Chap 5 Piecewise Interpolation and Calculus

Calculus —discretization→ (Linear) Algebra

- differentiation
- integration

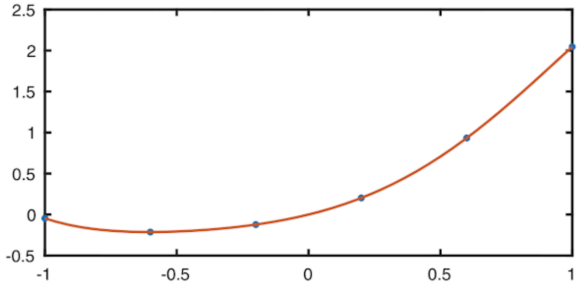
- polynomial
- nonlinear functions

§5.1 The Interpolation Problem

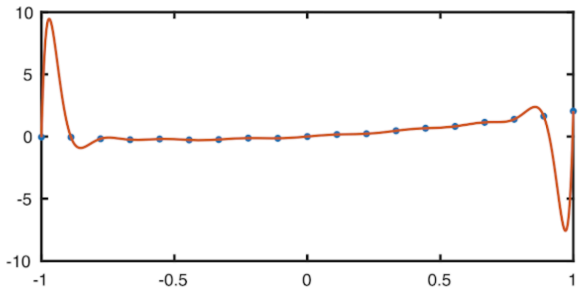
Interpolation problem: Given $n+1$ distinct points (t_0, y_0), (t_1, y_1), …, (t_n, y_n), with $t_0 < t_1 < \cdots < t_n$, find a function $p(x)$, called the **interpolant**, such that $p(t_k) = y_k$ for $k = 0, \ldots, n$.

One polynomial for all points (Global Polynomial)

```
n = 5;
t = linspace(-1,1,n+1)';  y = t.^2 + t + 0.05*sin(20*t);
plot(t,y,'.')
c = polyfit(t,y,n);      % polynomial coefficients
p = @(x) polyval(c,x);
hold on, fplot(p,[-1 1])
```



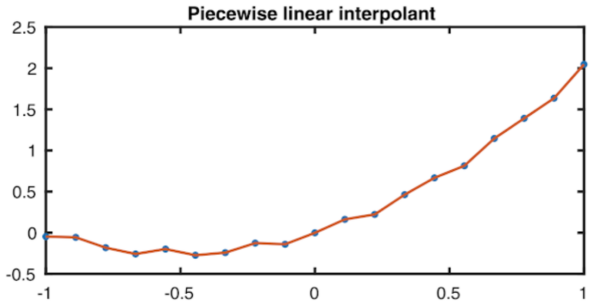

```
n = 18;
t = linspace(-1,1,n+1)';  y = t.^2 + t + 0.05*sin(20*t);
clf, plot(t,y,'.')
c = polyfit(t,y,n);      % polynomial coefficients
p = @(x) polyval(c,x);
hold on, fplot(p,[-1 1])
```



Piecewise polynomials

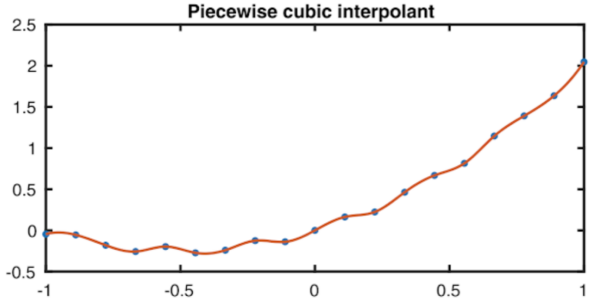

```
n = 18;
t = linspace(-1,1,n+1)';  y = t.^2 + t + 0.05*sin(20*t);
clf, plot(t,y,'.')
```

```
x = linspace(-1,1,400)';
hold on, plot(x,interp1(t,y,x))
```



We may instead request a smoother interpolant that is piecewise cubic.

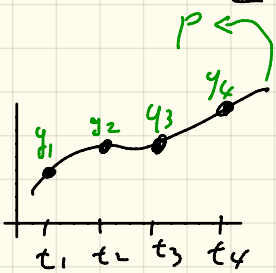
```
cla
plot(t,y,'.')
plot(x,interp1(t,y,x,'spline'))
```



Conditioning of Interpolation

- Let $\mathcal{I}: \mathbb{R}^n \longrightarrow$ interpolant p (in a function space)

  $\mathcal{I}(y) = p$ where $p(t_k) = y_k$



- Assume p is linear to simplify the analysis

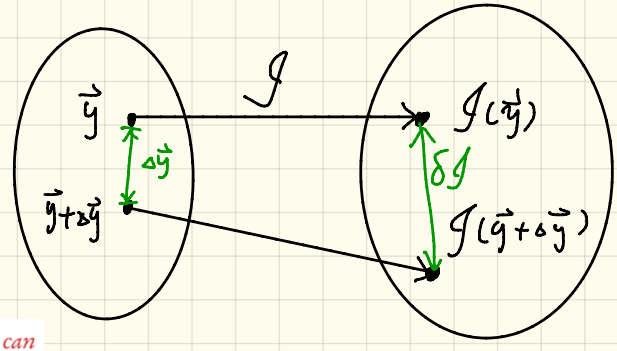
  (i.e. $\mathcal{I}(\alpha\vec{y} + \beta\vec{z}) = \alpha\,\mathcal{I}(\vec{y}) + \beta\,\mathcal{I}(\vec{z})$)

Theorem 5.1.1: Conditioning of interpolation

Suppose that $\mathcal{I}$ is a linear interpolation method. Then the absolute condition number of $\mathcal{I}$ satisfies

$$\max_{0\le k\le n} \|\mathcal{I}(\boldsymbol{e}_k)\|_\infty \le \kappa(\boldsymbol{y}) \le \sum_{k=0}^{n} \|\mathcal{I}(\boldsymbol{e}_k)\|_\infty \qquad (5.1.4)$$

if vectors and functions are measured in the infinity norm.



Theorem 5.1.1 says that *assessing the condition number of interpolation for any data can be simplified to measuring the effect of interpolation with each node "switched on" one at a time.* The results of such interpolations are known as **cardinal functions.**

基本函數

pf: $\delta f = f(y+\Delta y) - f(y) = f(y) + f(\Delta y) - f(y) = f(\Delta y)$

Notation:

$$\begin{bmatrix} e_0 & e_1 & \cdots & e_n \end{bmatrix} \quad \text{1st column}$$

$$= f\left(\begin{bmatrix} \Delta y_1 \\ \Delta y_2 \\ \vdots \\ \Delta y_n \end{bmatrix}\right) = f\left(\begin{bmatrix} \Delta y_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \Delta y_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \cdots + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \Delta y_n \end{bmatrix}\right)$$

$$= f(\Delta y_0 e_0 + \cdots + \Delta y_n e_n) = f\left[\sum_{k=0}^{n} (\Delta y)_k f(e_k)\right] \quad \text{---(†)}$$

① Right inequality

$$k(y) = \frac{\|\delta f\|_\infty}{\|\Delta f\|_\infty} \begin{array}{l} \rightarrow \text{Output (function norm)} \\ \rightarrow \text{Input (vector norm)} \end{array}$$

$$\text{by (†)} = \frac{\|f[\Sigma (\Delta y)_k f(e_k)]\|}{\|\Delta y\|_\infty} \leq \frac{|(\Delta y)_0| \|f(e_0)\| - \cdots + |(\Delta y)_n| \cdot \|f(e_n)\|}{\|\Delta y\|_\infty}$$

$$\frac{|(\Delta y)_k|}{\|\Delta y\|_\infty} \leq 1 \quad \leq \|f(e_0)\| + \cdots + \|f(e_n)\| = \sum_{k=0}^{n} \|f(e_k)\|_\infty$$

② Left inequality

Suppose $\|g(e_j)\| \geq \|g(e_0)\|, \cdots, \|g(e_n)\|$

Take $\Delta y = e_j = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ jth

$$k(y) = \frac{\|\delta g\|_\infty}{\|\Delta y\|_\infty} = \frac{\|g[\Sigma(\Delta y)_k g(e_k)]\|_\infty}{\|\Delta y\|_\infty} = \frac{\|g(e_j)\|_\infty}{1}$$

$$= \max_{0 \leq k \leq n} \|g(e_k)\|$$

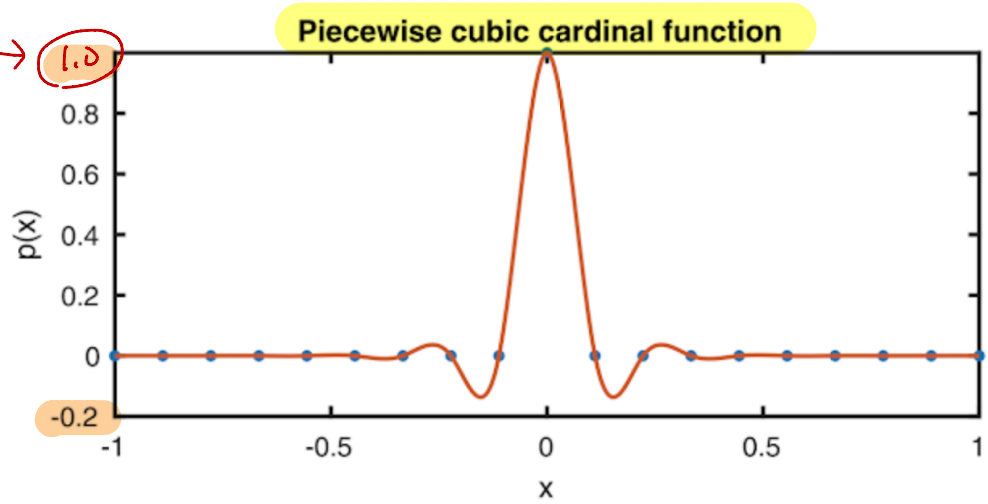
$\Rightarrow k(y) \geq \max_{0 \leq k \leq n} \|g(e_k)\|$

所有的情況中最大的

(存在一個例子，有這麼大)

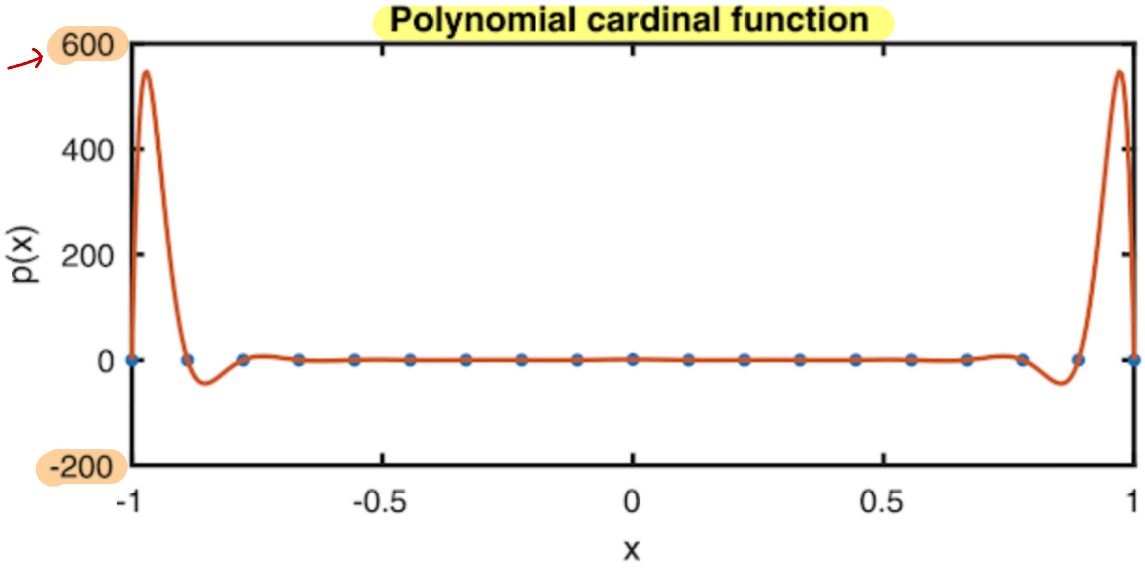
```
n = 18;
t = linspace(-1,1,n+1)';
y = [zeros(9,1);1;zeros(n-9,1)];  % 10th cardinal function
plot(t,y,'.'), hold on

x = linspace(-1,1,400)';
plot(x,interp1(t,y,x,'spline'))
```

Smaller cond. #



The piecewise cubic cardinal function is nowhere greater than one in absolute value. This happens to be true for all the cardinal functions, ensuring a good condition number for the interpolation. But the story for global polynomials is very different.

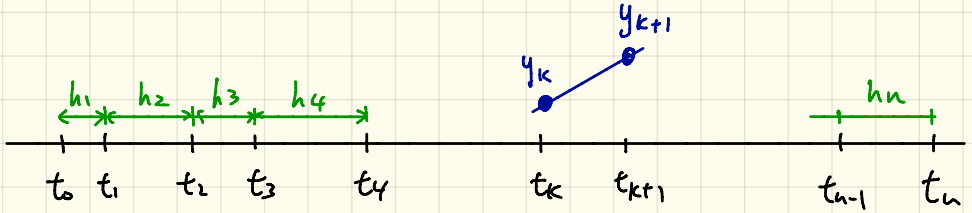
```
c = polyfit(t,y,n);
hold on, plot(x,polyval(c,x))
```



Bigger cond. #

From the figure we can see that the condition number for polynomial interpolation on these nodes is at least 500.

§ 5.2 Piecewise Linear Interpolation



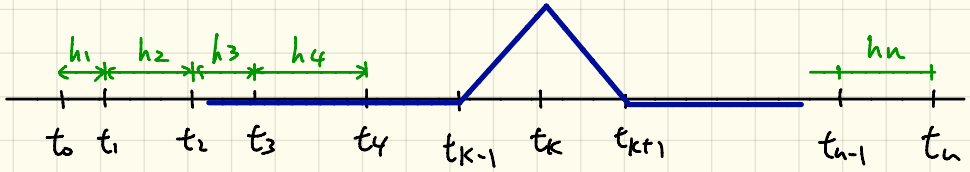
$(n+1)$ points

Idea 1

slope

$$p(x) = y_k + \frac{y_{k+1} - y_k}{t_{k+1} - t_k}(x - t_k) \qquad \forall\, x \in [t_k, t_{k+1}]$$

Idea 2 Hat function (linear combination of basis functions)



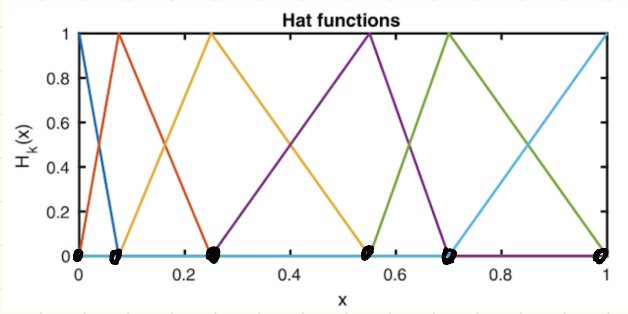
$(n+1)$ points

$$H_k(x) = \begin{cases} \dfrac{x - t_{k-1}}{t_k - t_{k-1}} & \text{if } x \in [t_{k-1}, t_k], \\ \dfrac{t_{k+1} - x}{t_{k+1} - t_k} & \text{if } x \in [t_k, t_{k+1}], \\ 0 & \text{otherwise,} \end{cases} \qquad k = 0, \ldots, n.$$

$$\Rightarrow \sum_{k=0}^{n} c_k H_k(x)$$

Note :

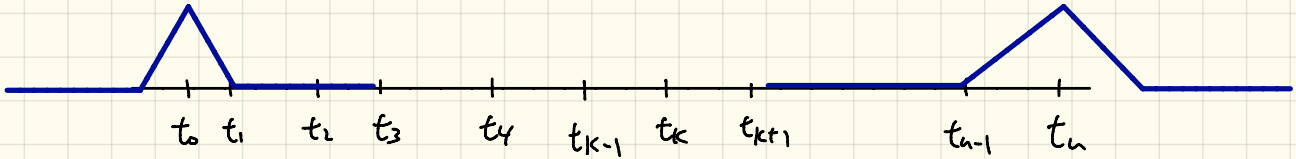
- Basis and linear combination



$$p(x) = \sum_{k=0}^{n} y_k H_k(x)$$

- Cardinality conditions : $H_k(t_i) = \begin{cases} 1 & \text{if } i = k \\ 0 & \text{otherwise} \end{cases}$

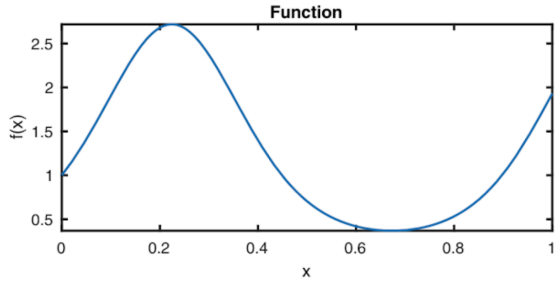
- H_0 & H_n for code uniformality



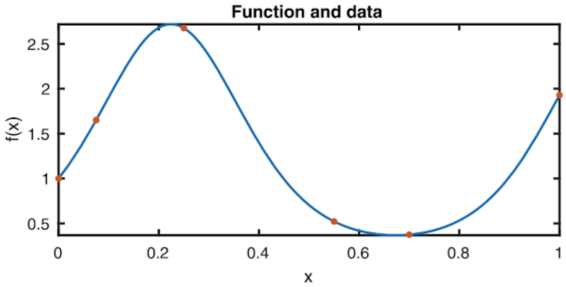
Function 5.2.1 (hatfun) Hat function/piecewise linear basis function.

```
function H = hatfun(x,t,k)
% HATFUN   Hat function/piecewise linear basis function.
% Input:
%   x       evaluation points (vector)
%   t       interpolation nodes (vector, length n+1)
%   k       node index (integer, in 0,...,n)
% Output:
%   H       values of the kth hat function

n = length(t)-1;
k = k+1;  % adjust for starting with index=1

% Fictitious nodes to deal with first, last funcs.
t = [ 2*t(1)-t(2); t(:); 2*t(n+1)-t(n) ];
k = k+1;  % adjust index for the fictitious first node

H1 = (x-t(k-1))/(t(k)-t(k-1));   % upward slope
H2 = (t(k+1)-x)/(t(k+1)-t(k));   % downward slope

H = min(H1,H2);
H = max(0,H);
```

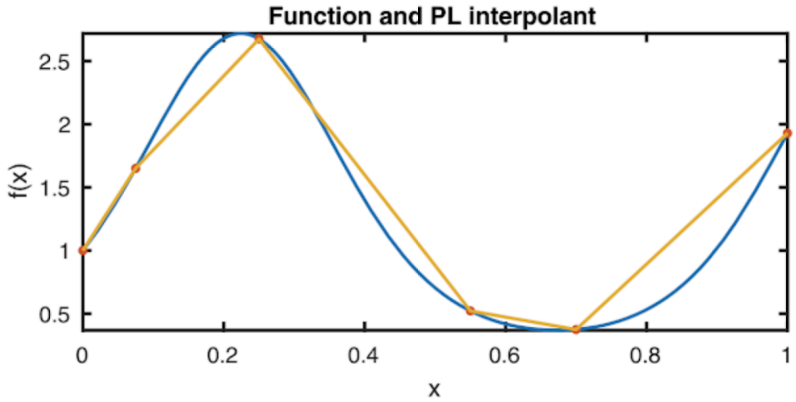
```
f = @(x) exp(sin(7*x));
fplot(f,[0 1])
```




```
t = [0 0.075 0.25 0.55 0.7 1];    % nodes
y = f(t);
hold on, plot(t,y,'.')
```




```
p = plinterp(t,y);
fplot(p,[0 1])
```



Conditioning (Well-conditioned!)

Theorem 5.2.1

The absolute condition number of piecewise linear interpolation in the infinity norm equals one. More specifically, if $\mathcal{I}$ is the piecewise linear interpolation operator, then

$$\|\mathcal{I}(y+z)-\mathcal{I}(y)\|_\infty = \|z\|_\infty. \tag{5.2.6}$$

(The norm on the left side is on functions, while the norm on the right side is on vectors.)

Convergence: 2nd order

Theorem 5.2.2

Suppose that $f(x)$ has a continuous second derivative in $[a,b]$ (often expressed as $f \in C^2[a,b]$). Let $p_n(x)$ be the piecewise linear interpolant of $\big(t_i, f(t_i)\big)$ for $i = 0,\ldots,n$, where $t_i = a+ih$ and $h = (b-a)/n$. Then

$$\|f-p_n\|_\infty = \max_{x\in[a,b]} |f(x)-p(x)| \le Mh^2, \tag{5.2.7}$$

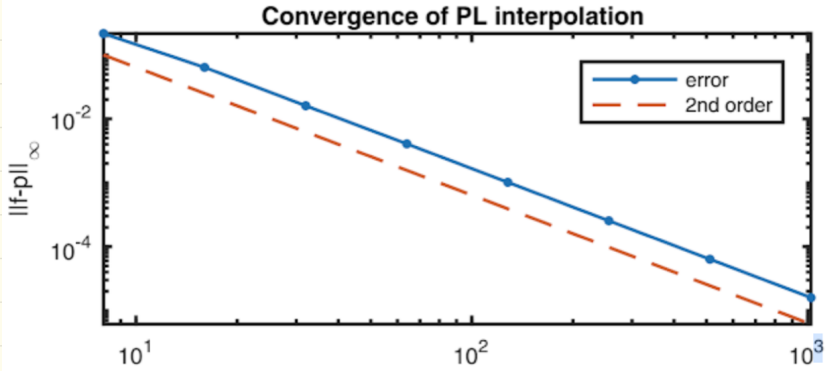
where $M = \|f''\|_\infty$.

We measure the convergence rate for piecewise linear interpolation of $e^{\sin 7x}$.

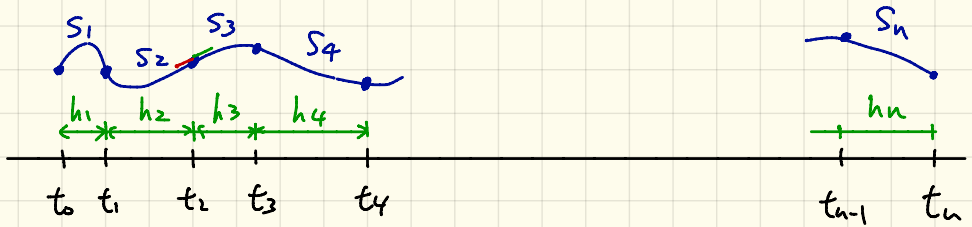
```
f = @(x) exp(sin(7*x));
x = linspace(0,1,10001)';  % sample the difference at many
    points
n_ = 2.^(3:10)';
err_ = 0*n_;
for i = 1:length(n_)
    n = n_(i);
    t = linspace(0,1,n+1)';   % interpolation nodes
    p = plinterp(t,f(t));
    err = max(abs( f(x) - p(x) ));
    err_(i) = err;
end
```

Since we expect convergence that is $O(h^2) = O(n^{-2})$, we use a log–log graph of error and expect a straight line of slope −2.

```
loglog( n_, err_, '.-' )
hold on, loglog( n_, 0.1*(n_/n_(1)).^(-2), '--' )
```



§5.3 Cubic Splines



can be non-uniformed

$$\begin{cases} h_k = t_k - t_{k-1} \\ S_k = a_k + b_k(x - t_{k-1}) + c_k(x - t_{k-1})^2 + d_k(x - t_{k-1})^3 \end{cases}$$

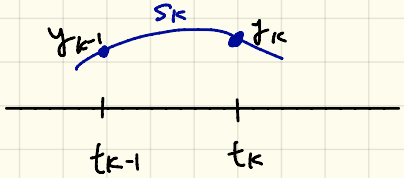
$k = 1 : n$ (n 個 區間 段) $\Rightarrow$ $4n$ unknowns

$4n$ conditions

(1) Interpolation by S_k at both of its endpoints	$2n$
(2) Continuity of $S'(x)$ at interior points	$(n-1)$
(3) : $S''(x)$:	$(n-1)$
(4) 2 End constraints at t_0 & t_n	2

(1) Interpolation by S_k at both of its endpoints

$$S_k = a_k + b_k(x - t_{k-1}) + c_k(x - t_{k-1})^2 + d_k(x - t_{k-1})^3$$



Left $\left\{ S_k\big|_{x=t_{k-1}} \Rightarrow a_k = y_{k-1} \right.$

Right $\left\{ S_k\big|_{x=t_k} \Rightarrow a_k + b_k h + c_k h_k^2 + d_k h_k^3 = y_k \right.$

$k = 1:n$

Left

$$\begin{bmatrix} 1 & & & & 0 & & 0 & 0 \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \end{bmatrix}_{n \times 4n} \begin{bmatrix} a_1 \\ \vdots \\ a_n \\ \hline b_1 \\ \vdots \\ b_n \\ \hline c_1 \\ \vdots \\ c_n \\ \hline d_1 \\ \vdots \\ d_n \end{bmatrix}_{4n \times 1} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix}_{n \times 1} \Rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} y_0 \\ \vdots \\ y_n \end{bmatrix}$$

Right $S_k \big|_{x=t_k} \Rightarrow a_k + b_k h_k + c_k h_k^2 + d_k h_k^3 = y_k \qquad k = 1:n$

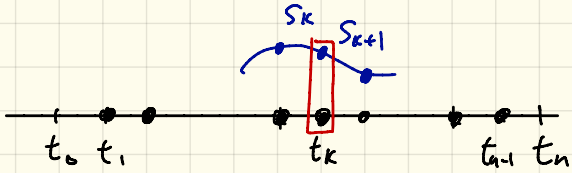
$$\begin{bmatrix} 1 & & & & h_1 & & & h_1^2 & & & h_1^3 & & \\ & 1 & & & & h_2 & & & h_2^2 & & & h_2^3 & \\ & & \ddots & & & & \ddots & & & \ddots & & & \ddots \\ & & & 1 & & & h_n & & & h_n^2 & & & h_n^3 \end{bmatrix}_{n \times 4n} \begin{bmatrix} a_1 \\ \vdots \\ a_n \\ \hline b_1 \\ \vdots \\ b_n \\ \hline c_1 \\ \vdots \\ c_n \\ \hline d_1 \\ \vdots \\ d_n \end{bmatrix}_{4n \times 1} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}_{n \times 1}$$

$$\Rightarrow \quad \begin{bmatrix} I & H & H^2 & H^3 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

(2) Continuity of $S'(x)$ at interior points

$$S_k = a_k + b_k(x-t_{k-1}) + c_k(x-t_{k-1})^2 + d_k(x-t_{k-1})^3$$

$$\Rightarrow \begin{cases} S_k' = b_k + 2c_k(x-t_{k-1}) + 3d_k(x-t_{k-1})^2 \Big|_{x=t_k} \\ S_{k+1}' = b_{k+1} + 2c_{k+1}(x-t_k) + 3d_{k+1}(x-t_k)^2 \Big|_{x=t_k} \end{cases} \qquad k = 1 : (n-1)$$



$$\Rightarrow S_k'\Big|_{x=t_k} = b_k + 2c_k h_k + 3d_k h_k^2 = S_{k+1}'\Big|_{x=t_k} = b_{k+1}$$

$$\Rightarrow (b_k - b_{k+1}) + 2h_k \cdot c_k + 3h_k^2 \cdot d_k = 0 \qquad k = 1 : (n-1)$$

$$\begin{bmatrix} 1 & 0 & & & \\ & 1 & 0 & & \\ & & \ddots & \ddots & \\ & & & 1 & 0 \end{bmatrix}_{(n-1)\times n} \begin{bmatrix} 0 & & & \\ & 1 & -1 & & \\ & & 1 & -1 & \\ & & & \ddots & -1 \\ & & & & 1 \end{bmatrix} \begin{bmatrix} 2h_1 & & \\ & \ddots & \\ & & 2h_n \end{bmatrix} \begin{bmatrix} 3h_1^2 & & \\ & \ddots & \\ & & 3h_n^2 \end{bmatrix}_{n\times 4n} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}_{4n\times 1} = 0_{(n-1)\times 1}$$

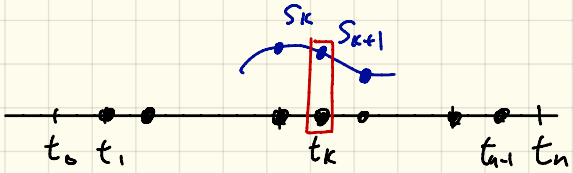
$$E \begin{bmatrix} 0 & J & 2H & 3H^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = 0$$

(3) Continuity of $S''(x)$ at interior points

$$S_k = a_k + b_k(x - t_{k-1}) + c_k(x - t_{k-1})^2 + d_k(x - t_{k-1})^3$$

$$\Rightarrow \begin{cases} S_k'' = 2c_k + 6d_k(x - t_{k-1}) \Big|_{x = t_k} \\ S_{k+1}'' = 2c_{k+1} + 6d_{k+1}(x - t_k) \Big|_{x = t_k} \end{cases}$$

$k = 1 : (n-1)$



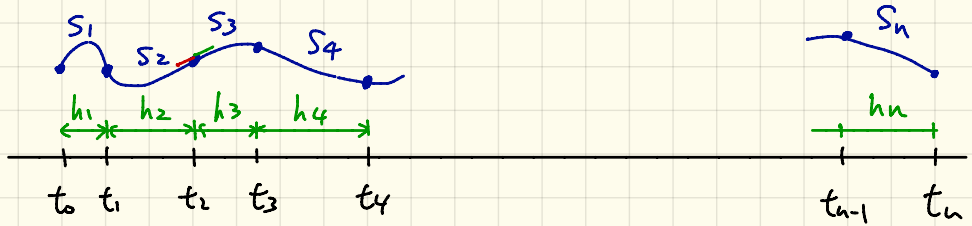
$$\Rightarrow \quad 2c_k + 6d_k h_k = 2c_{k+1}$$

$$\Rightarrow \quad (2c_k - 2c_{k+1}) + 6h_k \cdot d_k = 0$$

$$\Rightarrow \quad E \begin{bmatrix} 0 & 0 & J & 3H \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = 0$$

E: $(n-1) \times n$; $[0\ 0\ J\ 3H]$: $n \times 4n$; $\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$: $4n \times 1$; 0: $(n-1) \times 1$

(4) End constraints

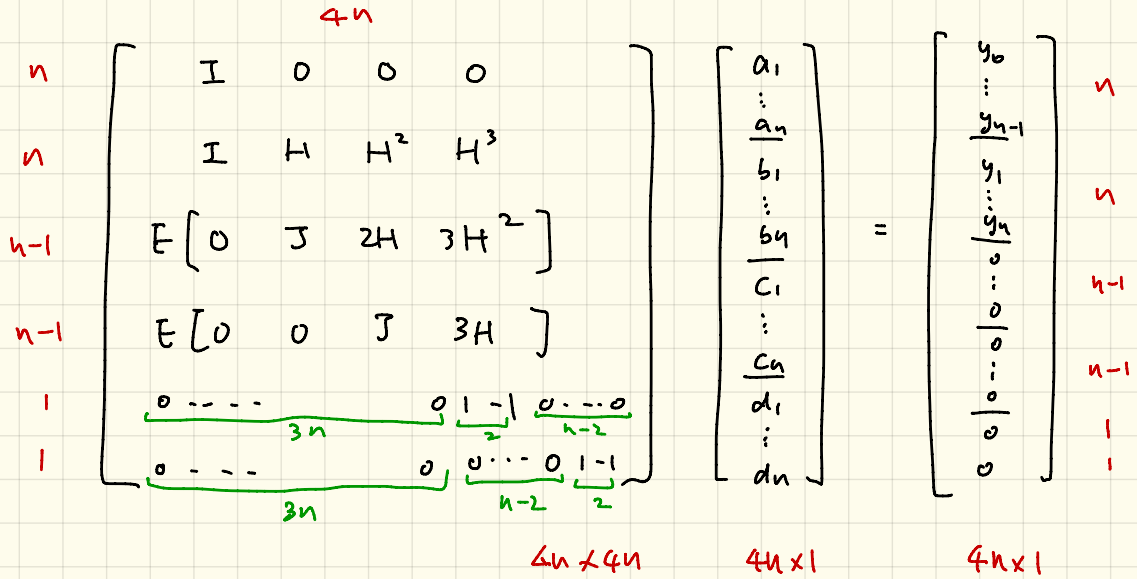


(a) Natural spline $S_1''(t_0) = S_n''(t_n) = 0$

important theoretical properties

(b) Not-a-Knot spline $\begin{cases} S_1'''(t_1) = S_2'''(t_1) \\ S_{n-1}'''(t_{n-1}) = S_n'''(t_{n-1}) \end{cases} \Rightarrow \begin{cases} d_1 = d_2 \\ d_{n-1} = d_n \end{cases} \Rightarrow \begin{cases} d_1 - d_2 = 0 \\ d_{n-1} - d_n = 0 \end{cases}$

Better pointwise approximation



Function 5.3.1

```
function S = spinterp(t,y)
% SPINTERP   Cubic not-a-knot spline interpolation.
% Input:
%   t     interpolation nodes (vector, length n+1)
%   y     interpolation values (vector, length n+1)
% Output:
%   S     not-a-knot cubic spline (function)

t = t(:);  y = y(:);  % ensure column vectors
n = length(t)-1;
h = diff(t);          % differences of all adjacent pairs

% Preliminary definitions.
Z = zeros(n);
I = eye(n);  E = I(1:n-1,:);
J = I - diag(ones(n-1,1),1);
H = diag(h);

% Left endpoint interpolation:
AL = [ I, Z, Z, Z ];
vL = y(1:n);

% Right endpoint interpolation:
AR = [ I, H, H^2, H^3 ];
vR = y(2:n+1);

% Continuity of first derivative:
A1 = E*[ Z, J, 2*H, 3*H^2 ];
v1 = zeros(n-1,1);

% Continuity of second derivative:
A2 = E*[ Z, Z, J, 3*H ];
v2 = zeros(n-1,1);

% Not-a-knot conditions:
nakL = [ zeros(1,3*n), [1,-1, zeros(1,n-2)] ];
nakR = [ zeros(1,3*n), [zeros(1,n-2), 1,-1] ];

% Assemble and solve the full system.
A = [ AL; AR; A1; A2; nakL; nakR ];
v = [ vL; vR; v1; v2; 0 ;0 ];
z = A\v;

% Break the coefficients into separate vectors.
rows = 1:n;
a = z(rows);
b = z(n+rows);  c = z(2*n+rows);  d = z(3*n+rows);
S = @evaulate;

    % This function evaluates the spline when called with a value for x.
    function f = evaulate(x)
        f = zeros(size(x));
        for k = 1:n       % iterate over the pieces
            % Evalaute this piece's cubic at the points inside it.
            index = (x>=t(k)) & (x<=t(k+1));
            f(index) = polyval( [d(k),c(k),b(k),a(k)], x(index)-t(k) );
        end
    end

end
```

Convergence : 4th order

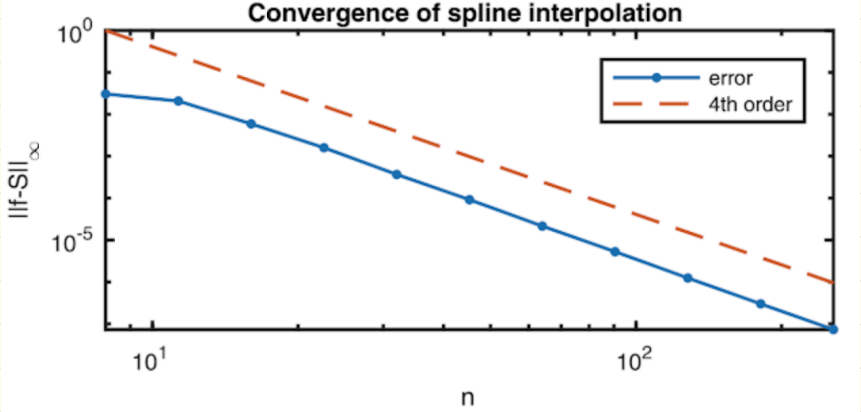
Theorem 5.3.1

Suppose that $f(x)$ has four continuous derivatives in $[a,b]$ (i.e., $f \in C^4[a,b]$). Let $S_n(x)$ be the not-a-knot cubic spline interpolant of $(t_i, f(t_i))$ for $i = 0,\ldots,n$, where $t_i = a + ih$ and $h = (b-a)/n$. Then for all sufficiently small h there is a constant $C > 0$ such that

$$\|f - S_n\|_\infty \le Ch^4. \tag{5.3.16}$$

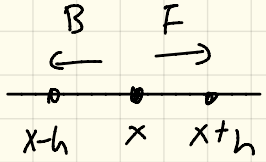
Example 5.3.1

```
clf, loglog( n_, err_, '.-' )
hold on, loglog( n_, (n_/n_(1)).^(-4), '--' )
```



- Plot the convergence figures in Ex. 5.2.3 & Ex. 5.3.1 in the same figure to compare the 2nd & 4th order convergence

§ 5.4 Finite Difference



Definition: $f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$

Forward difference formula: $f'(x) = \frac{f(x+h) - f(x)}{h} + O(h)$

$$f(x+h) = f(x) + h f'(x) + \frac{h^2}{2} f''(\xi)$$

Backward difference formula: $f'(x) = \frac{f(x) - f(x-h)}{h} + O(h)$

$$f(x-h) = f(x) - h f'(x) + \frac{h^2}{2} f''(\xi)$$

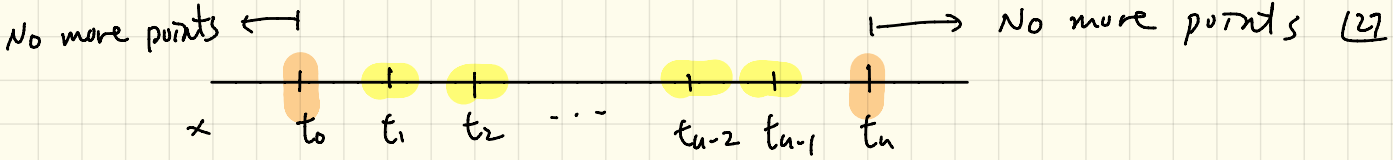
Centered finite difference:

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(\xi_1)$$

$$-) \quad f(x-h) = f(x) - hf'(x) + \frac{h^2}{2}f''(x) - \frac{h^3}{6}f'''(\xi_2)$$

$$f(x+h) - f(x-h) = 2hf'(x) + O(h^3)$$

$$\Rightarrow \quad f'(x) = \frac{f(x+h) - f(x-h)}{2h} + O(h^2)$$



- Consider n+1 equispaced points

$$x_i = x_0 + ih,\ i = 0, \ldots, n, \text{ with } h > 0$$

- How about the 2nd order centered finite difference for the two endpoints?
- Homework: Show that the following finite difference approximations are 2nd order.

$$\frac{1}{2h}\left[-3f(x_0) + 4f(x_1) - f(x_2)\right] \quad \text{at } x_0,$$

$$\frac{1}{2h}\left[3f(x_n) - 4f(x_{n-1}) + f(x_{n-2})\right] \text{ at } x_n,$$

Higher derivatives

$$f(x+h) = f(x) + h f'(x) + \frac{h^2}{2} f''(x) + \frac{h^3}{6} f'''(x) + O(h^4)$$

$$+) \quad f(x-h) = f(x) - h f'(x) + \frac{h^2}{2} f''(x) - \frac{h^3}{6} f'''(x) + O(h^4)$$

$$f(x+h) + f(x-h) = 2f(x) + h^2 f''(x) + O(h^4)$$

$$\Rightarrow \quad f''(x) = \frac{f(x-h) - 2f(x) + f(x+h)}{h^2} + O(h^4)$$

Order of Accuracy Ex. 5.5.4

We revisit Example 5.5.2 to compare the first-order forward difference formula with the second-order centered formula.

```
f = @(x) sin( exp(x+1) );
exact_value = cos(exp(1))*exp(1);
```

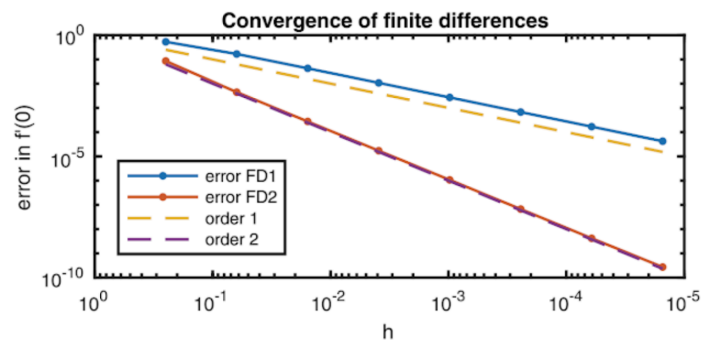
We run both formulas in parallel for a sequence of h values.

```
h = 4.^(-1:-1:-8)';
FD1 = 0*h;  FD2 = 0*h;
for k = 1:length(h)
    FD1(k) = (f(h(k)) - f(0)) / h(k);
    FD2(k) = (f(h(k)) - f(-h(k))) / (2*h(k));
end
```

In each case h is decreased by a factor of 4, so that the error is reduced by a factor of 4 in the first-order method and 16 in the second-order method.

```
error_FD1 = exact_value-FD1;
error_FD2 = exact_value-FD2;
table(h,error_FD1,error_FD2)
```

```
ans =
  8x3 table
        h          error_FD1       error_FD2
    __________    __________    ___________
         0.25       0.53167       -0.085897
       0.0625       0.16675       -0.0044438
     0.015625       0.042784      -0.00027403
    0.0039062       0.010751      -1.7112e-05
   0.00097656      0.0026911      -1.0695e-06
   0.00024414     0.00067298      -6.6841e-08
   6.1035e-05     0.00016826      -4.1767e-09
   1.5259e-05     4.2065e-05      -2.7269e-10
```



Conditioning

§ 5.6 Numerical Integration