ensure that the interpolating function passes through y_j and y_{j+1}.

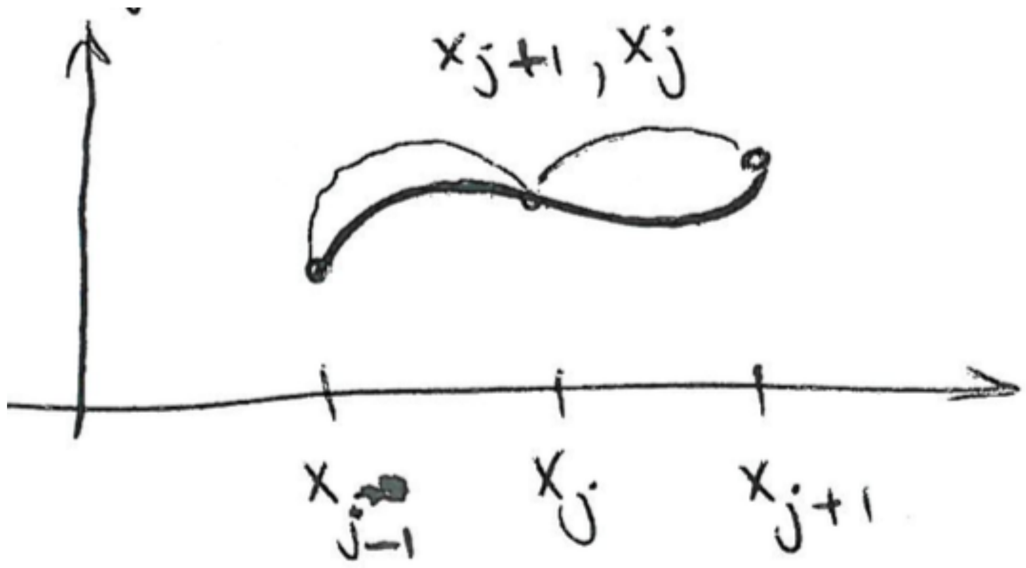
However, there is a condition we can enforce: continuity of the first derivatives, i.e.,

$$\frac{dy}{dx} = \underbrace{y'|_{x_j^-}}_{\text{Computed using } x_{j-1} \text{ and } x_j} = \underbrace{y'|_{x_j^+}}_{\text{Computed using } x_{j+1} \text{ and } x_j}$$

GLOBAL condition



LOCALLY, for each $[x_j,x_{j+1}]$, I create a cubic, but this way I end up with different splines for each interval (local condition).



By setting the continuity of the first derivatives at x_j, I have a GLOBAL spline (cubic spline) and a smooth function (global).

$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

➡

$$\frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}$$

$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

$$\Rightarrow \quad \frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}$$

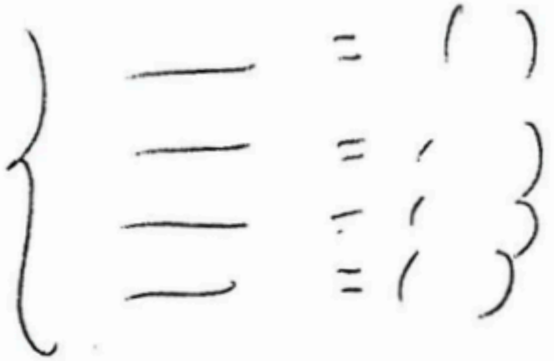
Repeating this also for j=2, …, N-1:

$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

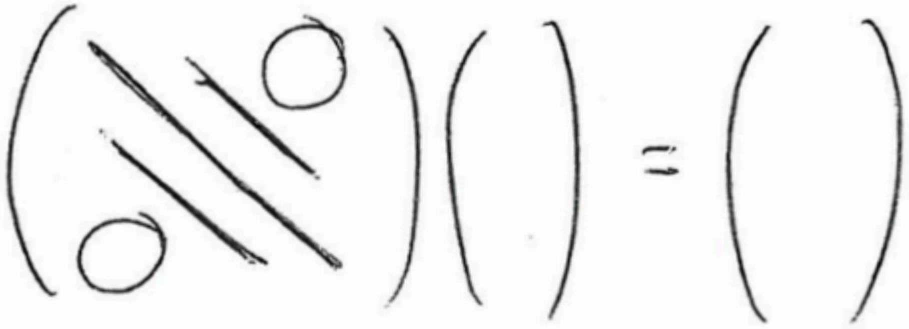
$$\rightarrow \frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}$$

Repeating this also for j=2, …, N-1:



↔

tridiagonal system
(set of equations)

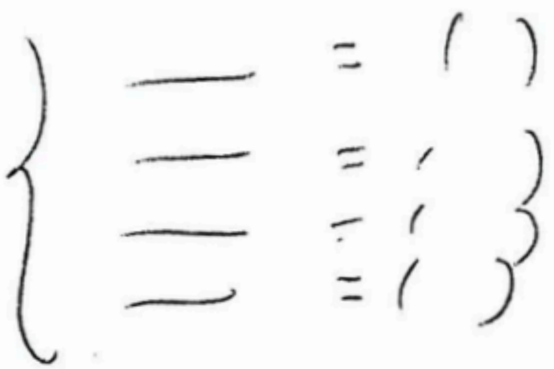


$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

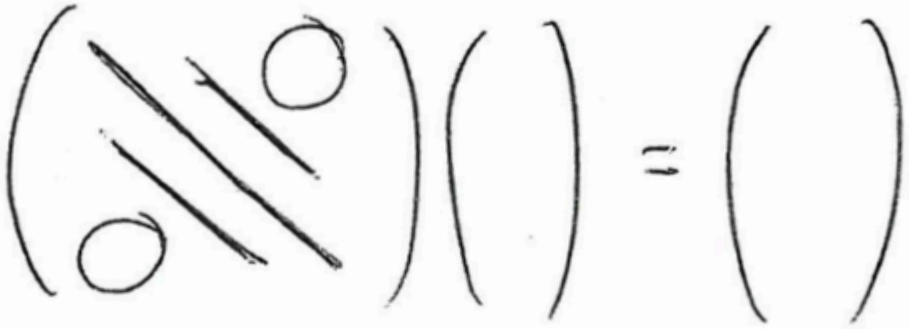
$$\rightarrow \quad \frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}$$

Repeating this also for j=2, …, N-1:

▲ 

↔ tridiagonal system (set of equations)

NxN tridiagonal matrix

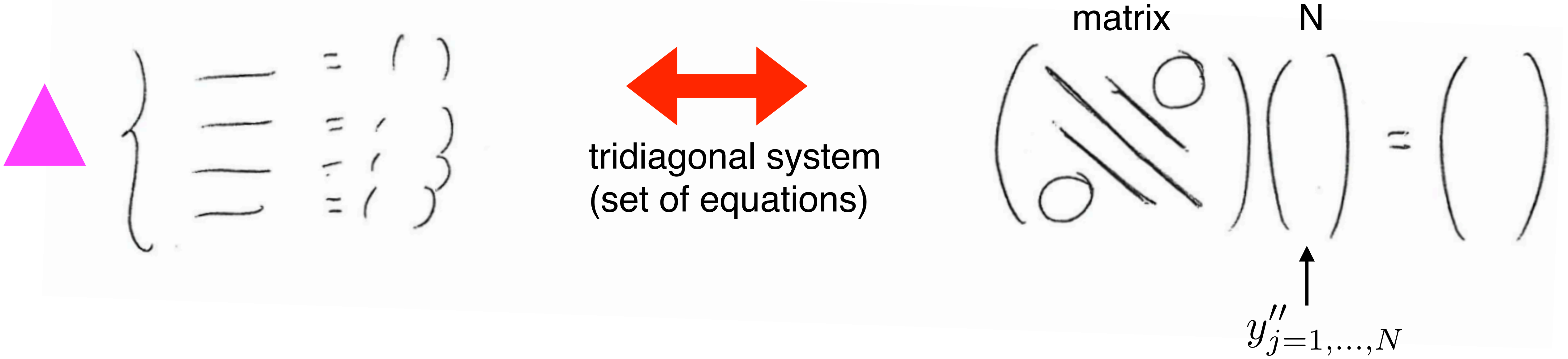


$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

➡ $$\frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}$$

Repeating this also for j=2, …, N-1:

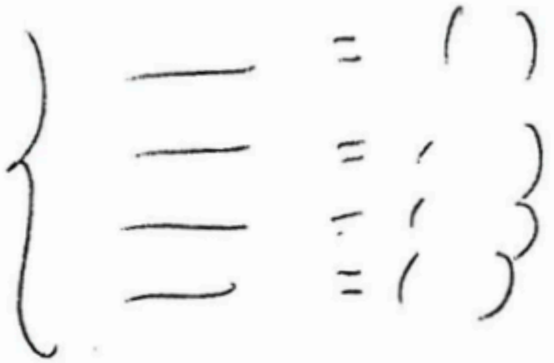


$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

$$\Rightarrow \frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \boxed{\frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}}$$

Repeating this also for j=2, …, N-1:

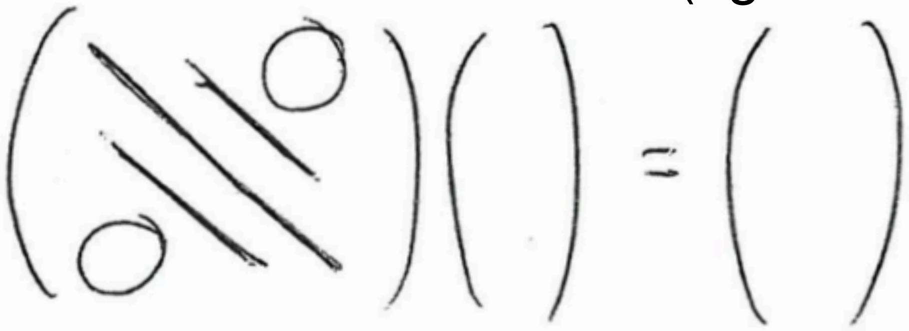


tridiagonal system
(set of equations)

NxN
tridiagonal
matrix

N

N
coefficients
(right hand)



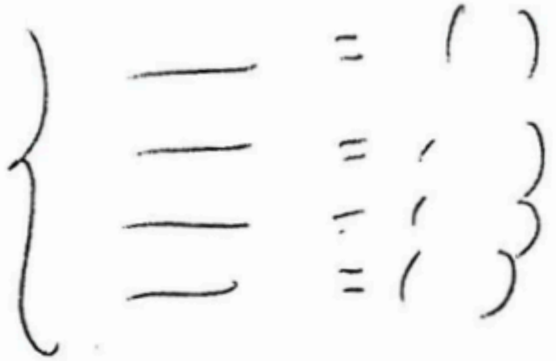
$y''_{j=1,\ldots,N}$

$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

$$\Rightarrow \frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \boxed{\frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}}$$

Repeating this also for j=2, …, N-1:

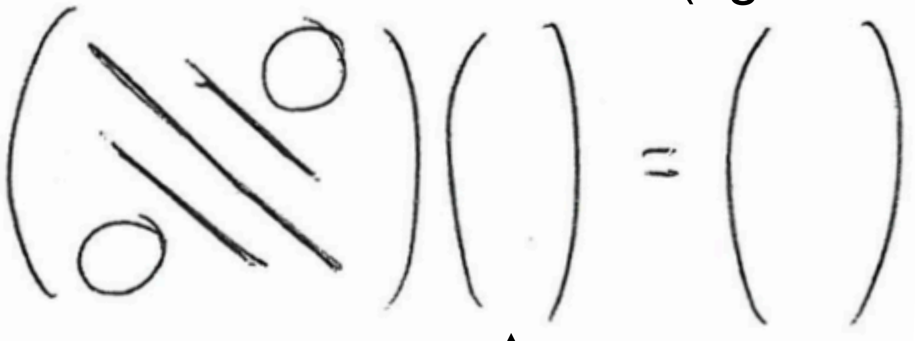


tridiagonal system
(set of equations)

NxN
tridiagonal
matrix

N

N
coefficients
(right hand)



$y''_{j=1,\dots,N}$

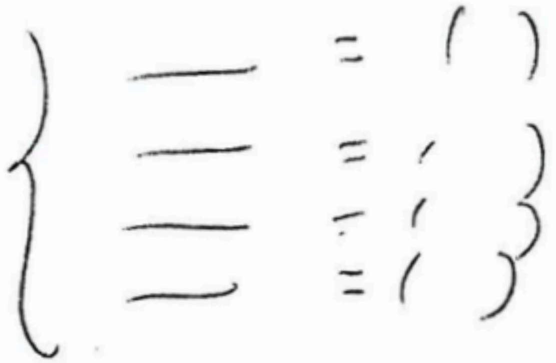
N-2 equations in N unknowns $y''_{j=1,\dots,N}$, i.e., each y”$_j$ is coupled only to its nearest neighbors at j+1 and j-1.

$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

→ $$\frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \boxed{\frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}}$$

Repeating this also for j=2, …, N-1:

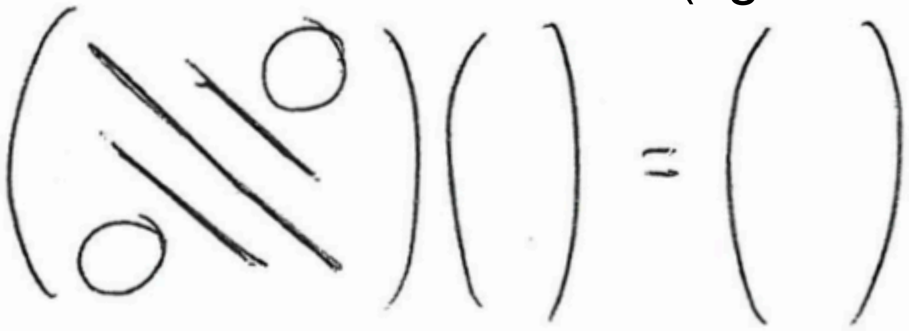


tridiagonal system (set of equations)

NxN tridiagonal matrix

N

N coefficients (right hand)



$y''_{j=1,\ldots,N}$

N-2 equations in N unknowns $y''_{j=1,\ldots,N}$, i.e., each y”$_j$ is coupled only to its nearest neighbors at j+1 and j-1.

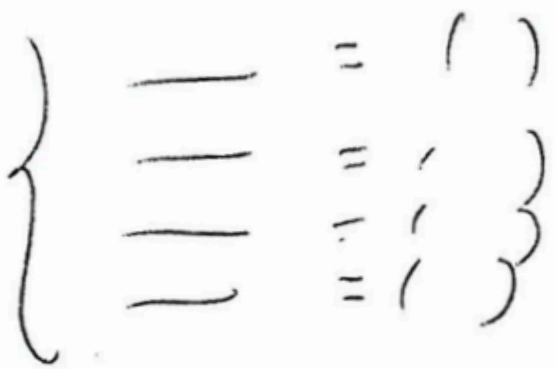
Imposing the continuity of the first derivative translates into a set of N-2 equations, which can be solved with linear algebra techniques to yield y”$_2$, …, y”$_{N-1}$

$$y'|_{x_j^-} = \frac{y_j - y_{j-1}}{x_j - x_{j-1}} + \frac{1}{6}(x_j - x_{j-1})y''_{j-1} + \frac{1}{3}(x_j - x_{j-1})y''_j$$

$$y'|_{x_j^+} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{1}{3}(x_{j+1} - x_j)y''_j + \frac{1}{6}(x_{j+1} - x_j)y''_{j+1}$$

$$\Rightarrow \frac{x_j - x_{j-1}}{6}y''_{j-1} + \frac{x_{j+1} - x_{j-1}}{3}y''_j + \frac{x_{j-1} - x_j}{6}y''_{j+1} = \boxed{\frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}}$$

Repeating this also for j=2, …, N-1:

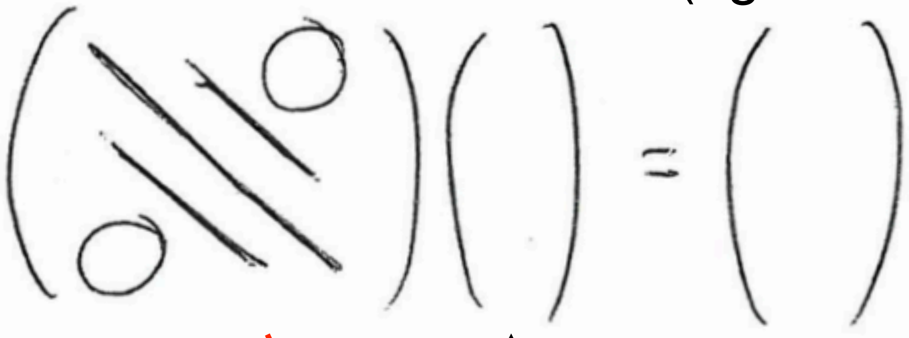


tridiagonal system (set of equations)

NxN tridiagonal matrix

N

N coefficients (right hand)



$y''_{j=1,\ldots,N}$

N-2 equations in N unknowns $y''_{j=1,\ldots,N}$, i.e., each y”$_j$ is coupled only to its nearest neighbors at j+1 and j-1.

Imposing the continuity of the first derivative translates into a set of N-2 equations, which can be solved with linear algebra techniques to yield y”$_2$, …, y”$_{N-1}$

Solve the NxN linear system of equations using Cramer’s rule

Q: What about y''_1 and y''_N, boundary conditions?

Q: What about y''_1 and y''_N, boundary conditions?

A: Two choices: 1) $y''_1 = y''_N = 0$, i.e., natural spline; 2) calculate them from one-sided differences, i.e., set y''_1 and y''_N to values calculated from equation 🟦 .

Q: What about y''_1 and y''_N, boundary conditions?

A: Two choices: 1) $y''_1 = y''_N = 0$, i.e., natural spline; 2) calculate them from one-sided differences, i.e., set y''_1 and y''_N to values calculated from equation 🟦 .

▲ —> there is a 2-parameter family of possible solutions; for a unique solution, you need to specify the boundary conditions y''_1 and y''_N.

Q: What about y''_1 and y''_N, boundary conditions?

A: Two choices: 1) $y''_1 = y''_N = 0$, i.e., natural spline; 2) calculate them from one-sided differences, i.e., set y''_1 and y''_N to values calculated from equation 🟦 .

▲ —> there is a 2-parameter family of possible solutions; for a unique solution, you need to specify the boundary conditions y''_1 and y''_N.

IN GENERAL (except for the assignment), just use the function/routine in the library for linear, quadratic, cubic, or cubic spline interpolations.