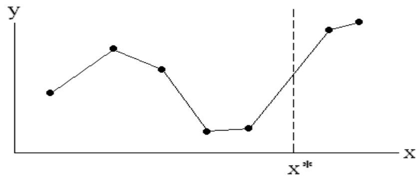
3 Interpolation

Given: $\{(x_k, y_k), k = 0,1,2,\cdots,N\}$

Find: $y(x^*)$ for some $x^* \in [Min(x_k), Max(x_k)]$



• simpliest method: piecewise linear interpolation

§ Polynomial Approximation

Theorem (Weierstrass Approximation Theorem --- existence)

If $f \in C[a,b]$ and given ε>0, then there exists a polynomial $P(x)$ defined on $[a,b]$ with the property that $|f(x) - P(x)| < \varepsilon$ for all $x \in [a,b]$.

Given: $\{(x_k, y_k), k = 0,1,2,\cdots,N\}$

Solution: find the unique polynomial of degree N, $P_N(x)$, passing through all these $(N+1)$ points.

$$P_N(x) = a_N x^N + a_{N-1}x^{N-1} + \cdots + a_1 x + a_0$$

Definition: Lagrange interpolating polynomial

$$L_{N,k}(x) \equiv \prod_{\substack{i=0 \\ i\neq k}}^{N} \frac{(x - x_i)}{(x_k - x_i)}$$

$$= \frac{(x-x_0)(x-x_1)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_N)}{(x_k-x_0)(x_k-x_1)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_N)}$$

- polynomial of degree N
- $L_{N,k}(x_j) = \delta_{jk} = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases}$ $\qquad P_N(x) = \sum_{k=0}^{N} y_k * L_{N,k}(x)$

Consequently

$$P_N(x_j) = \sum_{k=0}^{N} y_k * L_{N,k}(x_j) = \sum_{k=0}^{N} y_k * \delta_{jk} = y_j \text{ , for all } j$$

Theorem If $x_0, x_1, \cdots, x_N$ are distrinct numbers in $[a,b]$ and $f \in C^{N+1}[a,b]$, then for each $x \in [a,b]$, $\exists \xi \in [a,b]$ such that

$$f(x) = P_N(x) + \frac{f^{(N+1)}(\xi)}{(N+1)!}(x-x_0)(x-x_1)\cdots(x-x_N)$$

§ piecewise linear interpolation

~ use $P_1(x)$ between two adjacent points, i.e. for $[x_i, x_{i+1}]$

$$\text{Lagrange error} = \left| \frac{f''(\xi)}{2!}(x-x_i)(x-x_{i+1}) \right|$$

$$\leq \frac{\max(f'')}{2} \underset{x_i \leq x \leq x_{i+1}}{Max} |(x-x_i)(x-x_{i+1})|$$

$$= \frac{\max(f'')}{2} \cdot \frac{(x_{i+1}-x_i)^2}{4} \qquad \sim O(\Delta x^2)$$

§ N^{th} Polynomial approximation

$$\text{Lagrange error} = \left| \frac{f^{(N+1)}(\xi)}{(N+1)!} \prod_{i=0}^{N} (x - x_i) \right|$$

depending on the choices of nodes

Consider $[a,b] = [-1,1]$

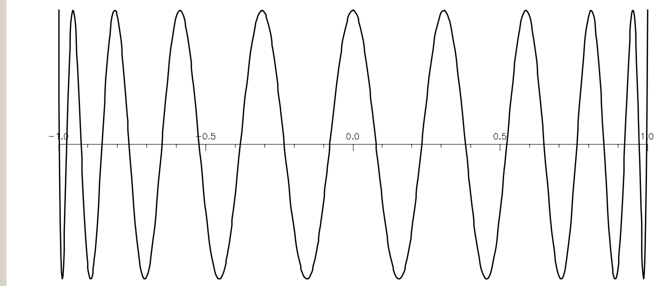
(i) uniformly spaced nodes: $x_{k+1} - x_k = \text{constant}$; $x_k = \frac{2k}{N} - 1,\ k = 0,1,\cdots,N$

(ii) Chebychev nodes: $x_k =$ roots of the $N+1^{th}$ Chebychev polynomial

$$T_{N+1}(x) \equiv \cos\left((N+1)\cos^{-1} x\right)$$

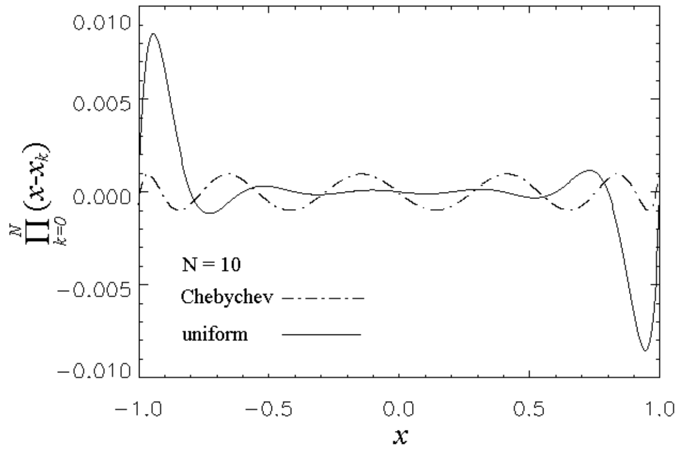
$$(N+1)\cos^{-1} x_k = \frac{\pi}{2} + k\pi \ ,\ \ k = 0,1,\cdots,N$$

$T_{20}(x)$



The Chebychev nodes minimize the maximum absolute value of

$$\left| \prod_{i=0}^{N} (x - x_i) \right| \quad \Rightarrow \quad \underset{\text{all sets of nodes}}{\text{Min}} \left\{ \underset{a \le x \le b}{\text{Max}} \left| \prod_{i=0}^{N} (x - x_i) \right| \right\} = 2^{-N}$$



§ Nevill's method --- computing higher-order polynomial based on lower-order polynomials

Define $P_{m_1, m_2, \cdots, m_k}(x) \equiv$ the polynomial of degree $k-1$ passing through the points $(x_{m_1}, y_{m_1}), (x_{m_2}, y_{m_2}), \cdots, (x_{m_k}, y_{m_k})$

e.g. $\{m_k\}_{k=1}^{3} = \{1,2,4\}$

$P_{1,2,4}(x) =$ the polynomial of degree 2 passing through the points $(x_1, y_1), (x_2, y_2), (x_4, y_4)$

- Construct the N^{th} order polynomial using the $(N-1)^{\text{th}}$ polynomials

$$P_{0,1,\cdots,N}(x)=\frac{(x-x_j)P_{0,1,\cdots,j-1,j+1,\cdots N}(x)-(x-x_i)P_{0,1,\cdots,i-1,i+1,\cdots N}(x)}{(x_i-x_j)}$$

$$\begin{array}{lllll}
P_0(x)=y_0 & & & & \\
P_1(x)=y_1 & P_{0,1}(x) & & & \\
P_2(x)=y_2 & P_{1,2}(x) & P_{0,1,2}(x) & & \\
P_3(x)=y_3 & P_{2,3}(x) & P_{1,2,3}(x) & P_{0,1,2,3}(x) & \\
P_4(x)=y_4 & P_{3,4}(x) & P_{2,3,4}(x) & P_{1,2,3,4}(x) & P_{0,1,2,3,4}(x)
\end{array}$$

$$P_{0,1,\cdots,N}(x)=\frac{(x-x_j)P_{0,1,\cdots,j-1,j+1,\cdots N}(x)-(x-x_i)P_{0,1,\cdots,i-1,i+1,\cdots N}(x)}{(x_i-x_j)}$$

Because $P_{0,1,\cdots j-1,j+1,N}(x_k)=y_k$ for $k\neq j$

$P_{0,1,\cdots i-1,i+1,N}(x_k)=y_k$ for $k\neq i$

- $P_{0,1,\cdots,N}(x_k)=\dfrac{(x_k-x_j)y_k-(x_k-x_i)y_k}{(x_i-x_j)}=y_k$ for $k\neq i, j$
- $P_{0,1,\cdots,N}(x_i)=\dfrac{(x_i-x_j)y_i-(x_i-x_i)P_{0,1,\cdots,i-1,i+1,N}(x_i)}{(x_i-x_j)}=y_i$
- $P_{0,1,\cdots,N}(x_j)=\dfrac{(x_j-x_j)P_{0,1,\cdots,j-1,j+1,N}(x_j)-(x_j-x_i)y_j}{(x_i-x_j)}=y_j$

§ Newton form of interpolation polynomial

Given: x^*

Find: $P_N(x^*)$

Write: $P_N(x)=\sum_{k=0}^{N}\left\{a_k\prod_{i=0}^{k-1}(x-x_i)\right\}$

$$=a_0+a_1(x-x_0)+a_2(x-x_0)(x-x_1)+a_3(x-x_0)(x-x_1)(x-x_2)+\cdots$$
$$+a_N(x-x_0)(x-x_1)\cdots(x-x_{N-1})$$

$$P_N(x)=a_0+a_1(x-x_0)+a_2(x-x_0)(x-x_1)+a_3(x-x_0)(x-x_1)(x-x_2)+\cdots$$
$$+a_N(x-x_0)(x-x_1)\cdots(x-x_{N-1})$$

$$=a_0+(x-x_0)\left\{\begin{array}{l}a_1+a_2(x-x_1)+a_3(x-x_1)(x-x_2)+\cdots\\+a_N(x-x_1)\cdots(x-x_{N-1})\end{array}\right\}$$

$$=a_0+(x-x_0)\{a_1+(x-x_1)[a_2+a_3(x-x_2)+\cdots+a_N(x-x_2)\cdots(x-x_{N-1})]\}$$

$$P_N(x)=a_0+(x-x_0)*\left\{\begin{array}{l}a_1+(x-x_1)*\\ \left[\begin{array}{l}a_2+(x-x_2)*\\ \left(\begin{array}{l}a_3+\cdots\\ \left[\begin{array}{l}a_{N-2}+(x-x_{N-2})*\\ (a_{N-1}+a_N(x-x_{N-1}))\end{array}\right]\end{array}\right)\end{array}\right]\end{array}\right\}$$

nested multiplication

§ Newton form of interpolation polynomial

$$P_N\left(x^*\right)=a_0+\left(x^*-x_0\right)*\left\{a_1+\left(x^*-x_1\right)*\left[a_2+\left(x^*-x_2\right)*\left(a_3+\cdots\left[a_{N-2}+\left(x^*-x_{N-2}\right)*\left(a_{N-1}+a_N\left(x^*-x_{N-1}\right)\right)\right]\right)\right]\right\}$$

STEP1: compute all a_k , $k = 0,1, \ldots, N$

STEP 2: $b_N = a_N$

$$b_{N-1}=a_{N-1}+a_N\left(x^*-x_{N-1}\right)=a_{N-1}+b_N\left(x^*-x_{N-1}\right)$$

$$b_{N-2}=a_{N-2}+\left(x^*-x_{N-2}\right)b_{N-1}$$

$$b_k=a_k+b_{k+1}\left(x^*-x_k\right) \quad , \quad k=N-1,N-2,\cdots,1,0$$

$$\boxed{P_N\left(x^*\right)=b_0}$$

STEP1: compute all a_k , $k = 0,1, \ldots, N$

Define Newton's divided difference :

$$f\left[x_i\right]\equiv y_i$$

$$f\left[x_i,x_{i+1}\right]\equiv\frac{f\left[x_{i+1}\right]-f\left[x_i\right]}{x_{i+1}-x_i}$$

$$f\left[x_i,x_{i+1},x_{i+2}\right]\equiv\frac{f\left[x_{i+1},x_{i+2}\right]-f\left[x_i,x_{i+1}\right]}{x_{i+2}-x_i}$$

$$f\left[x_i,x_{i+1},\cdots,x_{i+k}\right]\equiv\frac{f\left[x_{i+1},\cdots,x_{i+k}\right]-f\left[x_i,x_{i+1},\cdots,x_{i+k-1}\right]}{x_{i+k}-x_i}$$

$$P_N(x)=a_0+a_1(x-x_0)+a_2(x-x_0)(x-x_1)+a_3(x-x_0)(x-x_1)(x-x_2)+\cdots$$
$$+a_N(x-x_0)(x-x_1)\cdots(x-x_{N-1})$$

(i) $P_N(x_0)=a_0=y_0=f[x_0]$

(ii) $P_N(x_1)=a_0+a_1(x_1-x_0)=y_1=f[x_1] \qquad a_1=\dfrac{f[x_1]-f[x_0]}{x_1-x_0}=f[x_0,x_1]$

(iii) $P_N(x_2)=a_0+a_1(x_2-x_0)+a_2(x_2-x_0)(x_2-x_1)=y_2$

$$f[x_0]+f[x_0,x_1](x_2-x_0)+a_2(x_2-x_0)(x_2-x_1)=f[x_2]$$

$$a_2(x_2-x_0)(x_2-x_1)=f[x_2]-f[x_0]-f[x_0,x_1](x_2-x_0)$$

$$a_2(x_2-x_1)=\frac{f[x_2]-f[x_0]}{x_2-x_0}-f[x_0,x_1]$$

$$a_2=\frac{f[x_0,x_2]-f[x_0,x_1]}{x_2-x_1}=f[x_0,x_1,x_2]$$

$$a_k=f\left[x_0,x_1,\cdots,x_k\right]$$

$f[x_0]\equiv y_0=a_0$			
$f[x_1]\equiv y_1$	$f[x_0,x_1]=a_1$		
$f[x_2]\equiv y_2$	$f[x_1,x_2]$	$f[x_0,x_1,x_2]=a_2$	
$f[x_3]\equiv y_3$	$f[x_2,x_3]$	$f[x_1,x_2,x_3]$	$f[x_0,x_1,x_2,x_3]=a_3$

$$P_N(x)=\sum_{k=0}^{N}a_k\prod_{i=0}^{k-1}(x-x_i)$$

$$\boxed{P_N\left(x^*\right)=b_0}$$

$$b_N=a_N$$

$$b_k=a_k+b_{k+1}\left(x^*-x_k\right), \quad k=N-1,N-2,\cdots,1,0$$

§ Hermite interpolation

Given: $f(x_k)$ and $f'(x_k)$, $k = 0,1,\cdots,N$

Find: a polynomial passing all the points with the given slops

Theorem: If $f \in C^1[a,b]$ and $x_0, x_1, \cdots, x_N$ are distinct, the unique polynomial of least degree agreeing f and f' at $x_0, x_1, \cdots, x_N$ is the polynomialof degree $2N+1$ given by

$$H_{2N+1}(x) = \sum_{j=0}^{N} f(x_j) H_{N,j}(x) + \sum_{j=0}^{N} f'(x_j) \hat{H}_{N,j}(x)$$

$$H_{N,j}(x) = \left\{1 - 2(x - x_j) L'_{N,j}(x_j)\right\} * L^2_{N,j}(x)$$

$$\hat{H}_{N,j}(x) = (x - x_j) * L^2_{N,j}(x)$$

Theorem: If $f \in C^{2N+2}[a,b]$, then $\exists \xi \in [a,b] \ni$

$$f(x) = H_{2N+1}(x) + \frac{f^{(2N+2)}(\xi)}{(2N+2)!} \cdot \prod_{i=0}^{N} (x - x_i)^2$$

$H_{2N+1}(x)$ is called the Hermite polynomial.

§ Newton's form of Hermite polynomial

Define $\hat{x}_{2k} = \hat{x}_{2k+1} = x_k$, i.e. $\{\hat{x}_k\}_{k=0}^{2N} = \{x_0, x_0, x_1, x_1, x_2, x_2, \cdots\}$

$$H_{2N+1}(x) = a_0 + a_1(x - \hat{x}_0) + a_2(x - \hat{x}_0)(x - \hat{x}_1) + a_3(x - \hat{x}_0)(x - \hat{x}_1)(x - \hat{x}_2) + \cdots$$

$$= a_0 + a_1(x - x_0) + a_2(x - x_0)^2$$
$$+ a_3(x - x_0)^2(x - x_1) + a_4(x - x_0)^2(x - x_1)^2 + \cdots$$

$$= a_0 + (x - \hat{x}_0)\{a_1 + (x - \hat{x}_1)\{a_2 + (x - \hat{x}_2)\{\cdots$$
$$\left[a_{2N-1} + (x - \hat{x}_{2N-1})(a_{2N} + a_{2N+1}(x - \hat{x}_{2N}))\right]\}$$

Given: $x^* \in [a,b]$

Want: $f(x^*) \approx H_{2N+1}(x^*) = ?$

STEP 1: find a_k , $k = 0,1,2,\cdots,2N+1$

STEP 2: $b_{2N+1} = a_{2N+1}$

$b_k = a_k + b_{k+1} * (x^* - x_k)$ for $k = 2N, 2N-1, \cdots, 1, 0$

$b_0 = H_{2N+1}(x^*)$

§ Newton's form of Hermite polynomial

Define $\hat{x}_{2k} = \hat{x}_{2k+1} = x_k$, i.e. $\{\hat{x}_k\}_{k=0}^{2N} = \{x_0, x_0, x_1, x_1, x_2, x_2, \cdots\}$

Define $f[\hat{x}_{2k}] = f[\hat{x}_{2k+1}] = f(x_k)$

Define $\begin{cases} f[\hat{x}_{2k}, \hat{x}_{2k+1}] = f'(x_k) \\ f[\hat{x}_{2k-1}, \hat{x}_{2k}] = \dfrac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} \end{cases}$

Define $f[\hat{x}_i, \hat{x}_{i+1}, \ldots, \hat{x}_{i+k}] = \dfrac{f[\hat{x}_{i+1}, \hat{x}_{i+2}, \ldots, \hat{x}_{i+k}] - f[\hat{x}_i, \hat{x}_{i+1}, \ldots, \hat{x}_{i+k-1}]}{\hat{x}_{i+k} - \hat{x}_i}$

for $k \geq 3$

$\hat{x}_k$:	$\hat{x}_0$	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$	$\hat{x}_4$	$\hat{x}_5$	$\cdots$
$\hat{x}_k$:	x_0	x_0	x_1	x_1	x_2	x_2	$\cdots$
1st :	$f(x_0)$	$f(x_0)$	$f(x_1)$	$f(x_1)$	$f(x_2)$	$f(x_2)$	$\cdots$
2nd :		$f'(x_0)$	DD	$f'(x_1)$	DD	$f'(x_2)$	$\cdots$
3rd :			DD	DD	DD	DD	$\cdots$

$a_k = f[\hat{x}_0, \hat{x}_1, \cdots, \hat{x}_k]$

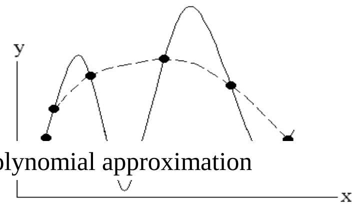
DD: Newton divided difference

§ Cubic spline interpolation ~ piecewise cubic polynomials

Given: $(x_k, y_k = f(x_k))$, $k = 0$

high-order polynomials => o

Alternative choice: piecewise polynomial approximation

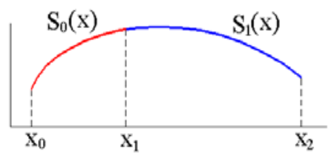


Piecewise linear interpolation:
disadvantage: discontinuous first derivatives at nodes in general

Piecewise cubic interpolation:
Goal: continuous 1st and 2nd derivatives at nodes

Use a cubic polynomial to fit for each subinterval, $S_k(x)$

for $x \in [x_k, x_{k+1}]$



C^0 : $S_0(x_0) = y_0$ $\quad S_0(x_1) = y_1$

$S_1(x_1) = y_1$ $\quad S_1(x_2) = y_2$

C^1 : $S_0'(x_1) = S_1'(x_1)$

C^2 : $S_0''(x_1) = S_1''(x_1)$

6 constraints
8 degrees of freedom

Cubic Spline Interpolation:

Let $a = x_0 < x_1 < \cdots < x_{N-1} < x_N = b$. A cubic spline interpolant

$S(x)$ for $f(x)$ is a function that satisfies the following conditions:

(i) $S(x)$ is a cubic polynomial on the interval $[x_k, x_{k+1}]$,

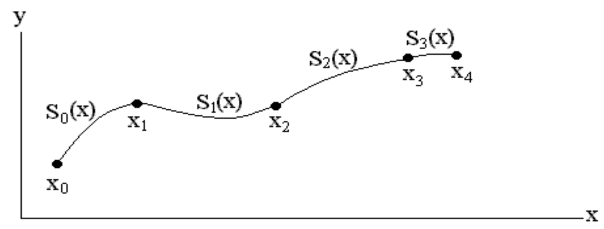
denoted by $S_k(x)$, for $k = 0,1,2,\cdots,N-1$

(ii) $S_k(x_k) = y_k$ and $S_k(x_{k+1}) = y_{k+1}$, for $k = 0,1,2,\cdots,N-1$ $\quad 2N$

(iii) $S'_k(x_{k+1}) = S'_{k+1}(x_{k+1})$, for $k = 0,1,2,\cdots,N-2$ $\quad N-1$

(iv) $S''_k(x_{k+1}) = S''_{k+1}(x_{k+1})$, for $k = 0,1,2,\cdots,N-2$ $\quad N-1$

$4N$ degrees of freedom!



boundary conditions:

(a) free or natural spline: $S''_0(x_0) = S''_{N-1}(x_N) = 0$

(b) clamped spline: $S'_0(x_0) = f'(x_0)$ and $S'_{N-1}(x_N) = f'(x_N)$

for $k = 0,1,2,\cdots,N-1$

write $S_k(x) = a_k + b_k(x - x_k) + c_k(x - x_k)^2 + d_k(x - x_k)^3$

$$S'_k(x) = b_k + 2c_k(x - x_k) + 3d_k(x - x_k)^2$$

$$S''_k(x) = 2c_k + 6d_k(x - x_k)$$

define $h_k \equiv x_{k+1} - x_k$ for $k = 0,1,\cdots,N-1$

C^0 : $S_k(x_k) = a_k = y_k$

$$S_k(x_{k+1}) = a_k + b_k h_k + c_k h_k^2 + d_k h_k^3 = y_{k+1}$$

for $k = 0,1,\cdots,N-1$

C^1 : for $k = 0,1,\cdots,N-2$

$$S'_k(x) = b_k + 2c_k(x - x_k) + 3d_k(x - x_k)^2$$

$$S'_{k+1}(x) = b_{k+1} + 2c_{k+1}(x - x_{k+1}) + 3d_{k+1}(x - x_{k+1})^2$$

$$S'_k(x_{k+1}) = b_k + 2c_k h_k + 3d_k h_k^2$$

$$S'_{k+1}(x_{k+1}) = b_{k+1}$$

$$b_k + 2c_k h_k + 3d_k h_k^2 = b_{k+1}$$

C^2 : for $k = 0,1,\cdots,N-2$

$$\left.\begin{aligned} S''_k(x) &= 2c_k + 6d_k(x - x_k) \\ S''_{k+1}(x) &= 2c_{k+1} + 6d_{k+1}(x - x_{k+1}) \end{aligned}\right\} \quad 2c_k + 6d_k h_k = 2c_{k+1}$$

Summary:

$(i)\quad y_k = a_k \qquad (k = 0,1,2,\cdots N-1)$

$(ii)\quad y_{k+1} = a_k + b_k h_k + c_k h_k^2 + d_k h_k^3 \quad (k = 0,1,\cdots,N-1)$

$(iii)\quad b_k + 2c_k h_k + 3d_k h_k^2 = b_{k+1} \qquad (k = 0,1,\cdots,N-2)$

$(iv)\quad 2c_k + 6d_k h_k = 2c_{k+1} \qquad (k = 0,1,\cdots,N-2)$

From (iv): $d_k = d_k(\vec{c}) \qquad (v)$

Substitute (v) into (ii): $b_k = b_k(\vec{a},\vec{c}) \quad (vi)$

Substitute $(v)(vi)$ into (iii): $c_k = c_k(\vec{a},\vec{c})$

For $k = 1,2,\ldots,N-1$

$$h_{k-1}c_{k-1} + 2(h_{k-1}+h_k)c_k + h_k c_{k+1} = \frac{3}{h_k}(a_{k+1}-a_k) - \frac{3}{h_{k-1}}(a_k - a_{k-1})$$

$a_N \equiv S_{N-1}(x_N) = y_N = a_{N-1} + b_{N-1}h_{N-1} + c_{N-1}h_{N-1}^2 + d_{N-1}h_{N-1}^3$

$2c_N \equiv S''_{N-1}(x_N) = 2c_{N-1} + 6d_{N-1}h_{N-1}$

~ $(N-1)$ equations for $(N+1)$ unknowns $\{c_k\}_{k=0}^N$

boundary conditions:

(a) free or natural spline: $S''_0(x_0) = S''_{N-1}(x_N) = 0$

$c_0 = c_N = 0$

(b) clamped spline: $S'_0(x_0) = f'(x_0)$ and $S'_{N-1}(x_N) = f'(x_N)$

$S'_0(x_0) = b_0 = f'(x_0), \quad S'_{N-1}(x_N) = b_{N-1} + 2c_{N-1}h_{N-1} + 3d_{N-1}h_{N-1}^2 = f'(x_N)$

(a) free or natural spline: $c_0 = c_N = 0$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & & & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \cdots & & & 0 \\ 0 & h_1 & 2(h_1+h_2) & h_2 & & & & \\ & & & \vdots & & & & \\ 0 & 0 & 0 & h_{k-1} & 2(h_{k-1}+h_k) & h_k & 0 \cdots & 0 \\ & & & & \vdots & & & \\ 0 & 0 & & & & h_{N-2} & 2(h_{N-2}+h_{N-1}) & h_{N-1} \\ 0 & 0 & \cdots & & & & 0 & 1 \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ \\ \\ \\ c_N \end{pmatrix} = \begin{pmatrix} 0 \\ r_1 \\ r_2 \\ \vdots \\ \\ \\ r_{N-1} \\ 0 \end{pmatrix}$$

$$r_k \equiv \frac{3}{h_k}(a_{k+1}-a_k) - \frac{3}{h_{k-1}}(a_k - a_{k-1})$$

~ tridiagonal matrix ~

(a) clamped spline: $\begin{cases} f'(x_0) = (a_1 - a_0)/h_0 - h_0(2c_0 + c_1)/3 \\ f'(x_N) = (a_N - a_{N-1})/h_{N-1} + h_{N-1}(c_{N-1} + 2c_N)/3 \end{cases}$

$$\begin{pmatrix} -2h_0/3 & -h_0/3 & 0 & 0 & \cdots & & & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \cdots & & & 0 \\ 0 & h_1 & 2(h_1+h_2) & h_2 & & & & \\ & & & \vdots & & & & \\ 0 & 0 & 0 & h_{k-1} & 2(h_{k-1}+h_k) & h_k & 0 \cdots & 0 \\ & & & \vdots & & & & \\ 0 & 0 & & & & h_{N-2} & 2(h_{N-2}+h_{N-1}) & h_{N-1} \\ 0 & 0 & \cdots & & & & h_{N-1}/3 & 2h_{N-1}/3 \end{pmatrix}$$

$\begin{cases} r_0 = f'(x_0) - (a_1 - a_0)/h_0 \\ r_N = f'(x_N) - (a_N - a_{N-1})/h_{N-1} \end{cases}$

~ tridiagonal matrix ~

§ Triagonal linear system: $Ax = r$

$$
\begin{pmatrix}
a_{11} & a_{12} & 0 & 0 & \cdots & & & 0 \\
a_{21} & a_{22} & a_{23} & 0 & \cdots & & & 0 \\
0 & a_{32} & a_{33} & a_{34} & & & & \\
 & & & \ddots & & & & \\
 & & & & \ddots & & & \\
0 & 0 & & a_{k,k-1} & a_{k,k} & a_{k,k+1} & & 0 \\
 & & & & & \ddots & & \\
 & & & & & & \ddots & \\
0 & 0 & & & a_{N-1,N-2} & a_{N-1,N-1} & a_{N-1,N} & \\
0 & 0 & & \cdots & & a_{N,N-1} & a_{N,N} &
\end{pmatrix}
\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ \\ \\ \\ \\ \\ x_{N-1} \\ x_N \end{pmatrix}
=
\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ \\ \\ \\ \\ \\ b_{N-1} \\ b_N \end{pmatrix}
$$

Given: x^*

Want: an approximation of $f\left(x^*\right)$

Step 1: find the subinterval that x^* belongs to, say $x^* \in \left[x_k, x_{k+1}\right]$

Step 2: compute $S_k\left(x^*\right) = a_k + b_k\left(x^* - x_k\right) + c_k\left(x^* - x_k\right)^2 + d_k\left(x^* - x_k\right)^3$

$$\approx f\left(x^*\right)$$

Theorem: Let $f \in C^4[a,b]$ with $\underset{a \le x \le b}{Max}\left|f^{(iv)}(x)\right| = M$. If $S(x)$ is the unique clamped cubic spline interpolant to f with respect to the nodes $a \le x_0 < x_1 < \cdots < x_N = b$. Then

$$\underset{a \le x \le b}{Max}\left|f(x) - S(x)\right| \le \frac{5M}{384} \cdot \underset{0 \le j \le N-1}{Max}\left(x_{j+1} - x_j\right)^4$$

$c.f.$ natural spline: 4th order too.

§ Boundary conditions for spline interpolations

(i) Natural spline

$$f''(x_0) = 0$$
$$f''(x_N) = 0$$

This choice minimizes the value of $\int_a^b f''^2(x)\,dx$.

(ii) Parabolic runout

$$f''(x_0) = f''(x_1)$$
$$f''(x_N) = f''(x_{N-1})$$

implying $f'''(x_0) = f'''(x_N) = 0$

(parabolic ending curves)

(iii) clamped spline

$f'(x_0)$, $f''(x_0)$, or $f'''(x_0)$ known

$f'(x_N)$, $f''(x_N)$, or $f'''(x_N)$ known

(iv) Fit a cubic polynomial $C(x)$ to the four ending points.

$$f'''(x_0) \approx C'''(x_0) \approx \frac{f''(x_1) - f''(x_0)}{h_0}$$

$$f'''(x_N) \approx C'''(x_N) \approx \frac{f''(x_N) - f''(x_{N-1})}{h_{N-1}}$$

(v) Periodic boundary condition

$$f(x_0) = f(x_N)$$
$$f'(x_0) = f'(x_N)$$
$$f''(x_0) = f''(x_N)$$

§ Tension spline interpolation

Let $a = x_0 < x_1 < \cdots < x_{N-1} < x_N = b$. A tension spline interpolant $S(x)$ for $f(x)$ is a function that satisfies the following conditions:

(i) $S(x)$ is a cubic polynomial on the interval $[x_k, x_{k+1}]$,

denoted by $S_k(x)$, for $k = 0,1,2,\cdots,N-1$

(ii) $S_k(x_k) = y_k$ and $S_k(x_{k+1}) = y_{k+1}$, for $k = 0,1,2,\cdots,N-1$ $\boxed{2N}$

(iii) $S_k'(x_{k+1}) = S_{k+1}'(x_{k+1})$, for $k = 0,1,2,\cdots,N-2$ $\boxed{N-1}$

(iv) $S_k''(x_{k+1}) = S_{k+1}''(x_{k+1})$, for $k = 0,1,2,\cdots,N-2$ $\boxed{N-1}$

(v) $S_k(x)$ is a solution of $S^{(iv)} - \tau^2 S'' = 0$.

(v) $S_k(x)$ is a solution of $S^{(iv)} - \tau^2 S'' = 0$.

- a special case $\tau = 0$ $\Rightarrow$ cubic polynomials $\Rightarrow$ cubic spline!
- another special case $\tau = \infty$ $\Rightarrow$ approximately linear fitting (still C^2)

for $x \in [x_k, x_{k+1}]$ $\quad S_k^{(iv)} - \tau^2 S_k'' = 0$

$$C^0 \begin{cases} S_k(x_k) = y_k \\ S_k(x_{k+1}) = y_{k+1} \end{cases} \quad \text{known}$$

$$C^2 \begin{cases} S_k''(x_k) = z_k \\ S_k''(x_{k+1}) = z_{k+1} \end{cases} \quad \text{unknown}$$

---

§ Tension spline interpolation

$$S_k(x) = \{z_k \sinh[\tau(x_{k+1} - x)] + z_{k+1} \sinh[\tau(x - x_k)]\} / \tau^2 \sinh(\tau h_k)$$
$$+ \left(y_k - z_k/\tau^2\right)(x_{k+1} - x)/h_k + \left(y_{k+1} - z_{k+1}/\tau^2\right)(x - x_k)/h_k$$

$$C^1 : \; S_k'(x_{k+1}) = S_{k+1}'(x_{k+1})$$

$$\alpha_{k-1} z_{k-1} + (\beta_{k-1} + \beta_k) z_k + \alpha_k z_{k+1} = \gamma_k - \gamma_{k-1}$$

$$\alpha_k = 1/h_k - \tau/\sinh(\tau h_k)$$
$$\beta_k = \tau \cosh(\tau h_k)/\sinh(\tau h_k) - 1/h_k$$
$$\gamma_k = \tau^2 (y_{k+1} - y_k)/h_k$$

for $k = 1,2,\cdots,N-1$

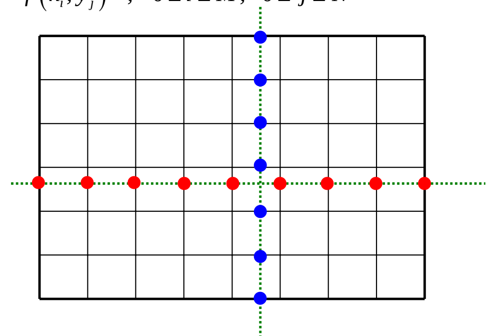
$(N-1)$ equations for $(N+1)$ unknowns $\{z_k = S''(x_k)\}$, $k = 0,1,2,\cdots,N$

---

§ Two dimensional interpolations

- Cartesian Product and Grid

Given: $f_{ij} = f(x_i, y_j)$, $0 \le i \le M$, $0 \le j \le N$



Solution: 2D interpolation => two-layer 1D interpolations

First, interpolate in one direction

$$f(x,y) \approx \sum_{i=0}^{M} f(x_i, y) * L_{M,i}(x) \qquad L_{M,i}(x) = \prod_{\substack{k=0\\k\neq i}}^{M} \frac{(x-x_k)}{(x_i-x_k)}$$

$$f(x,y) \approx \sum_{j=0}^{N} f(x, y_j) * L_{N,j}(y) \qquad L_{N,j}(y) = \prod_{\substack{k=0\\k\neq j}}^{N} \frac{(y-y_k)}{(y_j-y_k)}$$

$$\Rightarrow f(x_i,y) \approx \sum_{j=0}^{N} f(x_i, y_j) * L_{N,j}(y) = \sum_{j=0}^{N} f_{ij} * L_{N,j}(y)$$

$$\Rightarrow f(x,y) \approx \sum_{i=0}^{M}\sum_{j=0}^{N} f_{ij} * L_{N,j}(y) L_{M,i}(x)$$

- Irregular Grid

Given: $f_k = f(x_k, y_k)$, for $k = 0,1,2,\cdots,N$

Possible solution: a polynomial to fit and interpolate.

$\Pi_m(R^2)$ = the set consisting of all real-coefficient polynomials with two variables x and y of degree at most m.

If $P_m(x,y) \in \Pi_m(R^2)$, $P_m(x,y) = \sum_{0\le r+s\le m} C_{rs} x^r y^s$

degrees of freedom = $1+2+3+\cdots+(m+1) = (m+1)(m+2)/2$

Wish $f(x,y) \approx P_m(x,y)$

want $f(x_k,y_k) = f_k = P_m(x_k,y_k)$ for $k = 0,1,2,\cdots,N$

unique choice if $N+1 = (m+1)(m+2)/2$

example: $N+1 = 3 = (m+1)(m+2)/2$ or $m=1$

$$P_1(x,y) = C_{00} + C_{10}x + C_{01}y$$

$$P_1(x_k,y_k) = C_{00} + C_{10}x_k + C_{01}y_k = f_k \quad \text{for } k = 0,1,2$$

$$\begin{pmatrix} 1 & x_0 & y_0 \\ 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \end{pmatrix}\begin{pmatrix} C_{00} \\ C_{10} \\ C_{01} \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \end{pmatrix}$$

example: $N+1 = 6 = (m+1)(m+2)/2$ or $m=2$

$$P_2(x,y) = C_{00} + C_{10}x + C_{01}y + C_{20}x^2 + C_{11}xy + C_{02}y^2$$

$$\begin{pmatrix} 1 & x_0 & y_0 & x_0^2 & x_0y_0 & y_0^2 \\ 1 & x_1 & y_1 & x_1^2 & x_1y_1 & y_1^2 \\ 1 & x_2 & y_2 & x_2^2 & x_2y_2 & y_2^2 \\ 1 & x_3 & y_3 & x_3^2 & x_3y_3 & y_3^2 \\ 1 & x_4 & y_4 & x_4^2 & x_4y_4 & y_4^2 \\ 1 & x_5 & y_5 & x_5^2 & x_5y_5 & y_5^2 \end{pmatrix}\begin{pmatrix} C_{00} \\ C_{10} \\ C_{01} \\ C_{20} \\ C_{11} \\ C_{02} \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix}$$

Matrix invertable? Not always!

Theorem: Interpolation of arbitray data by the subspace $\Pi(R^2)$ is possible on a set of $(m+1)(m+2)/2$ nodes if these nodes lie on lines $L_0, L_1, \ldots, L_m$ in such a way that for each L_i contains exactly $(i+1)$ nodes.

§ Shepard interpolation method

Write $P=(x,y)$ represents a point in the R^2 space. Q is another point.

Let $\Phi(P,Q)$ be a real-valued function on $R^2 \times R^2$ which satisfies

$\Phi(P,Q) = 0$ if and only if $P = Q$

e.g. $\Phi(P,Q) \equiv |P-Q|^2 = (x_p - x_q)^2 + (y_p - y_q)^2$

Define 2D Lagrange interpolation function as

$$L_{N,i}(P) \equiv \prod_{\substack{k=0 \\ k \neq i}}^{N} \frac{\Phi(P, P_k)}{\Phi(P_i, P_k)}$$

or written as $$L_{N,i}(x,y) \equiv \prod_{\substack{k=0 \\ k \neq i}}^{N} \frac{\Phi(x,y;x_k,y_k)}{\Phi(x_i,y_i;x_k,y_k)}$$

$$L_{N,i}(P_j) \equiv \prod_{\substack{k=0 \\ k \neq i}}^{N} \frac{\Phi(P_j, P_k)}{\Phi(P_i, P_k)} = \delta_{ij}$$

$$f(x,y) \approx P(x,y) = \sum_{i=0}^{N} f_i * L_{N,i}(x,y)$$

$$P(x_j, y_j) = \sum_{i=0}^{N} f_i * L_{N,i}(x_j, y_j) = \sum_{i=0}^{N} f_i * \delta_{ij} = f_j$$

References: Kincaid & Cheny

Lancaster & Salkauskas "Curve & Surface Fitting" (1980)