Newton Divided Differences

Recursive Divided Differences, for all i, j (common nodes in blue)

$$f[x_i, x_{i+1}, \cdots, x_j, x_{j+1}] = \frac{f[x_{i+1}, \cdots, x_j, x_{j+1}] - f[x_i, x_{i+1}, \cdots, x_j]}{x_{j+1} - x_i},$$

Newton Divided Differences

Recursive Divided Differences, for all i, j (common nodes in blue)

$$f[x_i, x_{i+1}, \cdots, x_j, x_{j+1}] = \frac{f[x_{i+1}, \cdots, x_j, x_{j+1}] - f[x_i, x_{i+1}, \cdots, x_j]}{x_{j+1} - x_i},$$

then in

$$\begin{aligned} P(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \\ &\quad \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}), \end{aligned}$$

we have

$$\begin{aligned} a_0 &= f[x_0] \\ a_1 &= f[x_0, x_1] \\ a_2 &= f[x_0, x_1, x_2] \\ &\vdots \\ a_n &= f[x_0, x_1, \cdots, x_n] \end{aligned}$$

- computation flow

$$
\begin{array}{ccccccccc}
f[x_0] & & & & & & & & \\
f[x_1] & \uparrow & f[x_0, x_1] & & & & & & \\
f[x_2] & \uparrow & f[x_1, x_2] & \uparrow & f[x_0, x_1, x_2] & & & & \\
\vdots & & & & & & \ddots & & \\
f[x_{n-2}] & \uparrow & f[x_{n-3}, x_{n-2}] & \uparrow & f[x_{n-4}, x_{n-3}, x_{n-2}] & \uparrow & \cdots & & \\
f[x_{n-1}] & \uparrow & f[x_{n-2}, x_{n-1}] & \uparrow & f[x_{n-3}, x_{n-2}, x_{n-1}] & \uparrow & \cdots & & \\
f[x_n] & \uparrow & f[x_{n-1}, x_n] & \uparrow & f[x_{n-2}, x_{n-1}, x_n] & \uparrow & \cdots & & f[x_0, x_1, \cdots, x_n]
\end{array}
$$

- computation flow

$$
\begin{array}{ccccccccc}
f[x_0] & & & & & & & & \\
f[x_1] & \uparrow & f[x_0,x_1] & & & & & & \\
f[x_2] & \uparrow & f[x_1,x_2] & \uparrow & f[x_0,x_1,x_2] & & & & \\
\vdots & & & & & & \ddots & & \\
f[x_{n-2}] & \uparrow & f[x_{n-3},x_{n-2}] & \uparrow & f[x_{n-4},x_{n-3},x_{n-2}] & \uparrow & \cdots & & \\
f[x_{n-1}] & \uparrow & f[x_{n-2},x_{n-1}] & \uparrow & f[x_{n-3},x_{n-2},x_{n-1}] & \uparrow & \cdots & & \\
f[x_n] & \uparrow & f[x_{n-1},x_n] & \uparrow & f[x_{n-2},x_{n-1},x_n] & \uparrow & \cdots & & f[x_0,x_1,\cdots,x_n]
\end{array}
$$

- storage (with memory re-use): output is $F_0, F_1, \cdots, F_n$

$$
\begin{array}{ccccc}
\boxed{f[x_0] \overset{def}{=} F_0} & & & & \\
f[x_1] & \leftarrow & \boxed{f[x_0,x_1] \overset{def}{=} F_1} & & \\
f[x_2] & \leftarrow & f[x_1,x_2] & \leftarrow & \boxed{f[x_0,x_1,x_2] \overset{def}{=} F_2} \\
\vdots & & & & \ddots
\end{array}
$$

Interpolation via NDD: nested arithmetic

$$\begin{aligned} P(x) &= F_0 + F_1(x - x_0) + F_2 \cdot (x - x_0)(x - x_1) + \cdots \\ &\quad + F_n \cdot (x - x_0)(x - x_1) \cdots (x - x_{n-1}) \end{aligned}$$

Interpolation via NDD: nested arithmetic

$$
\begin{aligned}
P(x) &= F_0 + F_1(x - x_0) + F_2 \cdot (x - x_0)\,(x - x_1) + \cdots \\
&\quad + F_n \cdot (x - x_0)\,(x - x_1) \cdots (x - x_{n-1}) \\
&= F_0 + (x - x_0) \cdot \\
&\quad (F_1 + (x - x_1) \cdot (F_2 + \cdots (F_{n-1} + (x - x_{n-1}) \cdot F_n)))
\end{aligned}
$$

Interpolation via NDD: nested arithmetic

$$
\begin{aligned}
P(x) &= F_0 + F_1(x - x_0) + F_2 \cdot (x - x_0)(x - x_1) + \cdots \\
&\quad + F_n \cdot (x - x_0)(x - x_1) \cdots (x - x_{n-1}) \\
&= F_0 + (x - x_0) \cdot \\
&\quad (F_1 + (x - x_1) \cdot (F_2 + \cdots (F_{n-1} + (x - x_{n-1}) \cdot F_n)))
\end{aligned}
$$

- $f = F_n$

Interpolation via NDD: nested arithmetic

$$
\begin{aligned}
P(x) &= F_0 + F_1(x - x_0) + F_2 \cdot (x - x_0)\,(x - x_1) + \cdots \\
&\quad + F_n \cdot (x - x_0)\,(x - x_1) \cdots (x - x_{n-1}) \\
&= F_0 + (x - x_0) \cdot \\
&\quad (F_1 + (x - x_1) \cdot (F_2 + \cdots (F_{n-1} + (x - x_{n-1}) \cdot F_n)))
\end{aligned}
$$

- $f = F_n$
- **for** $i = n-1, \cdots, 1, 0$

$$f = F_i + (x - x_i) \cdot f.$$

Output $f = P(x)$.

Interpolation via NDD

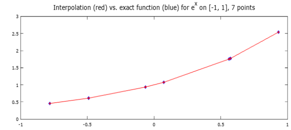
```
function f = EvaluateNDD(xnew,x,F)
%
% This function evaluates the interpolating polynomial given
% point xnew, nodes x, and NDF coefficients F
%
n = length(x);
m = length(xnew);
f = F(n)*ones(m,1);
for k=n-1:-1:1
   f = F(k) + f .* (xnew-x(k));
end
```

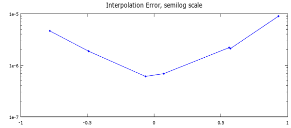
Polynomial Interpolation: Experiments:

- **Easy function**: $f(x) = e^x$ on $[-1, 1]$.

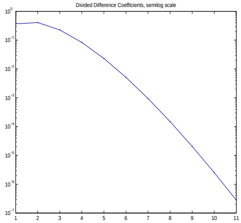
$$|f^{(n)}(\xi)| \le e \quad \text{for all} \quad \xi \in (-1, 1).$$

7 nodal points ($n = 6$)
random x points.





Divided difference coefficients for e^x on $[-1, 1]$



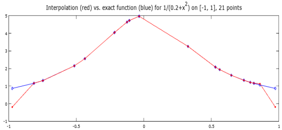
Polynomial Interpolation: Experiments:

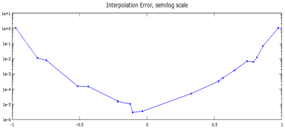
- **Hardest function**: $f(x) = \frac{1}{0.2+x^2}$ on $[-1, 1]$.

  $$|f^{(n)}(\xi)| \quad \text{can be very large for some} \quad \xi \in (-1, 1).$$

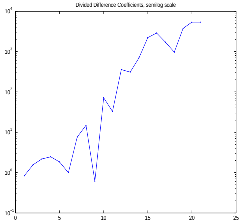
- random x points.

21 nodal points ($n = 20$)
random x points.

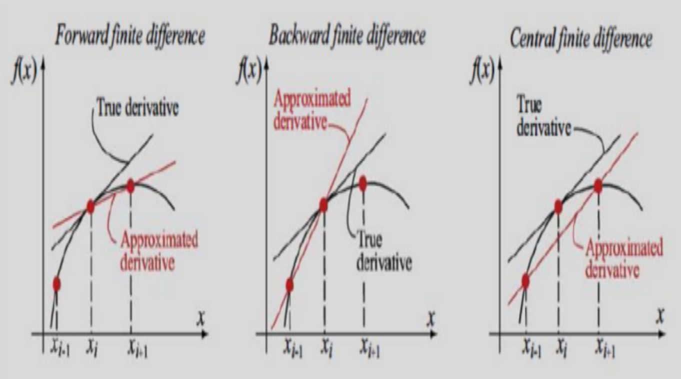




Divided difference coefficients for e^x on $[-1, 1]$



Finite Differences



Variation: Forward Differences

Let $x_1 = x_0 + h$, $x_2 = x_0 + 2h$

$$
\begin{aligned}
f[x_0, x_1] &= f[x_0, x_0 + h] = \frac{f(x_0 + h) - f(x_0)}{h} \quad \text{(first order FD)}, \\
f[x_0, x_1, x_2] &= f[x_0, x_0 + h, x_0 + 2h] = \frac{f[x_1, x_2] - f[x_0, x_1]}{2h} \\
&= \frac{f(x_2) + f(x_0) - 2f(x_1)}{2h^2} \quad \text{(second order FD)}
\end{aligned}
$$

Variation: Forward Differences

Let $x_1 = x_0 + h$, $x_2 = x_0 + 2h$

$$\begin{aligned}
f[x_0, x_1] &= f[x_0, x_0 + h] = \frac{f(x_0 + h) - f(x_0)}{h} \quad \text{(first order FD)}, \\
f[x_0, x_1, x_2] &= f[x_0, x_0 + h, x_0 + 2h] = \frac{f[x_1, x_2] - f[x_0, x_1]}{2h} \\
&= \frac{f(x_2) + f(x_0) - 2f(x_1)}{2h^2} \quad \text{(second order FD)}
\end{aligned}$$

```
>> x=0;h=0.1;x1=x+h;x2=x+2*h;
>> f1=(exp(x1)-exp(x))/h;
>> f2 = (exp(x2)+exp(x)-2*exp(x1))/(2*h^2);
>> f=exp(x);
>> [f f1 f2]
ans =

   1.00000   1.05171   0.55305
```

Variation: Backward Differences

Let $x_{n-1} = x_n - h$, $x_{n-2} = x_n - 2h$

$$
\begin{aligned}
f[x_{n-1}, x_n] &= f[x_n - h, x_n] = \frac{f(x_n) - f(x_n - h)}{h} \quad \text{(1st order BD)}, \\
f[x_{n-2}, x_{n-1}, x_n] &= f[x_n - 2h, x_n - h, x_n] \\
&= \frac{f[x_n - h, x_n] - f[x_n - 2h, x_n - h]}{2h} \\
&= \frac{f(x_n) + f(x_{n-2}) - 2f(x_{n-1})}{2h^2} \quad \text{(2nd order BD)}
\end{aligned}
$$

Variation: Backward Differences

Let $x_{n-1} = x_n - h$, $x_{n-2} = x_n - 2h$

$$
\begin{aligned}
f[x_{n-1}, x_n] &= f[x_n - h, x_n] = \frac{f(x_n) - f(x_n - h)}{h} \quad \text{(1st order BD)}, \\
f[x_{n-2}, x_{n-1}, x_n] &= f[x_n - 2h, x_n - h, x_n] \\
&= \frac{f[x_n - h, x_n] - f[x_n - 2h, x_n - h]}{2h} \\
&= \frac{f(x_n) + f(x_{n-2}) - 2f(x_{n-1})}{2h^2} \quad \text{(2nd order BD)}
\end{aligned}
$$

```
>> x=0;h=0.1;x1=x-h;x2=x-2*h;
>> f1=(exp(x)-exp(x1))/h;
>> f2 = (exp(x2)+exp(x)-2*exp(x1))/(2*h^2);
>> f=exp(x);
>> [f f1 f2]
ans =

   1.00000   0.95163   0.45280
```

§3.4 Double nodes: linear interpolation

- Given 2 distinct points

$$(x_0, f(x_0)), (x_1, f(x_1))$$

- Interpolating polynomial of degree ≤ 1

$$\begin{aligned} P(x) &= a_0 + a_1(x - x_0) \\ \text{with } P(x_0) &= f(x_0), \;\; P(x_1) = f(x_1), \end{aligned}$$

Where

$$a_0 = f(x_0), \quad a_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}.$$

§3.4 Double nodes: linear interpolation

- Given 2 distinct points

$$(x_0, f(x_0)), (x_1, f(x_1))$$

- Interpolating polynomial of degree ≤ 1

$$\begin{aligned} P(x) &= a_0 + a_1(x - x_0) \\ \text{with } P(x_0) &= f(x_0), \;\; P(x_1) = f(x_1), \end{aligned}$$

Where

$$a_0 = f(x_0), \quad a_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}.$$

Now let $x_1 \to x_0$, we obtain $a_1 \overset{def}{=} f[x_0, x_0] = f'(x_0)$.

$$P(x) = a_0 + a_1(x - x_0),$$

Interpolating polynomial now satisfies

$$P(x_0) = f(x_0), \;\; P'(x_0) = f'(x_0).$$

Double nodes: quadratic interpolation

- Given 3 distinct points

$$(x_0, f(x_0)), (x_1, f(x_1)), (x_2, f(x_2)),$$

- Interpolating polynomial of degree ≤ 2

$$P(x) \quad = \quad a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1),$$

where we have

$$a_0 = f[x_0], a_1 = f[x_0, x_1], a_2 = f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}.$$

Double nodes: quadratic interpolation

- Given 3 distinct points

$$(x_0, f(x_0)), (x_1, f(x_1)), (x_2, f(x_2)),$$

- Interpolating polynomial of degree ≤ 2

$$P(x) \quad = \quad a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1),$$

where we have

$$a_0 = f[x_0], a_1 = f[x_0, x_1], a_2 = f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0}.$$

Now let $x_2 \to x_1$, we obtain single node x_0, double node x_1:

$$\begin{aligned} f[x_1, x_1] &= f'(x_1), \\ a_2 &= f[x_0, x_1, x_1] \overset{def}{=} \frac{f[x_1, x_1] - f[x_0, x_1]}{x_1 - x_0} = \frac{f'(x_1) - f[x_0, x_1]}{x_1 - x_0}. \end{aligned}$$

Interpolating polynomial now satisfies

$$P(x_0) = f(x_0), \;\; P(x_1) = f(x_1) \;\; P'(x_1) = f'(x_1).$$

Double nodes twice: cubic interpolation (I)

- Given 4 distinct points

$$(x_0, f(x_0)), (x_1, f(x_1)), (x_2, f(x_2)), (x_3, f(x_3)),$$

- Interpolating polynomial of degree ≤ 3

$$P(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + a_3(x - x_0)(x - x_1)(x - x_2).$$

$$\text{where} \quad \begin{aligned} a_0 &= f[x_0], \\ a_1 &= f[x_0, x_1] \\ a_2 &= f[x_0, x_1, x_2] = \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} \\ a_3 &= f[x_0, x_1, x_2, x_3] = \frac{f[x_1, x_2, x_3] - f[x_0, x_1, x_2]}{x_3 - x_0}. \end{aligned}$$

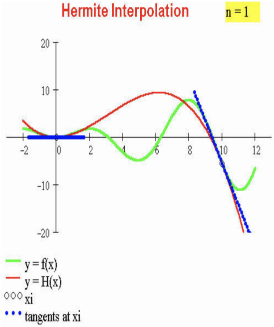
Double nodes twice: cubic interpolation (II)

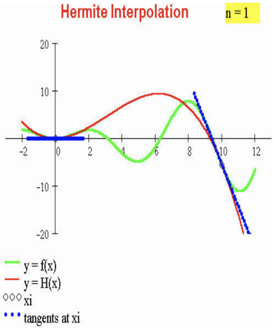
Let $x_1 \to x_0$, and $x_3 \to x_2$. It follows that

$$
\begin{aligned}
a_0 &= f[x_0] \\
a_1 &= f[x_0, x_0] = f'(x_0) \\
a_2 &= f[x_0, x_0, x_2] = \frac{f[x_0, x_2] - f[x_0, x_0]}{x_2 - x_0} = \frac{f[x_0, x_2] - f'(x_0)}{x_2 - x_0} \\
f[x_0, x_2, x_2] &= \frac{f[x_2, x_2] - f[x_2, x_0]}{x_2 - x_0} = \frac{f'(x_2) - f[x_0, x_2]}{x_2 - x_0} \\
a_3 &= f[x_0, x_0, x_2, x_2] = \frac{f[x_0, x_2, x_2] - f[x_0, x_0, x_2]}{x_2 - x_0}.
\end{aligned}
$$

This is a Hermite interpolation:

$$
\begin{aligned}
P(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + a_3(x - x_0)^2(x - x_2) \\
P(x_0) &= f(x_0), \;\; P'(x_0) = f'(x_0), \;\; P(x_2) = f(x_2), \;\; P'(x_2) = f'(x_2).
\end{aligned}
$$


Hermite Interpolation
n = 1
20
10
-10
-20
-2
0
2
4
6
8
10
12
y = f(x)
y = H(x)
xi
tangents at xi



- Given $n+1$ distinct points

$$(x_0, f(x_0), f'(x_0)), (x_1, f(x_1), f'(x_1)), \cdots, (x_n, f(x_n),$$

- Interpolating polynomial $H(x)$ of degree $\leq 2n+1$ with

$$\begin{aligned} H(x_0) &= f(x_0), \quad H'(x_0) = f'(x_0), \\ H(x_1) &= f(x_1), \quad H'(x_1) = f'(x_1), \\ &\vdots \\ H(x_n) &= f(x_n), \quad H'(x_n) = f'(x_n). \end{aligned}$$

- $2n+2$ conditions, $2n+2$ coefficients in $H(x)$.

Divided differences: double node version

- Given $m+1$ distinct points $(m = 2\,n-1)$

$$(z_0, f(z_0)), (z_1, f(z_1)), \cdots, (z_m, f(z_m)),$$

- Interpolating polynomial of degree $\leq m = 2\,n-1$

$$\begin{aligned} \text{with}\;\; P(x) &= a_0 + a_1(x-z_0) + a_2(x-z_0)(x-z_1) + \\ &\quad \cdots + a_m(x-z_0)(x-z_1)\cdots(x-z_{m-1}), \\ \text{satisfying}\;\; P(z_0) &= f(z_0), \;\; P(z_1) = f(z_1), \;\; \cdots, \;\; P(z_m) = f(z_m). \end{aligned}$$

- Coefficients satisfy

$$a_j = f[z_0, z_1, \cdots, z_j], \quad \text{for} \quad j = 0, 1, \cdots, m = 2\,n-1.$$

- Make each node a double node:

$$z_j \longrightarrow x_{\lfloor j/2 \rfloor} \quad \text{for} \quad j = 0, 1, \cdots, 2\,n-1.$$

Review: Newton Divided Difference Table

z_i	$f[z_i]$	$f[z_{i-1}, z_i]$	$f[z_{i-2}, z_{i-1}, z_i]$	$\cdots$	$f[z_0, z_1, \cdots, z_7]$
z_0	$\mathbf{f[z_0] \overset{def}{=} a_0}$				
		$\mathbf{f[z_0, z_1] \overset{def}{=} a_1}$			
z_1	$f[z_1]$		$\mathbf{f[z_0, z_1, z_2] \overset{def}{=} a_2}$		
		$f[z_1, z_2]$			
z_2	$f[z_2]$		$f[z_1, z_2, z_3]$		
		$f[z_2, z_3]$			
z_3	$f[z_3]$		$f[z_2, z_3, z_4]$		
		$f[z_3, z_4]$			
z_4	$f[z_4]$		$f[z_3, z_4, z_5]$	$\cdots$	$\mathbf{f[z_0, z_1, \cdots, z_7] \overset{def}{=} a_7}$
		$f[z_4, z_5]$			
z_5	$f[z_5]$		$f[z_4, z_5, z_6]$		
		$f[z_5, z_6]$			
z_6	$f[z_6]$		$f[z_5, z_6, z_7]$		
		$f[z_6, z_7]$			
z_7	$f[z_7]$				

Double nodes, $m = 2n+1$, $x_j = z_{\lfloor j/2 \rfloor}$, $f[x_j, x_j] = f'(x_j)$

x_i	$f[x_i]$	$f[z_{i-1}, z_i]$	$f[z_{i-2}, z_{i-1}, z_i]$	$\cdots$	$f[z_0, z_1, \cdots, z_7]$
x_0	$\mathbf{f[x_0] \overset{def}{=} a_0}$				
		$\underline{\mathbf{f[x_0, x_0] \overset{def}{=} f'(x_0)}} \overset{\mathbf{def}}{=} \mathbf{a_1}$			
x_0	$f[x_0]$		$\mathbf{f[x_0, x_0, x_1] \overset{def}{=} a_2}$		
		$f[x_0, x_1]$			
x_1	$f[x_1]$		$f[x_0, x_1, x_1]$		
		$\underline{f[x_1, x_1] \overset{def}{=} f'(x_1)}$			
x_1	$f[x_1]$		$f[x_1, x_1, x_2]$		
		$f[x_1, x_2]$			
x_2	$f[x_2]$		$f[x_1, x_2, x_2]$	$\cdots$	$\mathbf{f[x_0, x_0, \cdots, x_3]}$
		$\underline{f[x_2, x_2] \overset{def}{=} f'(x_2)}$			
x_2	$f[x_2]$		$f[x_2, x_2, x_3]$		
		$f[x_2, x_3]$			
x_3	$f[x_3]$		$f[x_2, x_3, x_3]$		
		$\underline{f[x_3, x_3] \overset{def}{=} f'(x_3)}$			
x_3	$f[x_3]$				

Newton Divided Differences for Hermite Interpolation

```
function F = NDD2(x,f,df)
%
% This function implements Newton's Divided Difference Algorithm
% for Hermite Interpolation. f is the vector of function values
% and df vector of derivatives.
%
% Updated by Ming Gu for Math 128A, Spring 2015
%
N = length(x);
x = x(:);
xx = reshape(repmat(x',2,1),2*N,1);
f = f(:);
df= df(:);
F  = reshape(repmat(f',2,1),2*N,1);
NN = N * 2;
F(2*(1:N))      = df;
F(1+2*(1:N-1)) = (f(2:N)-f(1:N-1))./(x(2:N)-x(1:N-1));
for k=3:2*N
    for j = 2*N:-1:k
       F(j) = (F(j)-F(j-1))/(xx(j)-xx(j-k+1));
    end
end
```

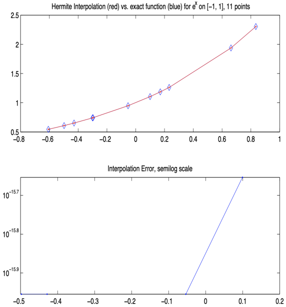
Newton Divided Differences for Hermite Interpolation

```
function F = NDD2(x,f,df)
%
% This function implements Newton's Divided Difference Algorithm
% for Hermite Interpolation. f is the vector of function values
% and df vector of derivatives.
%
% Updated by Ming Gu for Math 128A, Spring 2015
%
N = length(x);
x = x(:);
xx = reshape(repmat(x',2,1),2*N,1);
f = f(:);
df= df(:);
F  = reshape(repmat(f',2,1),2*N,1);
NN = N * 2;
F(2*(1:N))      = df;
F(1+2*(1:N-1)) = (f(2:N)-f(1:N-1))./(x(2:N)-x(1:N-1));
for k=3:2*N
    for j = 2*N:-1:k
        F(j) = (F(j)-F(j-1))/(xx(j)-xx(j-k+1));
    end
end
```

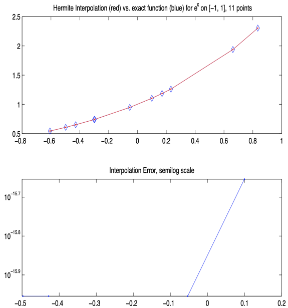
Evaluate Hermite Polynomial given coefficients

```
function f = EvaluateNDD2(xnew,x,F)
%
% This function evaluates the Hermite interpolating polynomial given
% point xnew, nodes x, and NDF coefficients F
%
n = length(x);
m = length(xnew);
f = F(2*n)*ones(m,1);
z = kron(x(:),ones(2,1));
for k=2*n-1:-1:1
   f = F(k) + f .* (xnew-z(k));
end
```

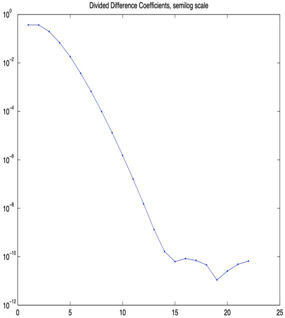
Hermite Interpolation on e^x on $[-1, 1]$



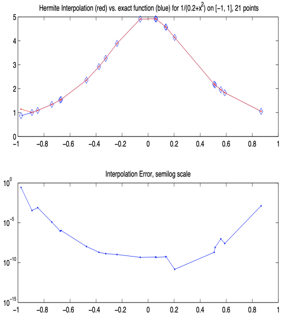
Hermite Interpolation on e^x on $[-1, 1]$



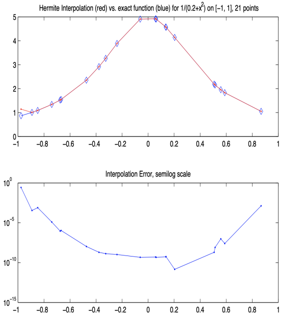
Coefficients a_j for Hermite Interpolation



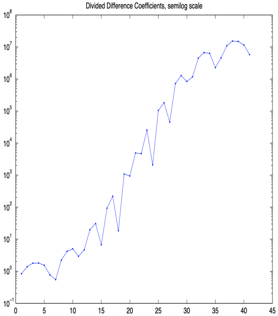
Hermite Interpolation on $\frac{1}{0.2+x^2}$ on $[-1, 1]$



Hermite Interpolation on $\frac{1}{0.2+x^2}$ on $[-1,1]$



Coefficients a_j for Hermite Interpolation



Hermite Interpolation, Alternative form

- Given $n+1$ distinct points

$$(x_0, f(x_0), f'(x_0)), (x_1, f(x_1), f'(x_1)), \cdots, (x_n, f(x_n), f'(x_n)),$$

- Interpolating polynomial $H(x)$ of degree $\leq 2n+1$ with

$$\begin{array}{rcl} H(x_0) & = & f(x_0), \quad H'(x_0) = f'(x_0), \\ H(x_1) & = & f(x_1), \quad H'(x_1) = f'(x_1), \\ \vdots & & \vdots \\ H(x_n) & = & f(x_n), \quad H'(x_n) = f'(x_n). \end{array}$$

- Alternative $H(x)$ form

$$H(x) = \sum_{j=0}^{n} f(x_j)\, H_j(x) + \sum_{j=0}^{n} f'(x_j)\, \widehat{H}_j(x), \quad \text{where}$$

$$H_j(x) = \left(1 - 2(x - x_j)L_j'(x_j)\right) L_j^2(x), \quad \widehat{H}_j(x) = (x - x_j)L_j^2(x),$$

with $L_j(x) = \prod_{i \neq j} \dfrac{x - x_i}{x_j - x_i}$.

Hermite Interpolation Error

Theorem: Suppose $x_0, x_1, \cdots, x_n$ are distinct numbers in the interval $[a, b]$ and $f \in C^{2n+2}[a, b]$. Then, for each $x \in [a, b]$, a number $\xi(x)$ between $, x_0, x_1, \cdots, x_n$ (hence $\in (a, b)$) exists with

$$f(x) = H(x) + \frac{f^{(2n+2)}(\xi(x))}{(2n+2)!}(x - x_0)^2(x - x_1)^2 \cdots (x - x_n)^2,$$

where $H(x)$ is the interpolating polynomial.

Hermite Interpolation Error: Proof

If $x = x_0, x_1, \cdots, \ x_n$, then error $= 0$ and theorem is true. Now let x be not equal to any node. Define function g for $t \in [a, b]$

$$\begin{aligned} g(t) &\stackrel{def}{=} (f(t) - H(t)) - (f(x) - H(x)) \frac{(t-x_0)^2(t-x_1)^2\cdots(t-x_n)^2}{(x-x_0)^2(x-x_1)^2\cdots(x-x_n)^2} \\ &= (f(t) - H(t)) - (f(x) - H(x)) \prod_{j=0}^{n} \frac{(t-x_j)^2}{(x-x_j)^2} \in C^{2n+2}[a, b]. \end{aligned}$$

Then $g(t)$ vanishes at $n+2$ distinct points:

$$g(x) = 0, \quad g(x_k) = 0, \quad \text{, for} \quad k = 0, 1, \cdots, n.$$

and $g'(t)$ vanishes at $n+1$ distinct points:

$$g'(x_k) = 0, \quad \text{, for} \quad k = 0, 1, \cdots, n.$$

There must be a ξ between x and nodal points such that

$$g^{(2n+2)}(\xi) = 0.$$

Hermite Interpolation Error: Proof

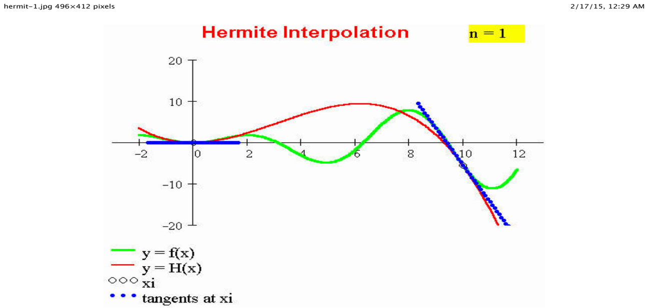
Since

$$
\begin{aligned}
g^{(2n+2)}(\xi) &= (f(t)-H(t))^{(2n+2)}\,|_{t=\xi} - (f(x)-H(x))\Big(\prod_{j=0}^{n}\frac{(t-x_j)^2}{(x-x_j)^2}\Big)^{(2n+2)}\,|_{\xi} \\
&= f^{(2n+2)}(\xi) - (f(x)-H(x))\,\frac{(2n+2)!}{\prod_{j=0}^{n}(x-x_j)^2} \\
&= 0
\end{aligned}
$$

Therefore

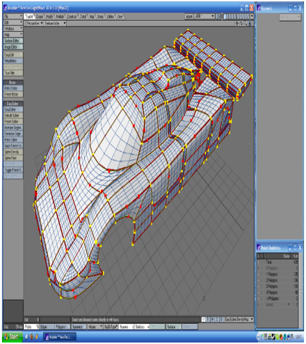
$$
f(x) = H(x) + \frac{f^{(2n+2)}(\xi(x))}{(2n+2)!}\,(x-x_0)^2(x-x_1)^2\cdots(x-x_n)^2,
$$

Hermite interpolation not good enough

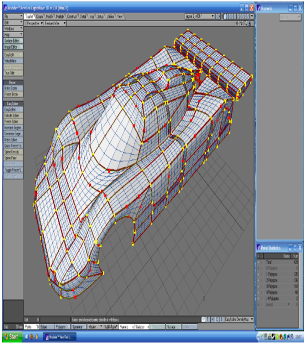


- For small n: some approximation, but error not small enough
- For large n: may not be any approximation

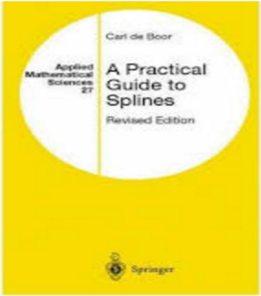
§3.5 A Car Wrapped in Splines



§3.5 A Car Wrapped in Splines



Carl de Boor: The book about Splines

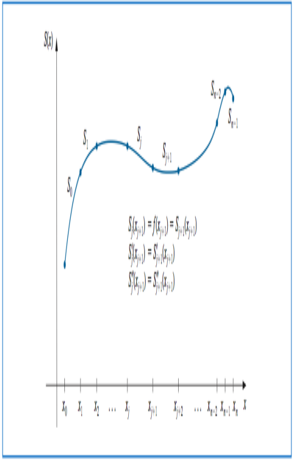


Carl de Boor

Carl-Wilhelm Reinhold de Boor

Born	3 December 1937 (age 79) Stolp, Germany (present-day Słupsk, Poland)
Fields	Mathematics (Numerical analysis)
Institutions	Purdue University University of Wisconsin–Madison University of Washington
Alma mater	University of Michigan
Notable awards	John von Neumann Prize (1996) National Medal of Science (2003)

The Splines Idea



Given $n+1$ distinct points

$$(x_0, f(x_0)), (x_1, f(x_1)), \cdots, (x_n, f(x_n)),$$

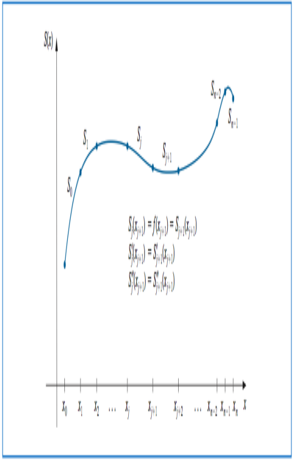
Find *cubic spline interpolant* $S(x)$:

- for $x \in [x_j, x_{j+1}]$, $j = 0, 1, \cdots, \underline{n-1}$,

$$S(x) = S_j(x) \stackrel{def}{=} a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3.$$

($S(x)$ piece-wise cubic: $4\,n$ unknowns)

The Splines Idea



Given $n+1$ distinct points

$$(x_0, f(x_0)), (x_1, f(x_1)), \cdots, (x_n, f(x_n)),$$

Find *cubic spline interpolant* $S(x)$:

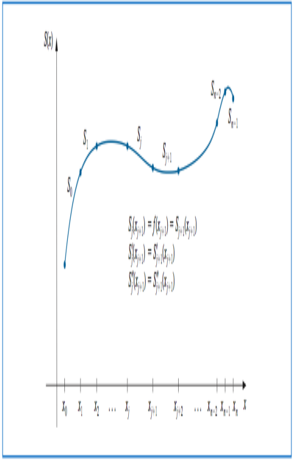
- for $x \in [x_j, x_{j+1}]$, $j = 0, 1, \cdots, \underline{n-1}$,

  $$S(x) = S_j(x) \stackrel{def}{=} a_j + b_j(x-x_j) + c_j(x-x_j)^2 + d_j(x-x_j)^3.$$

  ($S(x)$ piece-wise cubic: $4\,n$ unknowns)

- $S(x_j) = f(x_j)$ $j = 0, 1, \cdots, \underline{n}$.
  ($S(x) = f(x)$ all nodes: $n+1$ conditions)

The Splines Idea



Given $n+1$ distinct points

$$(x_0, f(x_0)), (x_1, f(x_1)), \cdots, (x_n, f(x_n)),$$

Find *cubic spline interpolant* $S(x)$:

- for $x \in [x_j, x_{j+1}]$, $j = 0, 1, \cdots, \underline{n-1}$,

  $$S(x) = S_j(x) \stackrel{def}{=} a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3.$$

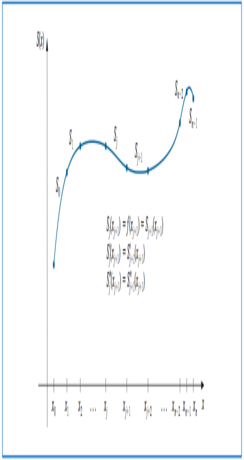
  ($S(x)$ piece-wise cubic: $4\,n$ unknowns)

- $S(x_j) = f(x_j)$ $j = 0, 1, \cdots, \underline{n}$.
  ($S(x) = f(x)$ all nodes: $n+1$ conditions)

- $S(x) \in C^2[x_0, x_n]$ (smooth-enough:
  $3\,(n-1)$ conditions)

- $4\,n$ unknowns vs. $4\,n-2$ conditions so far.

The Splines Equations (I)

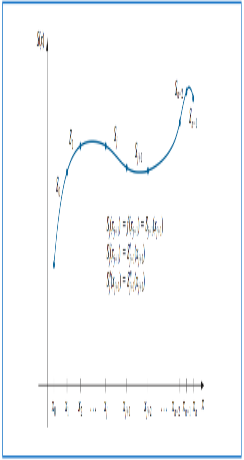


For $x \in [x_j, x_{j+1}],\ \ j = 0, 1, \cdots, n-1,$

$$S(x) = S_j(x) \overset{def}{=} a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$$

- for $j = 0, 1, \cdots, n-1 : a_j = S_j(x_j) = f(x_j)$

The Splines Equations (I)



For $x \in [x_j, x_{j+1}]$, $\;j = 0, 1, \cdots, n-1$,

$$S(x) = S_j(x) \overset{def}{=} a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3$$

- for $j = 0, 1, \cdots, n-1$: $a_j = S_j(x_j) = f(x_j)$
- let $a_n = f(x_n)$, and let $h_j = x_{j+1} - x_j$.
- for $j = 0, 1, \cdots, n-1$,

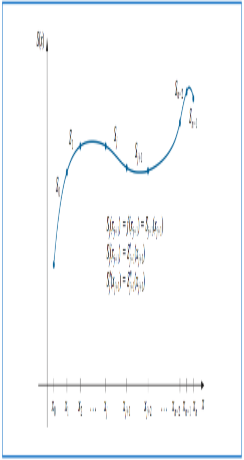
$$a_{j+1} = S_{j+1}(x_{j+1}) = S_j(x_{j+1}) = a_j + b_j h_j + c_j h_j^2 + d_j h_j^3$$

$$b_j + c_j h_j + d_j h_j^2 = \frac{a_{j+1} - a_j}{h_j}. \qquad (\ell_{\underline{0}})$$

$S(x) \in C^{\underline{0}}[x_0, x_n]$ (3 n unknowns, n conditions)

and $S(x_j) = f(x_j)$, for $j = 0, 1, \cdots, n-1, n$.

The Splines Equations (II)



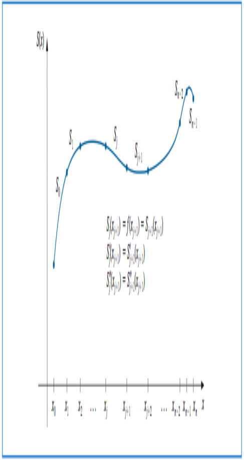
For $x \in [x_j, x_{j+1}], \;\; j = 0, 1, \cdots, n-1,$

$$\begin{aligned} S_j(x) &= a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \\ \Rightarrow \; S_j'(x) &= b_j + 2c_j(x - x_j) + 3d_j(x - x_j)^2 \end{aligned}$$

- Define $b_n \overset{def}{=} S_{n-1}'(x_n)$ (**artificial** new unknown)

The Splines Equations (II)

$$\text{For } x \in [x_j, x_{j+1}],\ \ j = 0, 1, \cdots, n-1,$$



$$\begin{aligned} S_j(x) &= a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \\ \Rightarrow\ S_j'(x) &= b_j + 2c_j(x - x_j) + 3d_j(x - x_j)^2 \end{aligned}$$

- Define $b_n \stackrel{def}{=} S_{n-1}'(x_n)$ (**artificial** new unknown)
- For $j = 0, \cdots, n-1$,

  $$b_{j+1} = S_{j+1}'(x_{j+1}) = S_j'(x_{j+1}) = b_j + 2c_j h_j + 3d_j h_j^2 \ (\ell_{\underline{1}})$$

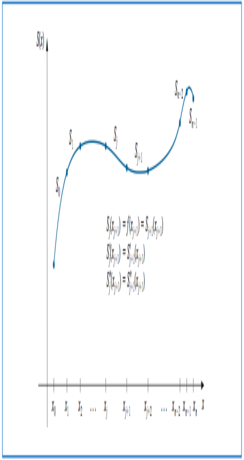
- Ensures $S(x) \in C^{\underline{1}}[x_0, x_n]$ (with $3\,n + 1$ unknowns, n additional conditions)

$$S_j'(x_{j+1}) = S_{j+1}'(x_{j+1}), \text{ for } j = 0, 1, \cdots, n-1.$$

The Splines Equations (III)

For $x \in [x_j, x_{j+1}], \;\; j = 0, 1, \cdots, n-1,$

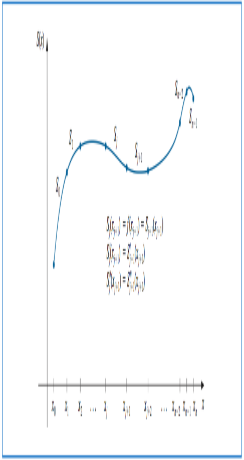
$$
\begin{aligned}
S_j(x) &= a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \\
S_j''(x) &= 2c_j + 6d_j(x - x_j)
\end{aligned}
$$



- Define $c_n = S_{n-1}''(x_n)/2$ (second **artificial** new unknown)

The Splines Equations (III)

For $x \in [x_j, x_{j+1}], \;\; j = 0, 1, \cdots, n-1,$



$$\begin{aligned} S_j(x) &= a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \\ S_j''(x) &= 2c_j + 6d_j(x - x_j) \end{aligned}$$

- Define $c_n = S''_{n-1}(x_n)/2$ (second **artificial** new unknown)
- For $j = 0, \cdots, n-1,$

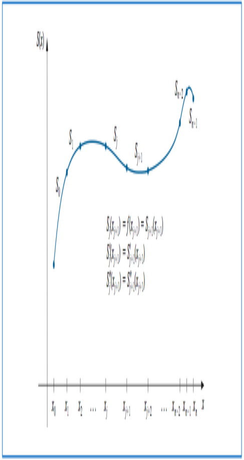
$$2c_{j+1} = S''_{j+1}(x_{j+1}) = S''_j(x_{j+1}) = 2c_j + 6d_j h_j. \qquad (\ell_{\underline{2}})$$

- Ensures $S(x) \in C^{\underline{2}}[x_0, x_n]$ (with $3\,n + 2$ unknowns, $3\,n$ conditions)

The Splines Equations (III)

For $x \in [x_j, x_{j+1}]$, $j = 0, 1, \cdots, n-1$,

$$\begin{aligned} S_j(x) &= a_j + b_j(x-x_j) + c_j(x-x_j)^2 + d_j(x-x_j)^3 \\ S_j''(x) &= 2c_j + 6d_j(x-x_j) \end{aligned}$$



- Define $c_n = S_{n-1}''(x_n)/2$ (second **artificial** new unknown)
- For $j = 0, \cdots, n-1$,

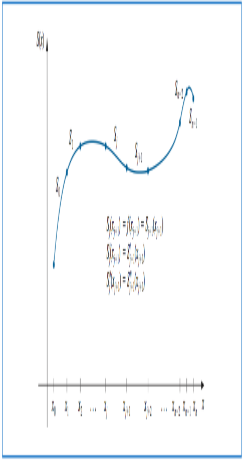
$$2c_{j+1} = S_{j+1}''(x_{j+1}) = S_j''(x_{j+1}) = 2c_j + 6d_j h_j. \qquad (\ell_{\underline{2}})$$

- Ensures $S(x) \in C^{\underline{2}}[x_0, x_n]$ (with $3\,n+2$ unknowns, $3\,n$ conditions)

$$\begin{aligned} b_j + c_j h_j + d_j h_j^2 &= \frac{a_{j+1} - a_j}{h_j} && (\ell_{\underline{0}}) \\ b_{j+1} &= b_j + 2c_j h_j + 3d_j h_j^2 && (\ell_{\underline{1}}) \\ 2c_{j+1} &= 2c_j + 6d_j h_j. && (\ell_{\underline{2}}) \end{aligned}$$

The Splines Equations (III)

For $x \in [x_j, x_{j+1}], \;\; j = 0, 1, \cdots, n-1,$



$$\begin{aligned} S_j(x) &= a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3 \\ S_j''(x) &= 2c_j + 6d_j(x - x_j) \end{aligned}$$

- Define $c_n = S_{n-1}''(x_n)/2$ (second **artificial** new unknown)
- For $j = 0, \cdots, n-1,$

$$2c_{j+1} = S_{j+1}''(x_{j+1}) = S_j''(x_{j+1}) = 2c_j + 6d_j h_j. \qquad (\ell_{\underline{2}})$$

- Ensures $S(x) \in C^{2}[x_0, x_n]$ (with $3\,n+2$ unknowns, $3\,n$ conditions)

$$\begin{aligned} b_j + c_j h_j + d_j h_j^2 &= \frac{a_{j+1} - a_j}{h_j} && (\ell_{\underline{0}}) \\ b_{j+1} &= b_j + 2c_j h_j + 3d_j h_j^2 && (\ell_{\underline{1}}) \\ 2c_{j+1} &= 2c_j + 6d_j h_j. && (\ell_{\underline{2}}) \end{aligned}$$

$$\boxed{\text{Next: solve for } d_j \text{ in } (\ell_{\underline{2}}) \text{ and } b_j \text{ in } (\ell_{\underline{0}})}$$

The Splines Equations (IV): Solving for d_j and b_j

$$
\begin{aligned}
b_j + c_j h_j + d_j h_j^2 &= \frac{a_{j+1} - a_j}{h_j}, \quad j = 0, \cdots, n-1, \quad (\ell_{\underline{0}}) \\
b_{j+1} &= b_j + 2c_j h_j + 3d_j h_j^2 \qquad (\ell_{\underline{1}}) \\
2c_{j+1} &= 2c_j + 6d_j h_j. \qquad (\ell_{\underline{2}})
\end{aligned}
$$

$\boxed{\text{Solve for } d_j}$ $(\ell_{\underline{2}})$: $\qquad 2c_{j+1} = 2c_j + 6d_j h_j, \quad \Rightarrow \quad d_j = \dfrac{c_{j+1} - c_j}{3h_j} \quad (\widehat{\ell_2})$

The Splines Equations (IV): Solving for d_j and b_j

$$\begin{aligned} b_j + c_j h_j + d_j h_j^2 &= \frac{a_{j+1} - a_j}{h_j}, \quad j = 0, \cdots, n-1, \quad (\ell_{\underline{0}}) \\ b_{j+1} &= b_j + 2c_j h_j + 3d_j h_j^2 \qquad (\ell_{\underline{1}}) \\ 2c_{j+1} &= 2c_j + 6d_j h_j. \qquad (\ell_{\underline{2}}) \end{aligned}$$

$\boxed{\text{Solve for } d_j}$ $(\ell_{\underline{2}})$:
$$2c_{j+1} = 2c_j + 6d_j h_j, \quad \Rightarrow \quad d_j = \frac{c_{j+1} - c_j}{3h_j} \quad (\widehat{\ell_2})$$

$\boxed{\text{Solve for } b_j :}$
$$b_j + c_j h_j + d_j h_j^2 = \frac{a_{j+1} - a_j}{h_j}, \xRightarrow{(\widehat{\ell_2})} b_j = -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1} - a_j}{h_j} \, (\widehat{\ell_0})$$

The Splines Equations (IV): Solving for d_j and b_j

$$\begin{aligned} b_j + c_j h_j + d_j h_j^2 &= \frac{a_{j+1} - a_j}{h_j}, \quad j = 0, \cdots, n-1, \quad (\ell_{\underline{0}}) \\ b_{j+1} &= b_j + 2c_j h_j + 3d_j h_j^2 \qquad (\ell_{\underline{1}}) \\ 2c_{j+1} &= 2c_j + 6d_j h_j. \qquad (\ell_{\underline{2}}) \end{aligned}$$

$\boxed{\text{Solve for } d_j}$ $(\ell_{\underline{2}})$: $\quad 2c_{j+1} = 2c_j + 6d_j h_j, \quad \Rightarrow \quad d_j = \dfrac{c_{j+1} - c_j}{3h_j} \quad (\widehat{\ell_2})$

$\boxed{\text{Solve for } b_j :}$ $b_j + c_j h_j + d_j h_j^2 = \dfrac{a_{j+1} - a_j}{h_j}, \stackrel{(\widehat{\ell_2})}{\Longrightarrow} b_j = -\dfrac{h_j}{3}(2c_j + c_{j+1}) + \dfrac{a_{j+1} - a_j}{h_j} \; (\widehat{\ell_0})$

$\boxed{\text{Get rid of } d_j \text{ in } (\ell_{\underline{1}}) :}$ $b_{j+1} = b_j + 2c_j h_j + 3d_j h_j^2, \stackrel{(\widehat{\ell_2})}{\Longrightarrow} b_{j+1} = b_j + h_j(c_j + c_{j+1}) \; (\widehat{\ell_1})$

The Splines Equations (IV): Solving for d_j and b_j

$$\begin{aligned} b_j + c_j h_j + d_j h_j^2 &= \frac{a_{j+1} - a_j}{h_j}, \quad j = 0, \cdots, n-1, \quad (\ell_{\underline{0}}) \\ b_{j+1} &= b_j + 2c_j h_j + 3d_j h_j^2 \qquad (\ell_{\underline{1}}) \\ 2c_{j+1} &= 2c_j + 6d_j h_j. \qquad (\ell_{\underline{2}}) \end{aligned}$$

$$\boxed{\text{Solve for } d_j} \quad (\ell_{\underline{2}}) : \qquad 2c_{j+1} = 2c_j + 6d_j h_j, \quad \Rightarrow \quad d_j = \frac{c_{j+1} - c_j}{3h_j} \quad (\widehat{\ell_2})$$

$$\boxed{\text{Solve for } b_j :} \; b_j + c_j h_j + d_j h_j^2 = \frac{a_{j+1} - a_j}{h_j}, \; \stackrel{(\widehat{\ell_2})}{\Longrightarrow} \; b_j = -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1} - a_j}{h_j} \; (\widehat{\ell_0})$$

$$\boxed{\text{Get rid of } d_j \text{ in } (\ell_{\underline{1}}) :} \; b_{j+1} = b_j + 2c_j h_j + 3d_j h_j^2, \; \stackrel{(\widehat{\ell_2})}{\Longrightarrow} \; b_{j+1} = b_j + h_j(c_j + c_{j+1}) \; (\widehat{\ell_1})$$

$$\boxed{(\widehat{\ell_1})} \quad -\frac{h_{j+1}}{3}(2c_{j+1} + c_{j+2}) + \frac{a_{j+2} - a_{j+1}}{h_{j+1}} = -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1} - a_j}{h_j} + h_j(c_j + c_{j+1})$$

The Splines Equations (IV): Solving for d_j and b_j

$$\begin{aligned} b_j + c_j h_j + d_j h_j^2 &= \frac{a_{j+1} - a_j}{h_j}, \quad j = 0, \cdots, n-1, \quad (\ell_{\underline{0}}) \\ b_{j+1} &= b_j + 2c_j h_j + 3d_j h_j^2 \qquad (\ell_{\underline{1}}) \\ 2c_{j+1} &= 2c_j + 6d_j h_j. \qquad (\ell_{\underline{2}}) \end{aligned}$$

$\boxed{\text{Solve for } d_j}$ $(\ell_{\underline{2}})$: $\quad 2c_{j+1} = 2c_j + 6d_j h_j, \quad \Rightarrow \quad d_j = \dfrac{c_{j+1} - c_j}{3h_j} \quad (\widehat{\ell_2})$

$\boxed{\text{Solve for } b_j :}$ $b_j + c_j h_j + d_j h_j^2 = \dfrac{a_{j+1} - a_j}{h_j}, \overset{(\widehat{\ell_2})}{\Longrightarrow} b_j = -\dfrac{h_j}{3}(2c_j + c_{j+1}) + \dfrac{a_{j+1} - a_j}{h_j} \; (\widehat{\ell_0})$

$\boxed{\text{Get rid of } d_j \text{ in } (\ell_{\underline{1}}) :}$ $b_{j+1} = b_j + 2c_j h_j + 3d_j h_j^2, \overset{(\widehat{\ell_2})}{\Longrightarrow} b_{j+1} = b_j + h_j(c_j + c_{j+1}) \; (\widehat{\ell_1})$

$\boxed{(\widehat{\ell_1})}$ $-\dfrac{h_{j+1}}{3}(2c_{j+1} + c_{j+2}) + \dfrac{a_{j+2} - a_{j+1}}{h_{j+1}} = -\dfrac{h_j}{3}(2c_j + c_{j+1}) + \dfrac{a_{j+1} - a_j}{h_j} + h_j(c_j + c_{j+1})$

Final equation only for $j = 1, \cdots, n-1$, simplifies to

$$h_{j-1}\, c_{j-1} + 2(h_{j-1} + h_j)\, c_j + h_j c_{j+1} = 3\left(\frac{a_{j+1} - a_j}{h_j} - \frac{a_j - a_{j-1}}{h_{j-1}}\right).$$

The Splines Equations (V)

Final equations, for $j = 1, \cdots, n-1$:

$$h_{j-1}\, c_{j-1} + 2(h_{j-1} + h_j)\, c_j + h_j c_{j+1} = 3 \left(\frac{a_{j+1} - a_j}{h_j} - \frac{a_j - a_{j-1}}{h_{j-1}} \right).$$

$n-1$ equations with $n+1$ unknowns.

The Splines Equations (V)

Final equations, for $j = 1, \cdots, n-1$:

$$h_{j-1}\, c_{j-1} + 2(h_{j-1} + h_j)\, c_j + h_j c_{j+1} = 3\left(\frac{a_{j+1} - a_j}{h_j} - \frac{a_j - a_{j-1}}{h_{j-1}}\right).$$

$n-1$ equations with $n+1$ unknowns.

$$\boxed{\text{Under-defined problem. Need two additional conditions}}$$

The Splines Equations (V)

Final equations, for $j = 1, \cdots, n-1$:

$$h_{j-1}\, c_{j-1} + 2(h_{j-1} + h_j)\, c_j + h_j c_{j+1} = 3\left(\frac{a_{j+1} - a_j}{h_j} - \frac{a_j - a_{j-1}}{h_{j-1}}\right).$$

$n-1$ equations with $n+1$ unknowns.

$$\boxed{\text{Under-defined problem. Need two additional conditions}}$$

Special case, **Natural Splines**: $S_0''(x_0) = S_{n-1}''(x_n) = 0$:

- $c_0 = S_0''(x_0)/2 = 0$, $c_n = S_{n-1}''(x_n)/2 = 0$.
- $n-1$ equations with $n-1$ unknowns

$$\begin{aligned}
2(h_0 + h_1)\, c_1 + h_1 c_2 &= 3\left(\frac{a_2 - a_1}{h_1} - \frac{a_1 - a_0}{h_0}\right),\\
h_{j-1}\, c_{j-1} + 2(h_{j-1} + h_j)\, c_j + h_j c_{j+1} &= 3\left(\frac{a_{j+1} - a_j}{h_j} - \frac{a_j - a_{j-1}}{h_{j-1}}\right), 2 \le j \le n-3,\\
h_{n-2}\, c_{n-2} + 2(h_{n-2} + h_{n-1})\, c_{n-1} &= 3\left(\frac{a_n - a_{n-1}}{h_{n-1}} - \frac{a_{n-1} - a_{n-2}}{h_{n-2}}\right)
\end{aligned}$$

Natural Splines: equations in matrix form

- Equations for $\{c_j\}_{j=1}^{n-1}$,

$$\begin{pmatrix} 2(h_0+h_1) & h_1 & & & \\ h_1 & 2(h_1+h_2) & & & \\ & \ddots & \ddots & \ddots & \\ & & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} \\ & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-2} \\ c_{n-1} \end{pmatrix} = 3 \begin{pmatrix} \frac{a_2-a_1}{h_1} - \frac{a_1-a_0}{h_0} \\ \vdots \\ \frac{a_n-a_{n-1}}{h_{n-1}} - \frac{a_{n-1}-a_{n-2}}{h_{n-2}} \end{pmatrix}$$

- Equations for $\{d_j\}_{j=0}^{n-1}, \{b_j\}_{j=0}^{n-1}$,

$$d_j = \frac{c_{j+1}-c_j}{3h_j}, \quad \text{and} \quad b_j = -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1}-a_j}{h_j}.$$

Clamped Splines: $S_0'(x_0) = f'(x_0),\ S_{n-1}'(x_n) = f'(x_n)$

Recall

$$
\begin{aligned}
b_j &= -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1} - a_j}{h_j}. \ (\widehat{\ell}_0) \\
b_{j+1} &= b_j + h_j(c_j + c_{j+1}). \qquad\qquad (\widehat{\ell}_1)
\end{aligned}
$$

- Equation for c_0, c_1:

$$
\begin{aligned}
f'(x_0) &= S_0'(x_0) = b_0 \overset{(\widehat{\ell}_0)}{=\!=} -\frac{h_0}{3}(2c_0 + c_1) + \frac{a_1 - a_0}{h_0}. \\
2h_0 c_0 + h_0 c_1 &= 3\left(\frac{a_1 - a_0}{h_0} - f'(x_0)\right).
\end{aligned}
$$

- Equation for c_{n-1}, c_n:

$$
\begin{aligned}
f'(x_n) &= S_{n-1}'(x_n) = b_n \overset{(\widehat{\ell}_1)}{=\!=} b_{n-1} + h_{n-1}(c_{n-1} + c_n) \\
&\overset{(\widehat{\ell}_0)}{=\!=} -\frac{h_{n-1}}{3}(2c_{n-1} + c_n) + \frac{a_n - a_{n-1}}{h_{n-1}} + h_{n-1}(c_{n-1} + c_n),
\end{aligned}
$$

Clamped Splines: equations in matrix form

- Equations for $\{c_j\}_{j=0}^{n}$,

$$\begin{pmatrix} 2h_0 & h_0 & & & \\ h_0 & 2(h_0+h_1) & & & \\ & \ddots & \ddots & \ddots & \\ & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ & & & h_{n-1} & 2h_{n-1} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix} = 3 \begin{pmatrix} \frac{a_1-a_0}{h_0} - f'(x_0) \\ \frac{a_2-a_1}{h_1} - \frac{a_1-a_0}{h_0} \\ \vdots \\ \frac{a_n-a_{n-1}}{h_{n-1}} - \frac{a_{n-1}-a_{n-2}}{h_{n-2}} \\ f'(x_n) - \frac{a_n-a_{n-1}}{h_{n-1}} \end{pmatrix}.$$

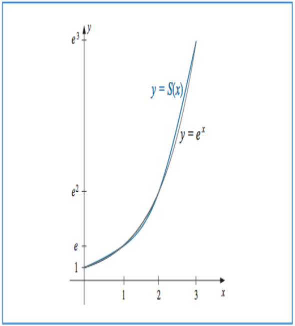
- Equations for $\{d_j\}_{j=0}^{n-1}, \{b_j\}_{j=0}^{n-1}$,

$$d_j = \frac{c_{j+1}-c_j}{3h_j}, \quad \text{and} \quad b_j = -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1}-a_j}{h_j}.$$

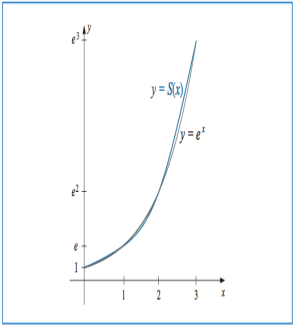
Clamped Splines

```
function Splines = ClampedSplines(x,f,df)
%
% This code implements the clamped splines
%
% Written by Ming Gu for Math 128A, Fall 2008
% Updated by Ming Gu for Math 128A, Spring 2015
%
n = length(x);
h = diff(x(:));
rhs = 3 * diff([df(1);diff(f(:))./h;df(2)]);

A = diag(h,1)+diag(h,-1)+2*diag([[0;h]+[h;0]]);

%
% compute the c coefficients. This is a simple
% but very slow way to do it.
%
c           = A \ rhs;
d           = (diff(c)./h)/3;
b           = diff(f(:))./h-(h/3).*(2*c(1:n-1)+c(2:n));
Splines.a = f(:);
Splines.b = b;
Splines.c = c;
Splines.d = d;
```

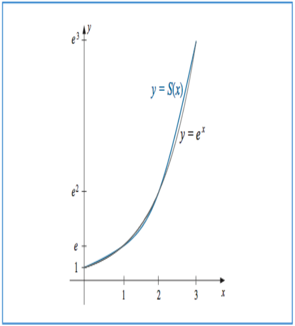
Natural Splines, $f(x) = e^x$**,**
$x_0 = 0, x_1 = 1, x_2 = 2, x_3 = 3$



Natural Splines, $f(x) = e^x$**,**
$x_0 = 0, x_1 = 1, x_2 = 2, x_3 = 3$



Clamped Splines, $f(x) = e^x$**,**
$x_0 = 0, x_1 = 1, x_2 = 2,$
$x_3 = 3, f'(0) = 1, f'(3) = e^3$



Review: Splines

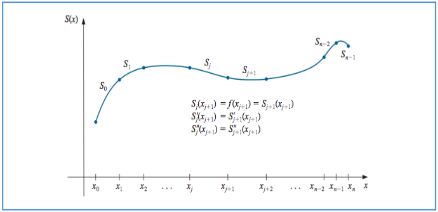
- Given $n+1$ distinct points

$$(x_0, f(x_0)), (x_1, f(x_1)), \cdots, (x_n, f(x_n)),$$

- Find *cubic spline interpolant* $S(x) \in C^2[x_0, x_n]$,

$$S(x) = S_j(x) \stackrel{def}{=} a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3,$$

  for $x \in [x_j, x_{j+1}]$, $0 \le j \le n-1$.
- $S(x_j) = f(x_j)$, $0 \le j \le n$.



Review: Splines

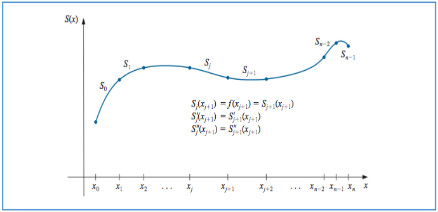
- Given $n+1$ distinct points

$$(x_0, f(x_0)), (x_1, f(x_1)), \cdots, (x_n, f(x_n)),$$

- Find *cubic spline interpolant* $S(x) \in C^2[x_0, x_n]$,

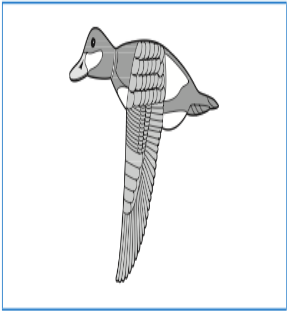
$$S(x) = S_j(x) \stackrel{def}{=} a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3,$$

  for $x \in [x_j, x_{j+1}]$, $0 \le j \le n-1$.
- $S(x_j) = f(x_j)$, $0 \le j \le n$.



A duck in Splines

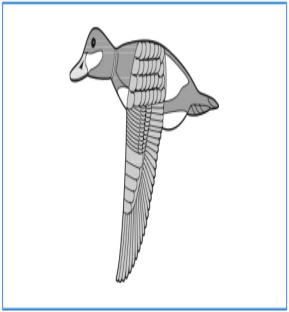
- A duck in flight



- Goal: to approximate the top profile.

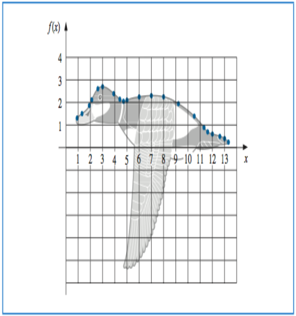
A duck in Splines

- A duck in flight



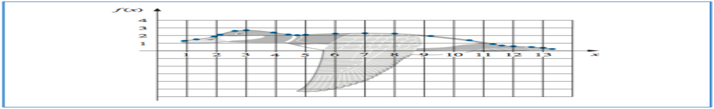
- Goal: to approximate the top profile.

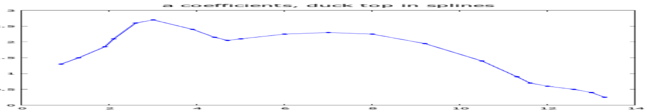
duck top profile



Duck top profile in Natural Splines, *a* coefficients

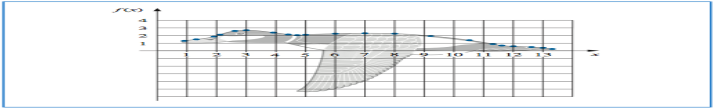
x	0.9	1.3	1.9	2.1	2.6	3.0	3.9	4.4	4.7	5.0	6.0	7.0	8.0	9.2	10.5	11.3	11.6	12.0	12.6	13.0	13.3
$f(x)$	1.3	1.5	1.85	2.1	2.6	2.7	2.4	2.15	2.05	2.1	2.25	2.3	2.25	1.95	1.4	0.9	0.7	0.6	0.5	0.4	0.25

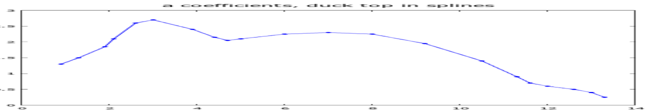




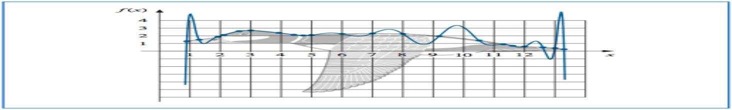
Duck top profile in Natural Splines, *a* coefficients

x	0.9	1.3	1.9	2.1	2.6	3.0	3.9	4.4	4.7	5.0	6.0	7.0	8.0	9.2	10.5	11.3	11.6	12.0	12.6	13.0	13.3
$f(x)$	1.3	1.5	1.85	2.1	2.6	2.7	2.4	2.15	2.05	2.1	2.25	2.3	2.25	1.95	1.4	0.9	0.7	0.6	0.5	0.4	0.25





Duck top profile in 20-degree polynomial interpolation



§3.6 Parametric Curve Approximation: $x = x(t), \quad y = y(t)$

- Given $n+1$ distinct points

$$(x_0, y_0), (x_1, y_1), \cdots, (x_n, y_n),$$

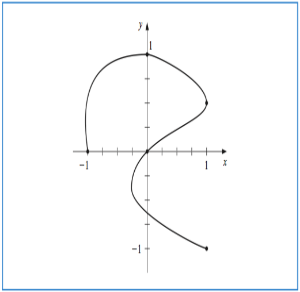
$$x_j = x(t_j), \quad y_j = y(t_j), \quad 0 \le j \le n.$$

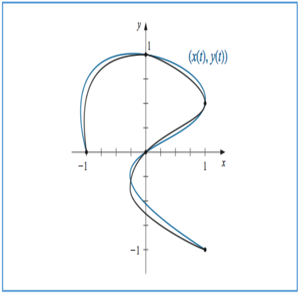
- Example

i	0	1	2	3	4
t_i	0	0.25	0.5	0.75	1
x_i	−1	0	1	0	1
y_i	0	1	0.5	0	−1

- 4^{th} degree interpolation on $x = x(t)$ and $y = y(t)$

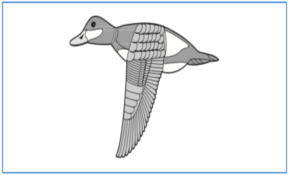
$$\begin{aligned} x(t) &= \left(\left(\left(64t - \frac{352}{3}\right)t + 60\right)t - \frac{14}{3}\right)t - 1, \\ y(t) &= \left(\left(\left(-\frac{64}{3}t + 48\right)t - \frac{116}{3}\right)t + 11\right)t. \end{aligned}$$





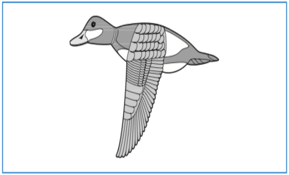
Bezier Curves in Computer Graphics

- **Design**: Piece-wise cubic Hermite polynomials.
- **Feature**: Each cubic Hermite polynomial is completely determined by function/derivative at endpoints.
- **Consequence**:, Each portion of the curve can be changed while leaving most of the curve the same.

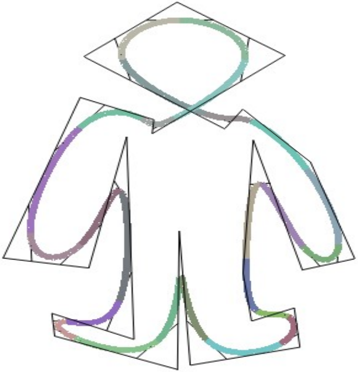


Bezier Curves in Computer Graphics

- **Design**: Piece-wise cubic Hermite polynomials.
- **Feature**: Each cubic Hermite polynomial is completely determined by function/derivative at endpoints.
- **Consequence**:, Each portion of the curve can be changed while leaving most of the curve the same.



- Bezier Curves with GUIDE POINTS



https://www.youtube.com/watch?v=TeXajQ62yZ8&ab_channel=corex