Numerical Methods in Engineering

2- INTERPOLATION AND CURVE FITTING

DR. ETEMADI

Interpolation

- Consider a table of values (x_i, f_i) , $i=1, 2, 3, \ldots$.

x_0	x_1	x_2	x_3	…	…	…	x_n
f_0	f_1	f_2	f_3	…	…	…	f_n

The process of estimating ***f*** for any intermediate value of ***x*** is called interpolation.

- $P_n(x) \cong f(x)$

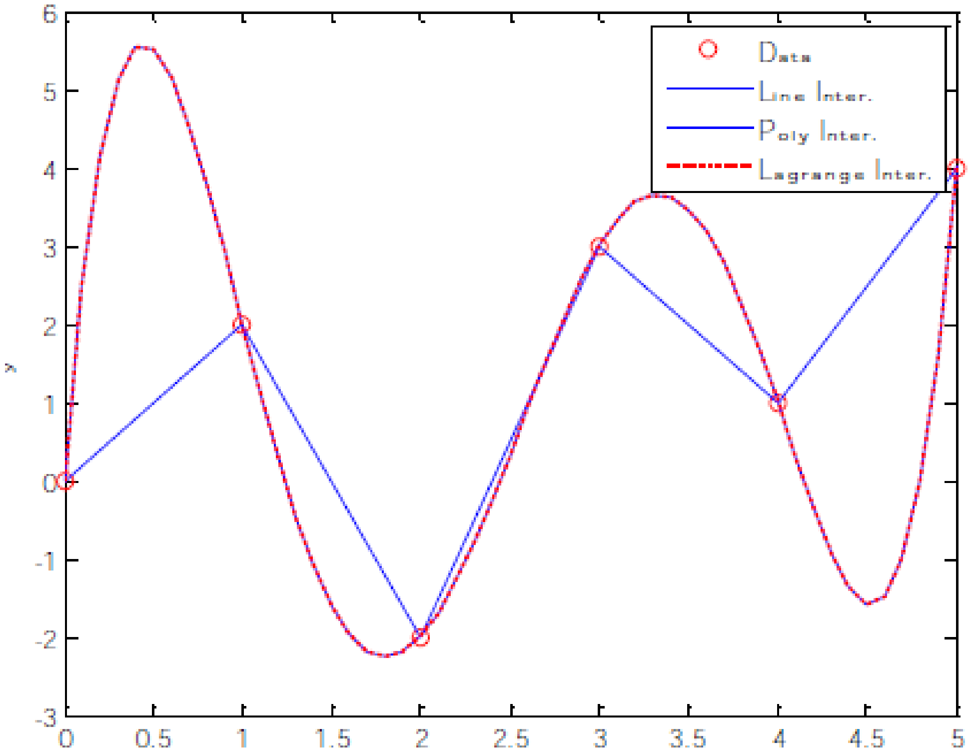
Polynomial ***P****n(x) is used as an estimation for the unknown function* ***f****(x)*

n *refers to the order of polynomial* ***P***

Interpolation

- ***Pn*** *descibes a curve that passes through the points of the table:*

x0	x1	x2	x3	…	…	…	xn
f0	f1	f2	f3	…	…	…	fn

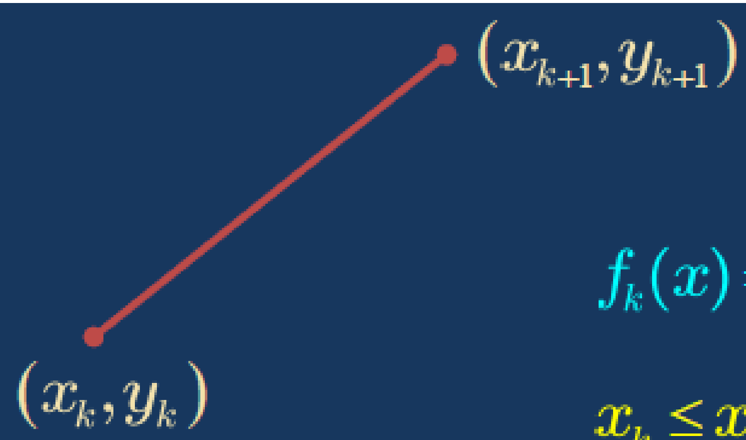


Interpolation Methods

- Newton method
  - Newton forward formula
  - Newton backward formula
  - Forward difference method
  - Backward difference method
- Lagrange's formula
- Spline interpolation
- …

Linear Interpolation

- Linear estimation (a line) between every adjacent points



$$f_k(x) = y_k + \frac{x - x_k}{x_{k+1} - x_k}(y_{k+1}, y_k)$$

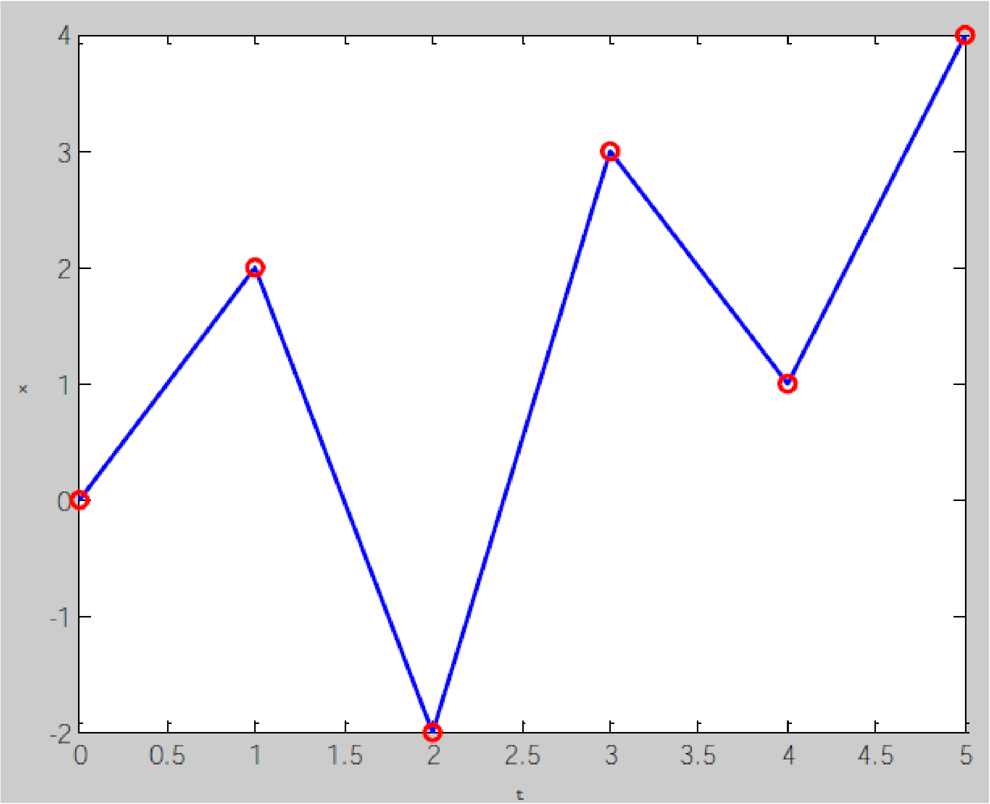
$$x_k \leq x \leq x_{k+1}$$

- The result is a function:
  - Continuous
  - Not differentiable

Linear Interpolation (Example)

- *Horizontal axis:* t
- *Vertical Axis:* x

$$t = [0, 1, 2, 3, 4, 5],\ x = [0, 2, -2, 3, 1, 4]$$



Polynomial Interpolation

General Form

- A set of ***n+1*** points $\rightarrow$ Order ***n*** Polynomial

$$\left\{(x_i, y_i)\right\}_{i=0}^{n}$$

$$f(x) = \sum_{k=0}^{n} a_k x^k \Rightarrow y_i = f(x_i) = \sum_{k=0}^{n} a_k x_i^k, \quad k = 0, \cdots, n$$

Polynomial Interpolation

General Form

$$\{(x_i, y_i)\}_{i=0}^{n}$$

$$f(x) = \sum_{k=0}^{n} a_k x^k \Rightarrow y_i = f(x_i) = \sum_{k=0}^{n} a_k x_i^k, \quad k = 0, \cdots, n$$

$$\left.\begin{array}{l} y_0 = a_0 + a_1 x_0^1 + \cdots + a_n x_0^n \\ y_1 = a_0 + a_1 x_1^1 + \cdots + a_n x_1^n \\ \vdots \\ y_n = a_0 + a_1 x_n^1 + \cdots + a_n x_n^n \end{array}\right\} \Rightarrow \begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

$$\therefore \mathbf{a} = \mathbf{x}^{-1} \mathbf{y}$$

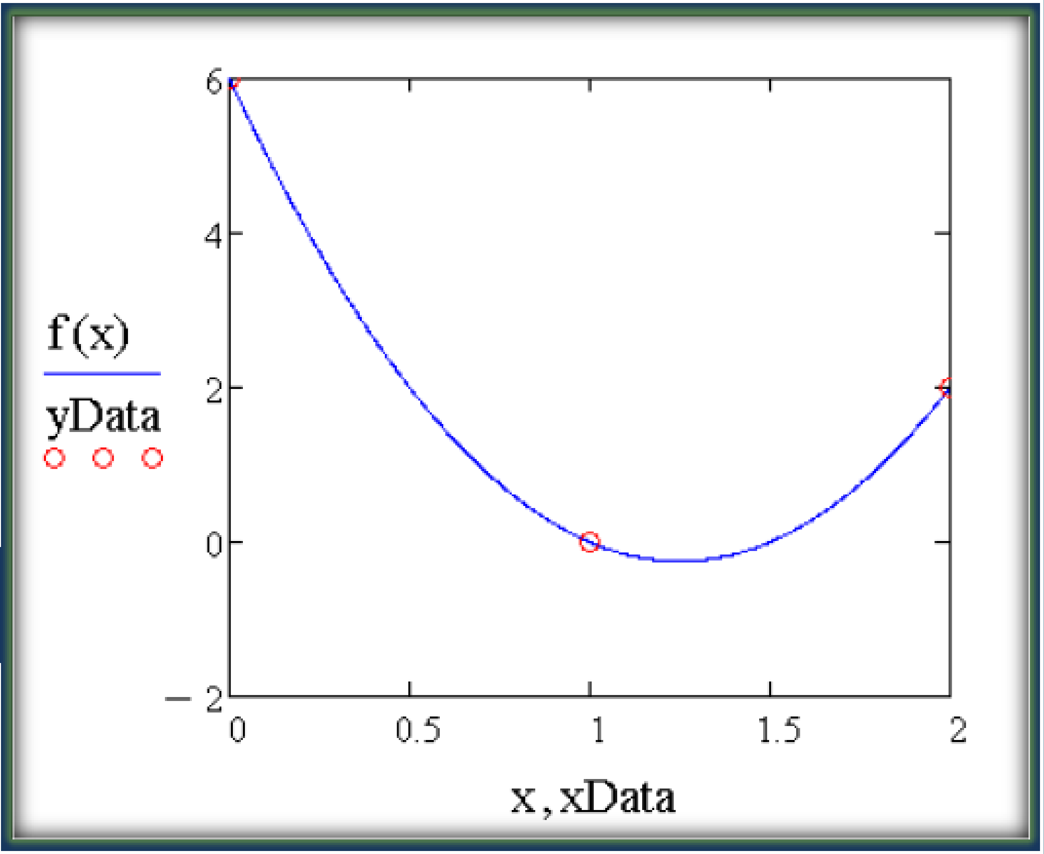
Polynomial Interpolation (Example 1)

General Form

- Determine interpolating polynomial: $\{(0,6),(1,0),(2,2)\}$

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 6 \\ 0 \\ 2 \end{bmatrix}$$

$$\Rightarrow \quad f(x) = 4x^2 - 10x + 6$$



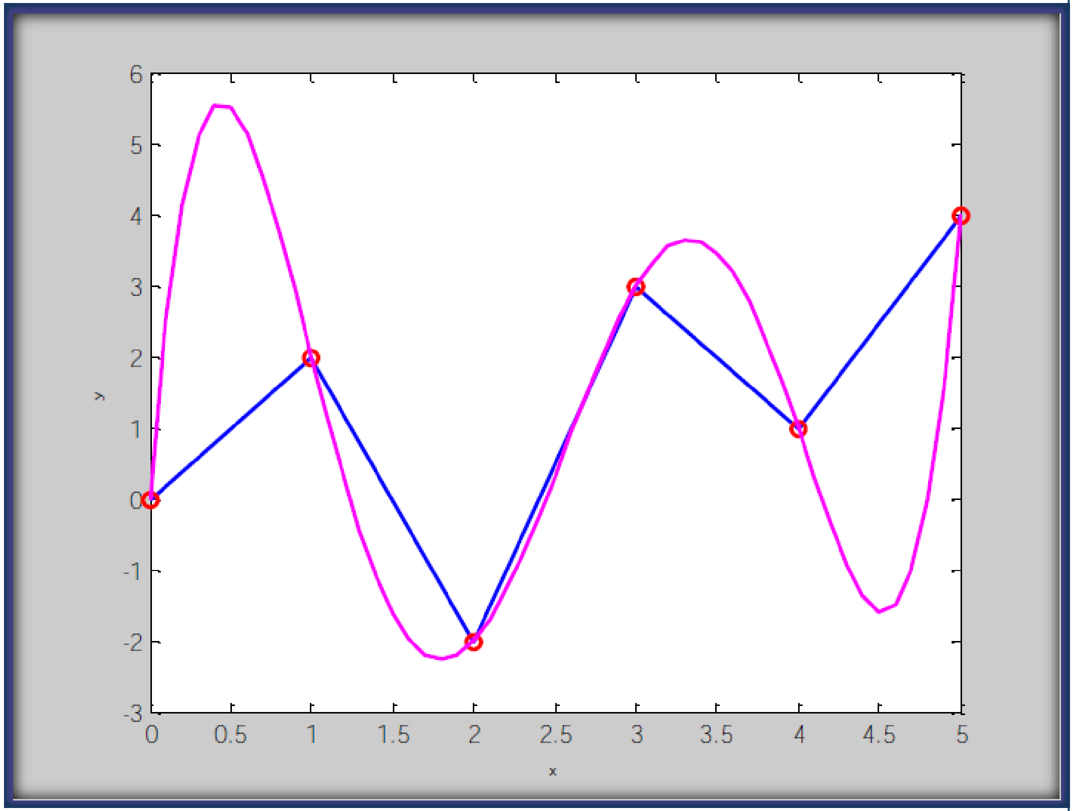
Polynomial Interpolation (Example 2)

General Form

1. Linear Interpolation
2. polynomial of order 5

$$x = [0, 1, 2, 3, 4, 5],$$

$$y = [0, 2, -2, 3, 1, 4]$$



$$f(x) = 0.4917x^5 - 6.2083x^4 + 27.4583x^3 - 49.2917x^2 + 29.55x$$

Polynomial Interpolation

- Disadvantages:
  - By increasing number of points, order of the polynomial grows
  - Solving the set of linear equations might be not very easy.

Lagrange's Interpolation Formula

- A unique polynomial that passes through all points.

$$\{(x_i, y_i)\}_{i=0}^{n}$$

$$\left.\begin{aligned} f(x) &= \sum_{k=0}^{n} L_k(x) y_k \\ L_k(x) &= \begin{cases} 1 & x = x_k \\ 0 & x \neq x_k \end{cases} \end{aligned}\right\} \Rightarrow f(x_k) = y_k$$

$$L_k(x) = \frac{(x-x_0)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)} = \prod_{\substack{i=0 \\ i\neq k}}^{n} \frac{x-x_i}{x_k-x_i},$$

$$k = 0, 1, \cdots, n$$

$$L_k(x_k) = 1, \quad L_k(x_i) = 0, \; x_i \neq x_k$$

Lagrange's Interpolation (Example)

- Data: $\{(0,6),(1,0),(2,2)\}$

$$L_0(x)=\frac{(x-1)(x-2)}{(0-1)(0-2)}=\frac{x^2-3x+2}{2}$$

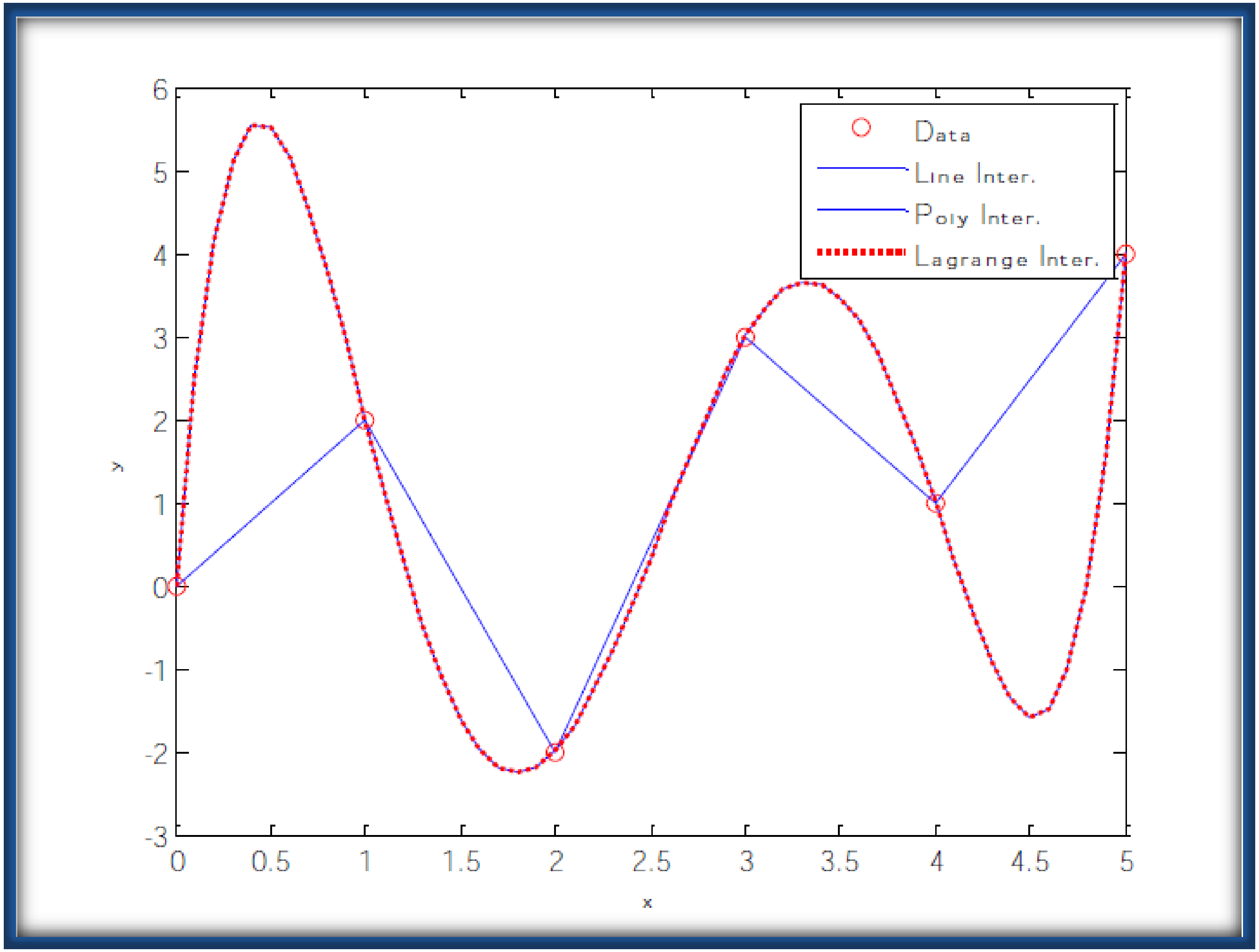
$$L_1(x)=\frac{(x-0)(x-2)}{(1-0)(1-2)}=\frac{x^2-2x}{-1}$$

$$L_2(x)=\frac{(x-0)(x-1)}{(2-0)(2-1)}=\frac{x^2-x}{2}$$

$$f(x)=\sum_{k=0}^{2}L_k(x)y_k=L_0(x)y_0+L_1(x)y_1+L_2(x)y_2$$

$$f(x)=\sum_{k=0}^{2}L_k(x)y_k=6L_0(x)+0L_1(x)+2L_2(x)=4x^2-10x+6$$

Lagrange's Interpolation (Example)



Newton's Interpolation Formula

$$\{(x_i, y_i)\}_{i=0}^{n}$$

$$n_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_n)$$

$$n_0(x_0) = a_0 = y_0$$

$$n_1(x_1) = a_0 + a_1(x_1 - x_0) = y_1 \rightarrow a_1 = \frac{y_1 - y_0}{x_1 - x_0} \equiv Df_0$$

$$n_2(x_2) = a_0 + a_1(x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) \rightarrow$$

$$a_2 = \frac{\dfrac{y_2 - y_1}{x_2 - x_1} - \dfrac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_0} = \frac{Df_1 - Df_0}{x_2 - x_0}$$

Newton's Interpolation Formula

$$n_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_n)$$

$$a_n = \frac{D^{n-1}f_1 - D^{n-1}f_0}{x_n - x_0} \equiv D^n f_0$$

x_k	y_k	Df_k	$D^2 f_k$	$D^3 f_k$
x_0	y_0	$Df_0 = \frac{y_1 - y_0}{x_1 - x_0}$	$D^2 f_0 = \frac{Df_1 - Df_0}{x_2 - x_0}$	$D^3 f_0 = \frac{D^2 f_1 - D^2 f_0}{x_3 - x_0}$
x_1	y_1	$Df_1 = \frac{y_2 - y_1}{x_2 - x_1}$	$D^2 f_1 = \frac{Df_2 - Df_1}{x_3 - x_1}$	—
x_2	y_2	$Df_2 = \frac{y_3 - y_2}{x_3 - x_2}$	—	
x_3	y_3	—		

	$f[x]$	$f[x, x]$	$f[x, x, x]$	$f[x, x, x, x]$	$f[x, x, x, x, x]$
x_0	f_0				
		f_0'			
x_0	f_0		$\frac{f_0''}{2}$		
		f_0'			
x_0	f_0		$\frac{f_{01} - f_0'}{h}$		
		f_{01}			
x_1	f_1		$\frac{f_1' - f_{01}}{h}$		
		f_1'			
x_1	f_1				

Newton's Interpolation Formula (Example)

x	0	1	-1	2	-2
y	-5	-3	-15	39	-9

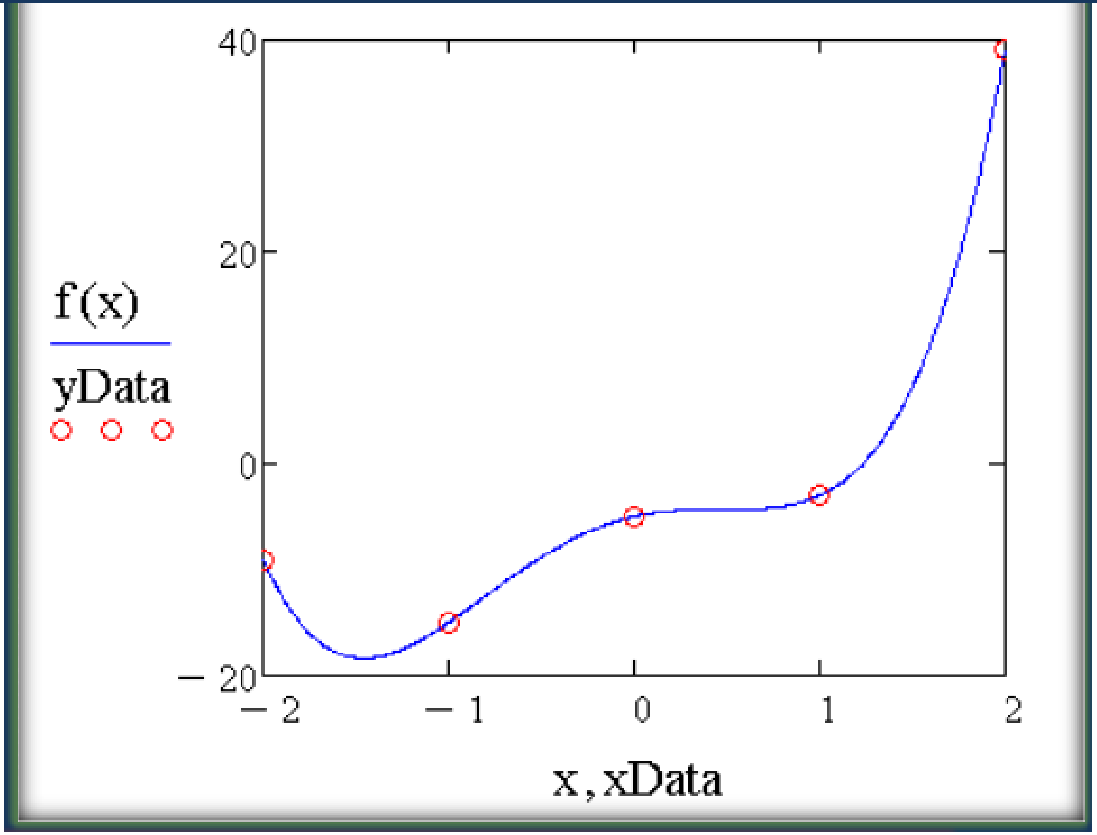
x_k	y_k	Df_k	$D^2 f_k$	$D^3 f_k$	$D^4 f_k$
0	-5	$Df_0 = \frac{-3-(-5)}{1-0} = 2$	$D^2 f_0 = \frac{6-2}{-1-0} = -4$	$D^3 f_0 = \frac{12-(-4)}{2-0} = 8$	$D^4 f_0 = \frac{2-8}{-2-0} = 3$
1	-3	$Df_1 = \frac{-15-(-3)}{-1-1} = 6$	$D^2 f_1 = \frac{18-6}{2-1} = 12$	$D^3 f_0 = \frac{6-12}{-2-1} = 2$	—
-1	-15	$Df_2 = \frac{39-(-15)}{2-(-1)} = 18$	$D^2 f_2 = \frac{12-18}{-2-(-1)} = 6$	—	
2	39	$Df_3 = \frac{-9-39}{-2-2} = 12$	—		
-2	-9	—			

$$n_4(x) = -5 + 2x - 4x(x-1) + 8x(x-1)(x+1) + 3x(x-1)(x+1)(x-2)$$

Newton's Interpolation Formula (Example)

x	0	1	–1	2	–2
y	–5	–3	–15	39	–9

$$n_4(x) = -5 + 2x - 4x(x-1) + 8x(x-1)(x+1) + 3x(x-1)(x+1)(x-2)$$



Newton or Lagrange

- Disadvantage of Lagrange's method of interpolation:

  To add new data, new set of lagrangian multipliers must be calculated

- Advantage of Newton method:

  By introducing new terms, new data can be added to the existing polynomial.

Newton's Interpolation Formula

2nd Method

x	0	1	−1	2	−2
y	−5	−3	−15	39	−9

$$n_0(x) = a_0 \rightarrow n_0(x) = -5$$

$$n_1(x) = n_0(x) + a_1(x - x_0) = -5 + a_1(x - 0) \xrightarrow{n_1(x_1)=y_1} n_1(x) = -5 + 2x$$

$$n_2(x) = n_1(x) + a_2(x - x_0)(x - x_1) \xrightarrow{n_2(x_2)=y_2} n_2(x) = -5 + 2x - 4x(x-1)$$

$$n_3(x) = n_2(x) + a_3(x - x_0)(x - x_1)(x - x_2) \xrightarrow{n_3(x_3)=y_3}$$

$$n_3(x) = -5 + 2x - 4x(x-1) + 8x(x-1)(x+1)$$

$$n_4(x) = n_3(x) + a_4(x - x_0)(x - x_1)(x - x_2)(x - x_3) \xrightarrow{n_4(x_4)=y_4}$$

$$n_4(x) = -5 + 2x - 4x(x-1) + 8x(x-1)(x+1) + 3x(x-1)(x+1)(x-2)$$

Equally spaced Interpolation

Newton's Forward Difference Method

- If the points of interpolation are equally spaced:

$$x_j = x_0 + jh; \qquad j = 0, \pm 1, \pm 2, \ldots .$$

- Assume forward difference operator as:

$$\Delta f(x) \equiv f(x + h) - f(x)$$

$$\Delta^{n+1} f(x) = \Delta^n[\Delta f(x)] = \Delta^n f(x + h) - \Delta^n f(x); \qquad n = 1, 2, \ldots .$$

x	$f(x)$	Δ	Δ^2	Δ^3	$\cdots$
x_0	$f(x_0)$				
		$f(x_1) - f(x_0) = \Delta f(x_0)$			
x_1	$f(x_1)$		$\Delta f(x_1) - \Delta f(x_0) = \Delta^2 f(x_0)$		
		$f(x_2) - f(x_1) = \Delta f(x_1)$		$\Delta^2 f(x_1) - \Delta^2 f(x_0) = \Delta^3 f(x_0)$	
x_2	$f(x_2)$		$\Delta f(x_2) - \Delta f(x_1) = \Delta^2 f(x_1)$		$\cdots$
		$f(x_3) - f(x_2) = \Delta f(x_2)$		$\Delta^2 f(x_2) - \Delta^2 f(x_1) = \Delta^3 f(x_1)$	
x_3	$f(x_3)$		$\Delta f(x_3) - \Delta f(x_2) = \Delta^2 f(x_2)$		
		$f(x_4) - f(x_3) = \Delta f(x_3)$		$\vdots$	
x_4	$f(x_4)$		$\vdots$		
$\vdots$	$\vdots$				

HomeWork

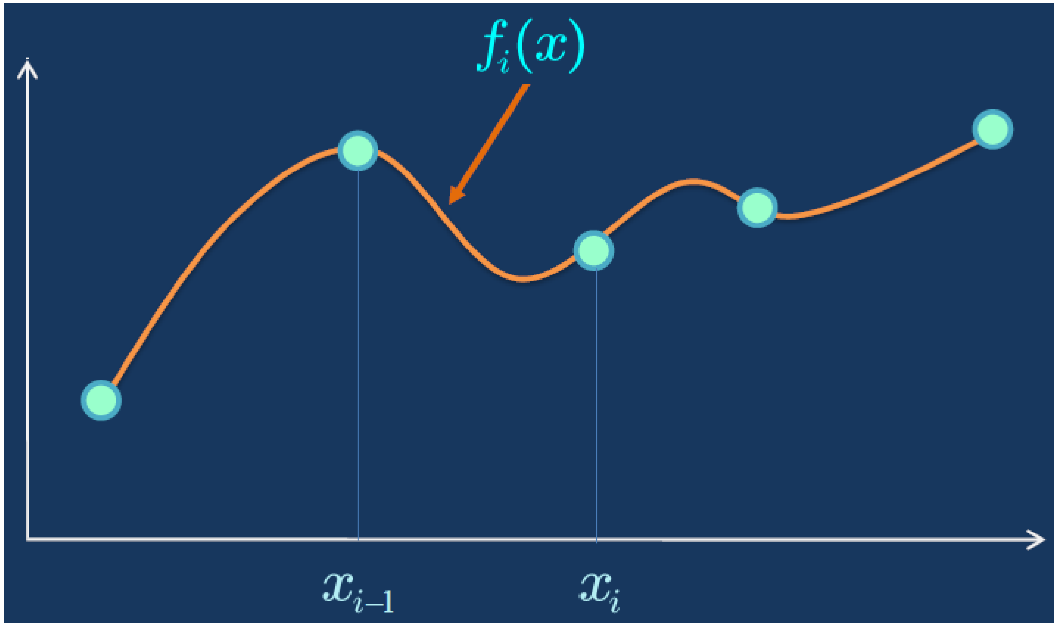
- Do some research and find the Newton's forward difference formula for equidistance data.

- Then solve the problem of finding the interpolation polynomial by Newton's forward difference formula for the following data:

x	0	1	–1	2	–2
y	–5	–3	–15	39	–9

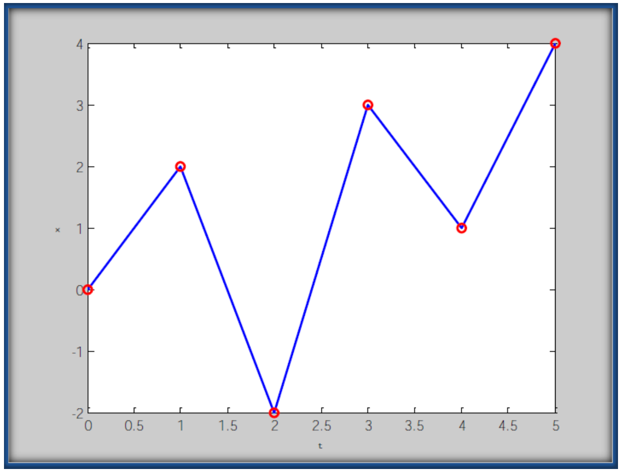
Spline Interpolation

- Fit a single polynomial between every pair of adjacent points.
- Composition of these polynomials are called Spline
- They all form a continuous composite curve.

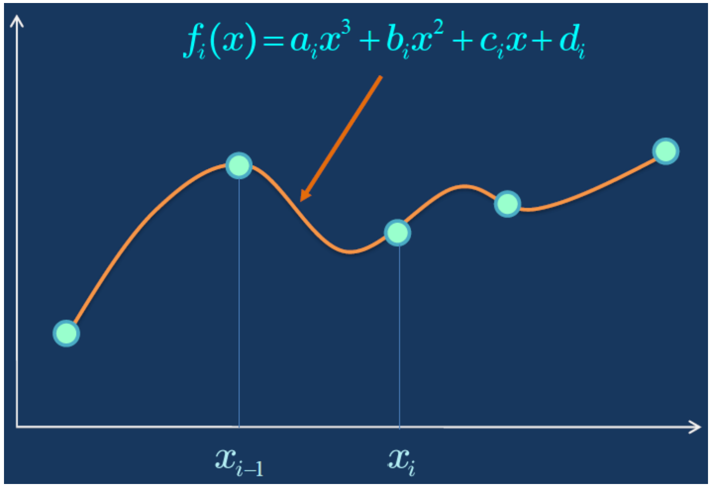


Spline Interpolation

- Linear spline: 1st-order lines between adjacent points
  - Continuous but not differentiable



- Cubic spline:
  3rd-order curves in each subinterval
  - f is Continuous at xi
  - f' is Continuous at xi
  - f'' is Continuous at xi



Why Spline?

- Too many data point → high order interpolating polynomial
- Cubic spline : just 3rd order polynomials

- Spline is computationally more economic for heavy problems

Spline Interpolation

- The spline is defined by a set of equations each defining a 3rd-order curve for a subinterval

$$\begin{cases} f_1(x) & x_0 \le x \le x_1 \\ f_2(x) & x_1 \le x \le x_2 \\ \vdots & \vdots \\ f_i(x) & x_{i-1} \le x \le x_i \\ \vdots & \vdots \\ f_n(x) & x_{n-1} \le x \le x_n \end{cases}$$

Continuouity
f1 (x1)=f2 (x1)
f2 (x2)=f3 (x2)
f3 (x3)=f4 (x3)
….

f ‘ Continuouity
f ‘1 (x1)= f ‘ 2 (x1)
f ‘2 (x2)= f ‘3 (x2)
f ‘3 (x3)= f ‘4 (x3)
….

f “ Continuouity
f “1 (x1)= f ”2 (x1)
f “2 (x2)= f “3 (x2)
…

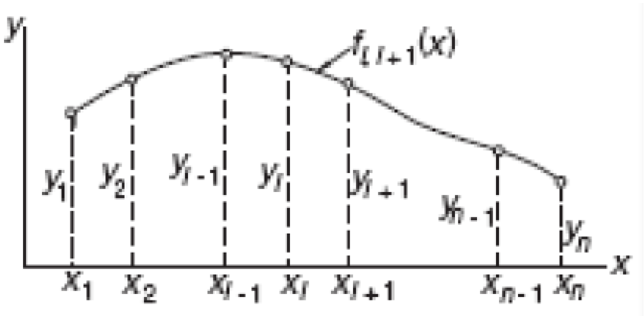
Spline Curve

- Number of data ***n*** → Number of spline curves ***n-1***
- ***f "*** is continuous at each point: $f''_{i-1,i}(x_i) = f''_{i,i+1}(x_i) = k_i$
- Cubic spline → curvature is linear

$$f''_{i,i+1}(x) = k_i l_i(x) + k_{i+1} l_{i+1}(x)$$

- From Lagrange's method

$$l_i(x) = \frac{x - x_{i+1}}{x_i - x_{i+1}} \qquad l_{i+1}(x) = \frac{x - x_i}{x_{i+1} - x_i}$$



$$f''_{i,i+1}(x) = \frac{k_i(x - x_{i+1}) - k_{i+1}(x - x_i)}{x_i - x_{i+1}}$$

Spline Curve

- Integration of curvature:

$$f''_{i,i+1}(x) = \frac{k_i(x - x_{i+1}) - k_{i+1}(x - x_i)}{x_i - x_{i+1}}$$

$$f_{i,i+1}(x) = \frac{k_i(x - x_{i+1})^3 - k_{i+1}(x - x_i)^3}{6(x_i - x_{i+1})} + A(x - x_{i+1}) - B(x - x_i)$$

- Using boundary values leads to:

$$f_{i,i+1}(x) = \frac{k_i}{6}\left[\frac{(x - x_{i+1})^3}{x_i - x_{i+1}} - (x - x_{i+1})(x_i - x_{i+1})\right] - \frac{k_{i+1}}{6}\left[\frac{(x - x_i)^3}{x_i - x_{i+1}} - (x - x_i)(x_i - x_{i+1})\right] + \frac{y_i(x - x_{i+1}) - y_{i+1}(x - x_i)}{x_i - x_{i+1}}$$

Spline Curve 1

- Calculating intermediate curvatures ***k**i, i=2,3,4, … , n-1.*

  Use slope continuity at each point:

$$f'_{i-1,i}(x_i) = f'_{i,i+1}(x_i)$$

$$k_{i-1}(x_{i-1} - x_i) + 2k_i(x_{i-1} - x_{i+1}) + k_{i+1}(x_i - x_{i+1}) = 6\left(\frac{y_{i-1} - y_i}{x_{i-1} - x_i} - \frac{y_i - y_{i+1}}{x_i - x_{i+1}}\right)$$

Note that ***k**0=**k**n=0*

- *Initial and final curvatures of the composite curve is set to zero*

Spline Curve 1

- Special case:

  For equidistant points: *h=xi+1 - xi*

$$\begin{cases} k_{i-1} + 4k_i + k_{i+1} = \dfrac{6}{h^2}(y_{i-1} - 2y_i + y_{i+1}), & i = 2,3,\cdots,n-1 \\ k_i = 0 & i = 0,n \end{cases}$$

Spline Interpolation (Example)

i	x	$f(x)$	$f''(x)$
1	–0.50	0.731	0.0
2	0.00	1.000	
3	0.25	1.268	
4	1.00	1.718	0.0

$$f_1'' = f_4'' = 0$$

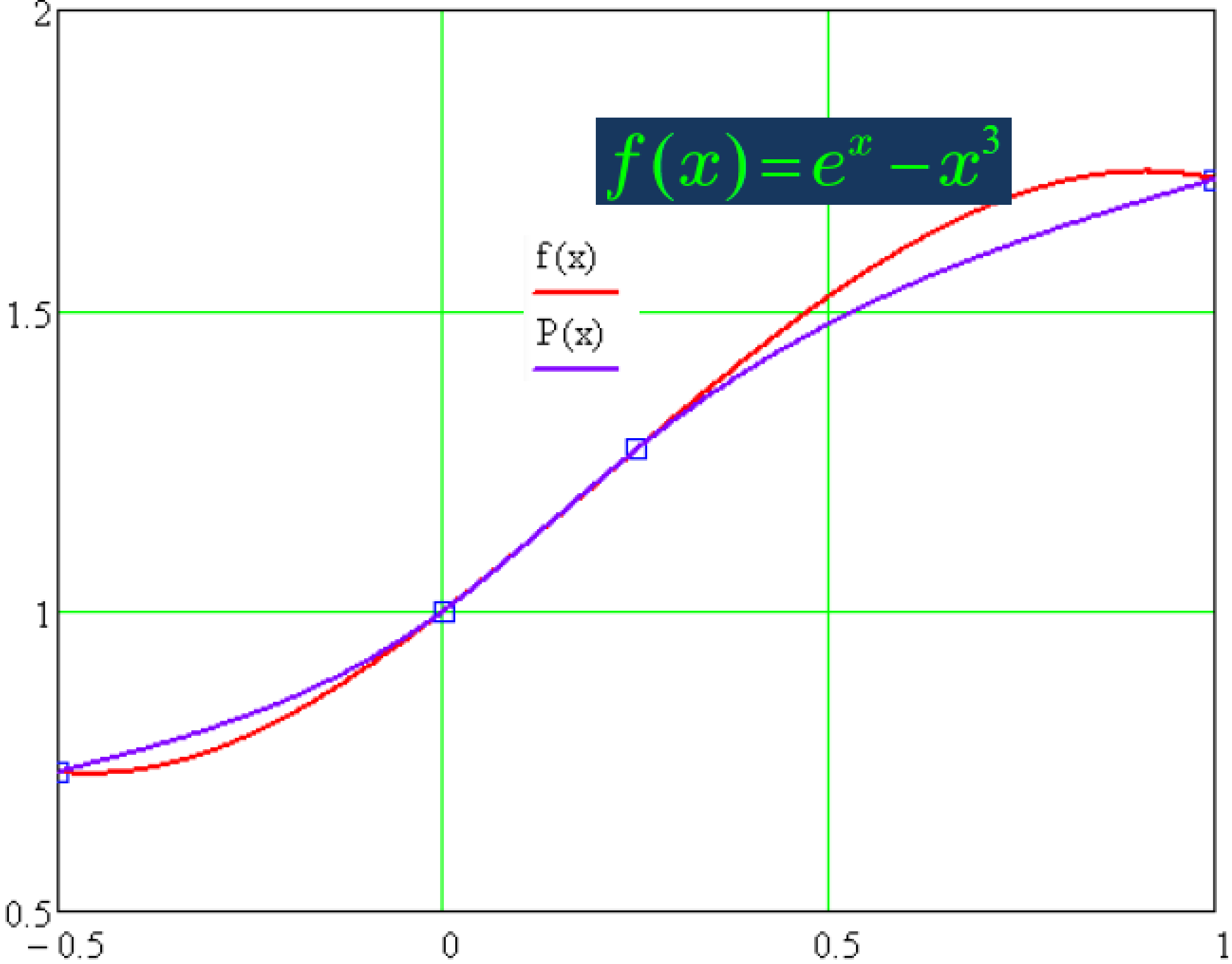
$$\begin{cases} i=2: \ 1.50 f_2'' + 0.25 f_3'' = 3.21997 \\ i=3: \ 0.25 f_2'' + 2.0 f_3'' = -2.8425 \end{cases} \rightarrow \begin{cases} f_2'' = 2.4342 \\ f_3'' = -1.7255 \end{cases}$$

$$f_1(x) = 0.81141x^3 + 1.21712x^2 + 0.94264x + 1$$

$$f_2(x) = -2.77319x^3 + 1.21712x^2 + 0.94164x + 1$$

$$f_3(x) = 0.38346x^3 - 1.15037x^2 + 1.53452x + 0.95068$$

Spline Interpolation (Example)



Spline Interpolation 2

- Instead of initial and final curvature, initial and final slopes, ***S0*** and ***S1*** are controlled
- For equidistant points: $x_{i+1} - x_i = h\ ,\ i = n-1,\ldots,0$

$$x_i \le x \le x_{i+1}$$

$$f_{i,i+1}(x) = \left[\frac{(x-x_{i+1})^2}{h^2} + 2\frac{(x-x_i)(x-x_{i+1})}{h^3}\right] y_i + \left[\frac{(x-x_i)^2}{h^2} - 2\frac{(x-x_i)^2(x-x_{i+1})}{h^3}\right] y_{i+1} + \frac{(x-x_i)(x-x_{i+1})^2}{h^2} S_i + \frac{(x-x_i)^2(x-x_{i+1})}{h^3} S_{i+1}$$

$$S_i + 4S_{i+1} + S_{i+2} = \frac{3}{h}(f_{i+2} - f_i), \qquad i = 0, 1, \cdots, n-1$$

Spline Interpolation 2 (Example)

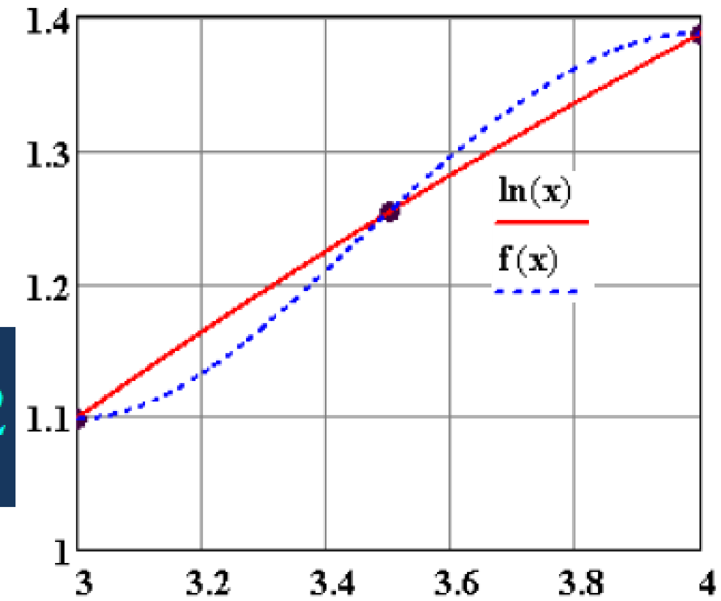
$$x=[3,3.5,4]; \quad y=[1.098,1.253,1.386]; n=3$$

$$S_0=S_2=0$$

- Solution:

$$S_i+4S_{i+1}+S_{i+2}=\frac{3}{h}(f_{i+2}-f_i), \quad i=0,1,\cdots,n-1$$

$$4S_1=\frac{3}{0.5}(1.386-1.098)\Rightarrow S_1=0.432$$



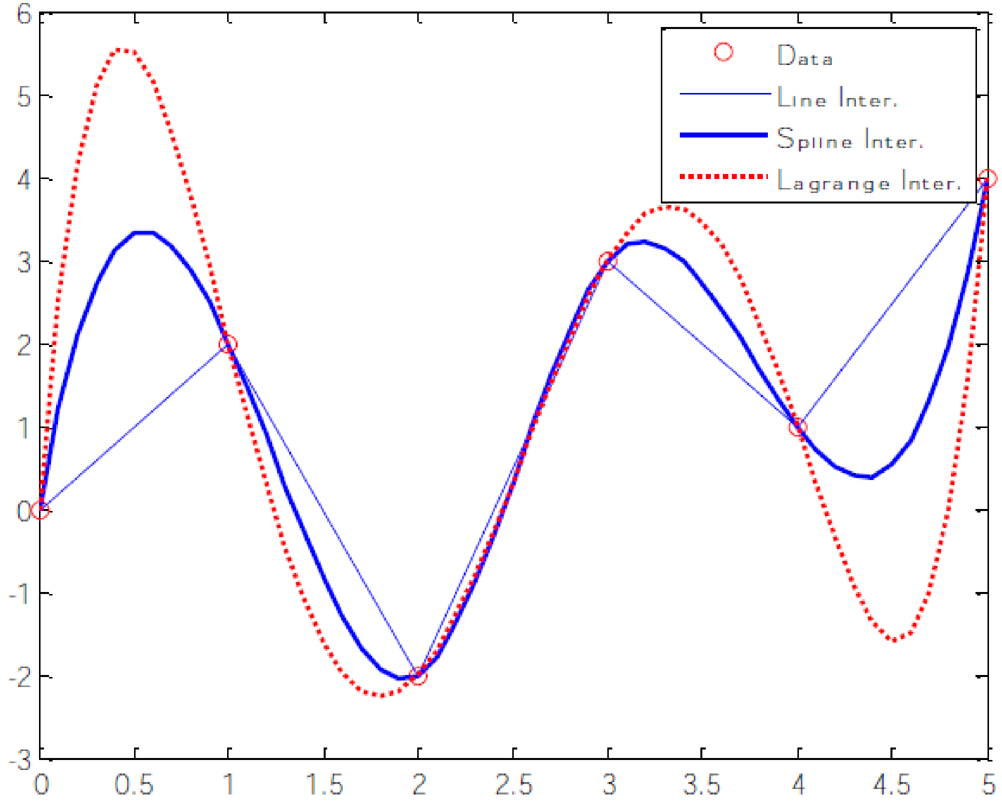
$$\begin{cases} f_{1,2}(x)=-0.752x^3+7.764x^2-26.28x+30.366 & 3\le x\le 3.5 \\ f_{2,3}(x)=-0.416x^3+4.248x^2-14.016x+16.107 & 3.5\le x\le 4 \end{cases}$$

$$f(3.541)=? \quad \rightarrow \quad f(3.541)=1.232$$

Spline Interpolation (Example)

- Compare Lagrange's interpolation curve with spline

$$x=[0,1,2,3,4,5], \qquad y=[0,2,-2,3,1,4]$$



Homework

- Use a computer program to determine cubic curves of a spline for tabulated points

x	0	0.2	0.3	0.4	0.5	0.6	0.7	0.8	1.0
y	0.302	0.185	0.106	0.093	0.240	0.579	0.561	0.468	0.302

- Solve the problem for these conditions:

1. $k_0 = k_n = 0$
2. $S_0 = 0$ and $S_n = \pi/2$

Deliver the solution on paper with details and attach a copy of computer program.

2D Interpolation (Lagrange method)

- Function $z=f(x, y)$ is evaluated in several points

$$z_i=f(x_i, y_i)$$

- Goal is to calculate a function to estimate z for any arbitrary value of x and y.

- Lagrange method

2D Lagrangian Interpolation

$$\left.\begin{array}{l} f(x,y)=\displaystyle\sum_{i=1}^{M}\sum_{j=1}^{N}L_{ij}(x,y)z_{ij} \\ L_{ij}(x,y)=\begin{cases}1, & x=x_i \;\; and \;\; y=y_i \\ 0, & otherwise.\end{cases}\end{array}\right\} \Rightarrow f(x_i,y_j)=z_{ij}$$

$$L_{ij}(x,y)=L_i(x)L_j(y)$$

$$L_k(x_k)=1, \quad L_k(x_i)=0, x_i \neq x_k$$

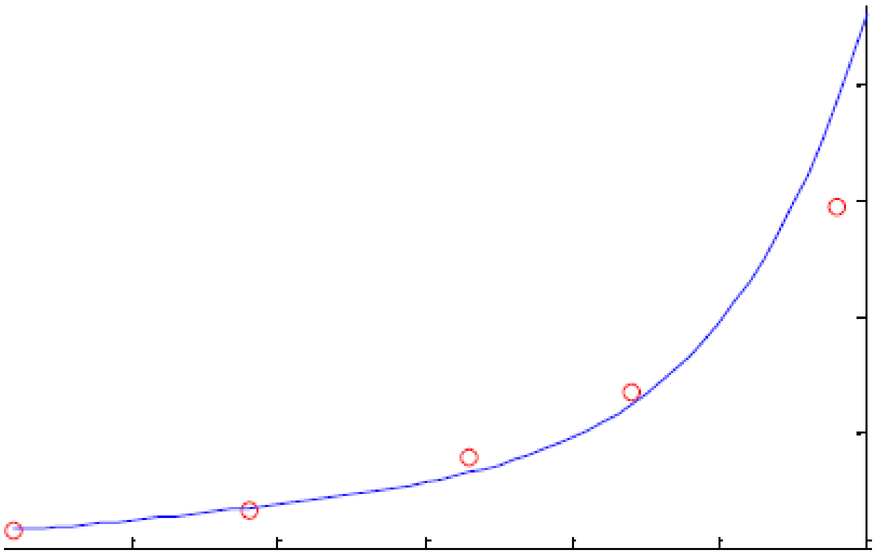
$$L_k(x)=\frac{(x-x_0)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)}$$

$$=\prod_{\substack{i=0\\ i\neq k}}^{n}\frac{x-x_i}{x_k-x_i}, \qquad k=0,1,\cdots,n$$

Curve Fitting

- A smooth curve passing not necessarily through the given points but with minimum distance from them.

- Polynomial interpolation passes through all points and yields an oscillatory curve.



Least Squares Method

- Given data $(x_i, y_i), i = 1, 2, \cdots, n$

- Approximating Function $f(x) = f(x; a_1, a_2, \cdots, a_m)$

  m<n

- Difference between every point and its approximated velue:

  $r_i = y_i - f(x_i) \equiv residual$

- Sum of squares of residuals: $S(a_1, a_2, \cdots, a_m) = \sum_{i=1}^{n} [y_i - f(x_i)]^2$

- To have the best approximation,

  S must be minimized wrt ***a****k*: $\frac{\partial S}{\partial a_k} = 0, k = 1, 2, \cdots, m$

Least Squares Method (Line Fitting)

- For a table of data (xi, yi), $i=1,2,3, \ldots, n$
  the approximating function is a line: $f(x)=a+bx$

- To solve the problem parameters a and b must be evaluated:

1. Construct the error function:

$$S(a,b)=\sum_{i=1}^{n}[y_i-a-bx_i]^2$$

2. Minimize $\boldsymbol{S}$ wrt a and b:

$$\frac{\partial S}{\partial a}=\sum_{i=1}^{n}-2(y_i-a-bx_i)=2\left(-\sum_{i=1}^{n}y_i+na+b\sum_{i=1}^{n}x_i\right)=0$$

$$\frac{\partial S}{\partial b}=\sum_{i=1}^{n}-2(y_i-a-bx_i)x_i=2\left(-\sum_{i=1}^{n}x_iy_i+a\sum_{i=1}^{n}x_i+b\sum_{i=1}^{n}x_i^2\right)=0$$

Least Squares Method (Line Fitting)

- *(xi, yi), i=1,2,3, … , n → f(x)=a+bx*

$$\begin{cases} a = \dfrac{\bar{y}\sum x_i^2 - \bar{x}\sum x_i y_i}{\sum x_i^2 - n\bar{x}^2} \\ b = \dfrac{\sum x_i y_i - n\overline{xy}}{\sum x_i^2 - n\bar{x}^2} \end{cases}$$

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

- *Alternative formula*

$$b = \frac{\sum y_i (x_i - \bar{x})}{\sum x_i (x_i - \bar{x})} \qquad a = \bar{y} - \bar{x}b$$

Least Squares Method (Line Fitting)

- Example

x	0.0	1.0	2.0	2.5	3.0
y	2.9	3.7	4.1	4.4	5.0

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{0.0+1.0+2.0+2.5+3.0}{5} = 1.7$$

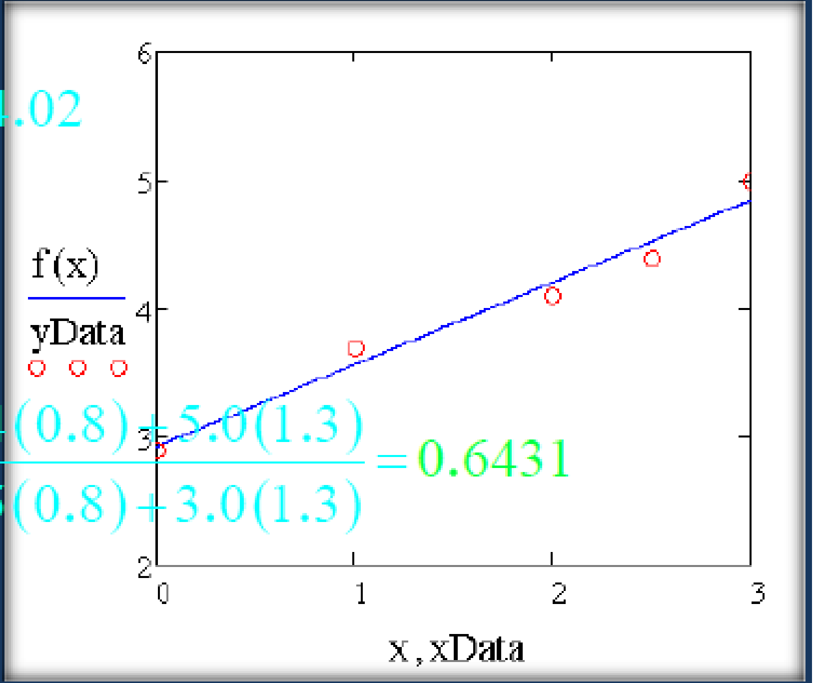
$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i = \frac{2.9+3.7+4.1+4.4+5.0}{5} = 4.02$$

$$b = \frac{\sum y_i (x_i - \bar{x})}{\sum x_i (x_i - \bar{x})}$$

$$= \frac{2.9(-1.7)+3.7(-0.7)+4.1(0.3)+4.4(0.8)+5.0(1.3)}{0.0(-1.7)+1.0(-0.7)+2.0(0.3)+2.5(0.8)+3.0(1.3)} = 0.6431$$

$$a = \bar{y} - \bar{x}b = 4.02 - 1.7(0.6431) = 2.927$$

$$f(x) = 2.927 + 0.6431x$$



Least Squares Method (Polynomial Fitting)

- Fitting a polynomial of order *m-1*:

$$f(x) = \sum_{j=1}^{m} a_j x^{j-1}$$

$$A = \begin{bmatrix} n & \sum x_i & \sum x_i^2 & \cdots & \sum x_i^m \\ \sum x_i & \sum x_i^2 & \sum x_i^3 & \cdots & \sum x_i^{m+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum x_i^{m-1} & \sum x_i^m & \sum x_i^{m+1} & \cdots & \sum x_i^{2m-2} \end{bmatrix} \qquad b = \begin{bmatrix} \sum y_i \\ \sum x_i y_i \\ \vdots \\ \sum x_i^{m-1} y_i \end{bmatrix}$$

- [*a1, a2, a3, … , am*]=A^(-1)*b

Least Squares Method (Polynomial Fitting)

- Note that:
- Increasing the order does not necessarily lead to more accuracy.
- Use standard deviation to find the best curve,

  n: number of data

  m: curve's polynomial order

$$\sigma^2 = \frac{\sum e_i^2}{n-m-1}$$

Homework

- Use a computer program and find the best curve to be fitted on the tabulated data(Check $m=2,3, \ldots, 10$)

xData	1.2	2.8	4.3	5.4	6.8	7.0
yData	7.5	16.1	38.9	67.0	146.6	266.2