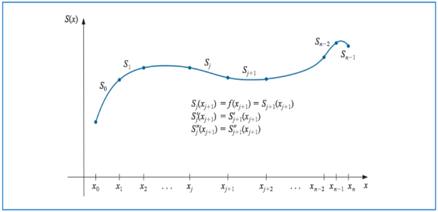
Review of Splines: Given $n+1$ distinct points

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

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



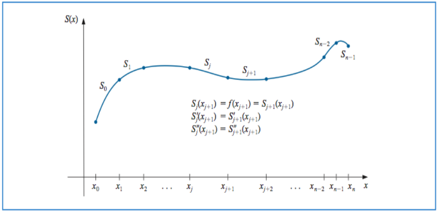
Review of Splines: Given $n+1$ distinct points

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

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



- Fake spine: $S_n(x) \stackrel{def}{=} a_n + b_n(x - x_n) + c_n(x - x_n)^2$, for $x \ge x_n$
- **Natural Splines**: $S''_0(x_0) = S''_{n-1}(x_n) = 0$.

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

Recall $n+1$ variables in c with $n-1$ equations:

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

where for $j = 0,\ 1, \cdots n-1$

$$\begin{aligned} b_j &= -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1} - a_j}{h_j}. \ (\widehat{\ell_0}) \\ b_{j+1} &= b_j + h_j(c_j + c_{j+1}). \qquad\qquad (\widehat{\ell_1}) \end{aligned}$$

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

Recall $n+1$ variables in c with $n-1$ equations:

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

where for $j = 0,\ 1, \cdots n-1$

$$\begin{aligned} b_j &= -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1} - a_j}{h_j}. \quad (\widehat{\ell_0}) \\ b_{j+1} &= b_j + h_j(c_j + c_{j+1}). \qquad\qquad (\widehat{\ell_1}) \end{aligned}$$

$$\begin{aligned} \text{Equation for } c_0, c_1\text{:}\ \ f'(x_0) &= S_0'(x_0) = b_0 \overset{(\widehat{\ell_0})}{=\!=} -\frac{h_0}{3}(2c_0 + c_1) + \frac{a_1 - a_0}{h_0}. \\ 2h_0 c_0 + h_0 c_1 &= 3\left(\frac{a_1 - a_0}{h_0} - f'(x_0)\right). \end{aligned}$$

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

Recall $n+1$ variables in c with $n-1$ equations:

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

where for $j = 0,\ 1, \cdots n-1$

$$\begin{aligned} b_j &= -\frac{h_j}{3}(2c_j + c_{j+1}) + \frac{a_{j+1} - a_j}{h_j}.\ (\widehat{\ell_0}) \\ b_{j+1} &= b_j + h_j(c_j + c_{j+1}). \qquad (\widehat{\ell_1}) \end{aligned}$$

$$\begin{aligned} \text{Equation for } c_0, c_1\text{: } f'(x_0) &= S_0'(x_0) = b_0 \overset{(\widehat{\ell_0})}{=\!=} -\frac{h_0}{3}(2c_0 + c_1) + \frac{a_1 - a_0}{h_0}. \\ 2h_0 c_0 + h_0 c_1 &= 3\left(\frac{a_1 - a_0}{h_0} - f'(x_0)\right). \\ \text{For } c_{n-1}, c_n\text{: } f'(x_n) &= S_{n-1}'(x_n) = b_n \overset{(\widehat{\ell_1})}{=\!=} b_{n-1} + h_{n-1}(c_{n-1} + c_n) \\ &\overset{(\widehat{\ell_0})}{=\!=} -\frac{h_{n-1}}{3}(2c_{n-1} + c_n) + \frac{a_n - a_{n-1}}{h_{n-1}} + h_{n-1}(c_{n-1} + c_n) \\ \text{or: } h_{n-1}c_{n-1} + 2h_{n-1}c_n &= 3\left(f'(x_n) - \frac{a_n - a_{n-1}}{h_{n-1}}\right) \end{aligned}$$

Clamped Splines: equations in matrix form

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

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

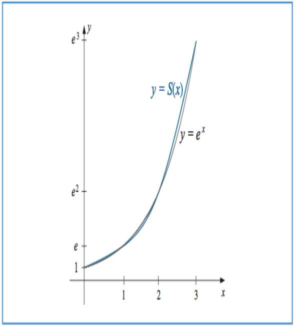
- Equations for $\{d_j\}_{j=0}^{n-1}, \{b_j\}_{j=0}^{n-1}$,

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

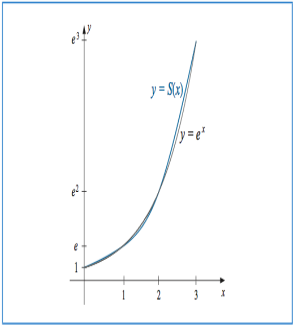
Clamped Splines

```
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;
```

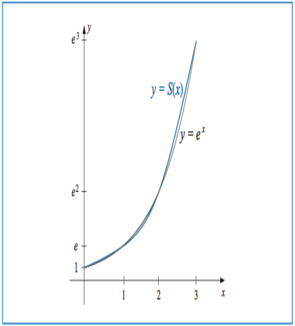
Natural Splines, $f(x) = e^x$, $x_0 = 0, x_1 = 1, x_2 = 2, x_3 = 3$



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

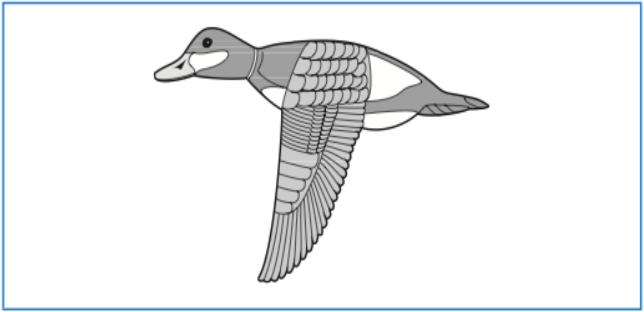


Clamped Splines, $f(x) = e^x$**,**
$x_0 = 0, x_1 = 1, x_2 = 2,$
$x_3 = 3, f'(0) = 1, f'(3) = e^3$



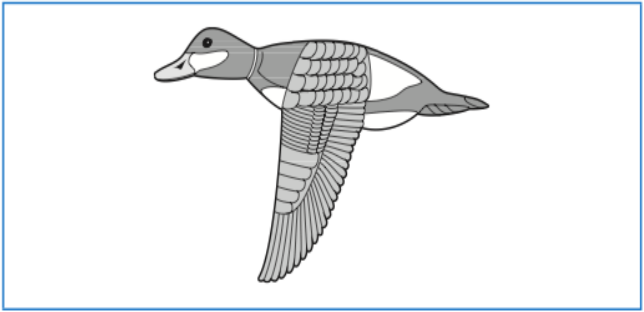
A duck in Splines: to approximate its top profile

- A duck in flight



A duck in Splines: to approximate its top profile

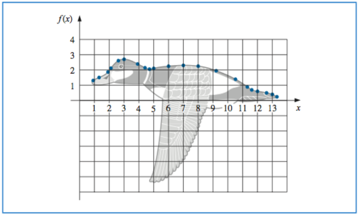
- A duck in flight

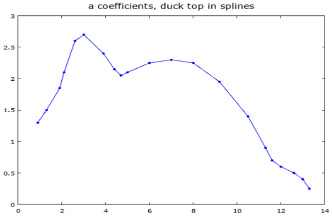


x	0.9	1.3	1.9	2.1	2.6	3.0	3.9	4.4	4.7	5.0	6.0	7.0	8.0	9.2	10.5	11.3	11.6	12.0	12.6	13.0	13.3
$f(x)$	1.3	1.5	1.85	2.1	2.6	2.7	2.4	2.15	2.05	2.1	2.25	2.3	2.25	1.95	1.4	0.9	0.7	0.6	0.5	0.4	0.25

Duck top profile

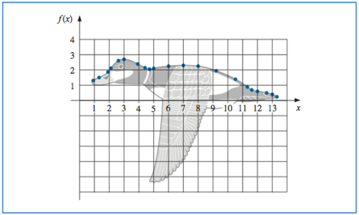
- Natural Splines, $\{a_k\}$ coefficients

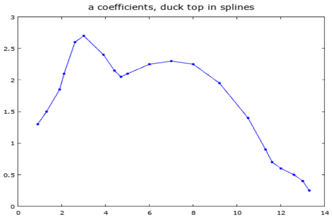




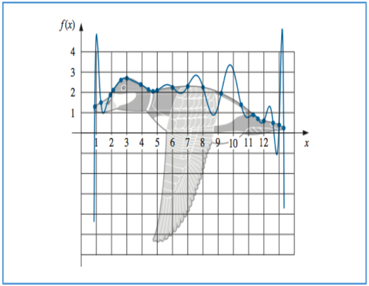
Duck top profile

- Natural Splines, $\{a_k\}$ coefficients





- 20-degree polynomial interpolation



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

- Given $n+1$ distinct points

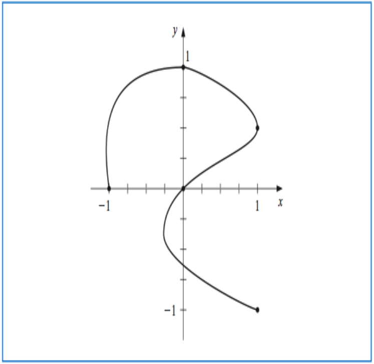
$$(x_0, y_0), (x_1, y_1), \cdots, (x_n, y_n),$$

where

$$x_j = x(t_j), \quad y_j = y(t_j), \quad j = 0, 1, \cdots, n.$$

- 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

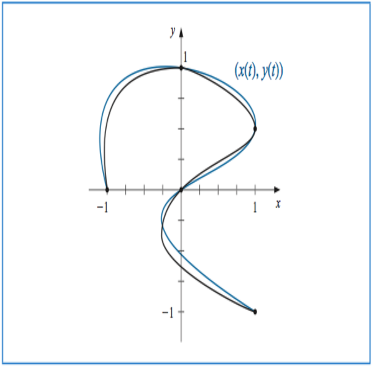


Parametric Curve with Polynomial Interpolation

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

$$x(t) = \left(\left(\left(64t - \frac{352}{3}\right) t + 60\right) t - \frac{14}{3}\right) t - 1,$$

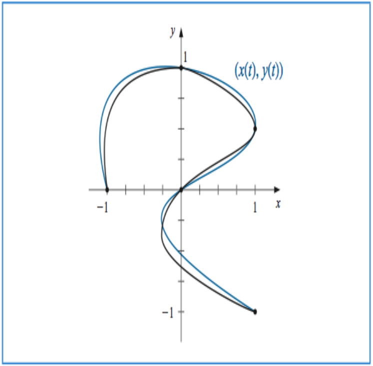
$$y(t) = \left(\left(\left(-\frac{64}{3} t + 48\right) t - \frac{116}{3}\right) t + 11\right) t.$$



Parametric Curve with Polynomial Interpolation

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

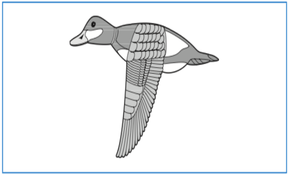
$$
\begin{aligned}
x(t) &= \left(\left(\left(64t - \frac{352}{3}\right) t + 60\right) t - \frac{14}{3}\right) t - 1, \\
y(t) &= \left(\left(\left(-\frac{64}{3} t + 48\right) t - \frac{116}{3}\right) t + 11\right) t.
\end{aligned}
$$



$\boxed{\text{Would like a better fit}}$

Bezier Curves in Computer Graphics

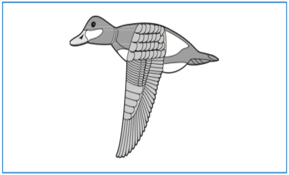
- **Design**: Piece-wise cubic Hermite polynomials.
- **Feature**: Each cubic Hermite polynomial is completely determined by function/derivative at endpoints.
- **Consequence**:, Each portion of the curve can be changed while leaving most of the curve the same.



As duck flies, parametric curve can effectively evolve.

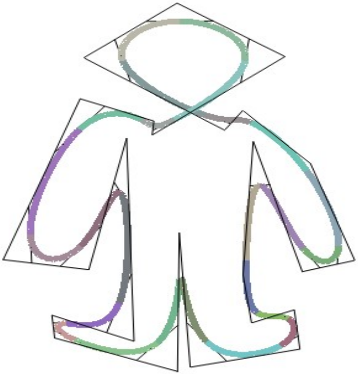
Bezier Curves in Computer Graphics

- **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.



As duck flies, parametric curve can effectively evolve.

- Bezier Curves with GUIDE POINTS



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

Parametric Curves: Piece-wise cubic Hermite polynomials

- **Given (I)**: $n+1$ data points $(x(t_0), y(t_0)), \cdots, (x(t_n), y(t_n))$.
- **Given (II)**: $n+1$ derivatives $\frac{dy}{dx}\Big|_{t=t_i}$, $0 \le i \le n$.

- **Find**: $2 \times n$ pieces of cubic Hermite polynomials:

$$x = x_i(t), \quad y = y_i(t), \quad \text{for} \quad i \in [t_i, t_{i+1}], \quad 0 \le i \le n,$$

such that

$$\begin{aligned}
\begin{pmatrix} x_i(t_i) \\ y_i(t_i) \end{pmatrix} &= \begin{pmatrix} x(t_i) \\ y(t_i) \end{pmatrix}, \begin{pmatrix} x_i(t_{i+1}) \\ y_i(t_{i+1}) \end{pmatrix} = \begin{pmatrix} x(t_{i+1}) \\ y(t_{i+1}) \end{pmatrix}, \\
\frac{dy}{dx}\Big|_{t=t_i} &= \frac{y_i'(t_i)}{x_i'(t_i)}, \quad \text{and} \quad \frac{dy}{dx}\Big|_{t=t_{i+1}} = \frac{y_i'(t_{i+1})}{x_i'(t_{i+1})}
\end{aligned}$$

Parametric Curves: Piece-wise cubic Hermite polynomials

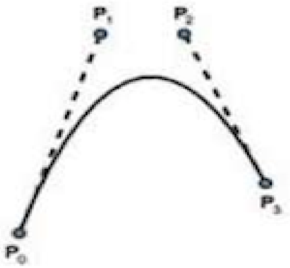
- **Given (I)**: $n+1$ data points $(x(t_0), y(t_0)), \cdots, (x(t_n), y(t_n))$.
- **Given (II)**: $n+1$ derivatives $\frac{dy}{dx}\Big|_{t=t_i}$, $0 \le i \le n$.
- **Find**: $2 \times n$ pieces of cubic Hermite polynomials:

$$x = x_i(t), \quad y = y_i(t), \quad \text{for} \quad i \in [t_i, t_{i+1}], \quad 0 \le i \le n,$$

such that

$$\begin{aligned} \begin{pmatrix} x_i(t_i) \\ y_i(t_i) \end{pmatrix} &= \begin{pmatrix} x(t_i) \\ y(t_i) \end{pmatrix}, \begin{pmatrix} x_i(t_{i+1}) \\ y_i(t_{i+1}) \end{pmatrix} = \begin{pmatrix} x(t_{i+1}) \\ y(t_{i+1}) \end{pmatrix}, \\ \frac{dy}{dx}\Big|_{t=t_i} &= \frac{y_i'(t_i)}{x_i'(t_i)}, \quad \text{and} \quad \frac{dy}{dx}\Big|_{t=t_{i+1}} = \frac{y_i'(t_{i+1})}{x_i'(t_{i+1})} \end{aligned}$$

- 8 parameters in $x_i(t)$, $y_i(t)$
- only 6 given conditions for each i
- $\boxed{\text{use guidepoints}}$

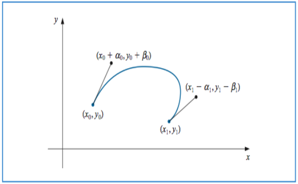


Cubic Bazier Curve

Guidepoints guide slopes (for $[t_i, t_{i+1}] = [0, 1]$)

- Unique cubic Hermite polynomials $x(t), y(t)$ with

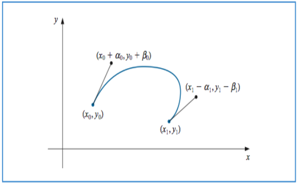
$$\begin{aligned}
\begin{pmatrix} x(0) \\ y(0) \end{pmatrix} &= \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}, \begin{pmatrix} x(1) \\ y(1) \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, \\
\begin{pmatrix} x'(0) \\ y'(0) \end{pmatrix} &= \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix}, \begin{pmatrix} x'(1) \\ y'(1) \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix}
\end{aligned}$$



Guidepoints guide slopes (for $[t_i, t_{i+1}] = [0, 1]$)

- Unique cubic Hermite polynomials $x(t), y(t)$ with

$$\begin{pmatrix} x(0) \\ y(0) \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}, \begin{pmatrix} x(1) \\ y(1) \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix},$$
$$\begin{pmatrix} x'(0) \\ y'(0) \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix}, \begin{pmatrix} x'(1) \\ y'(1) \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix}$$



- Guidepoints

$(x_0 + \alpha_0, y_0 + \beta_0)$ and $(x_1 - \alpha_1, y_1 - \beta_1)$

- Guidepoints guide slopes

$$\frac{dy}{dx}\,|_{t=0} = \frac{\beta_0}{\alpha_0}, \quad \text{and} \quad \frac{dy}{dx}\,|_{t=1} = \frac{\beta_1}{\alpha_1}.$$

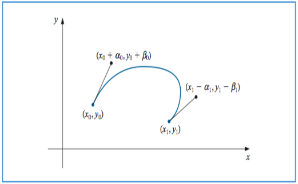
- two degrees of freedom in guidepoints

$$\beta_0 = \alpha_0 \left(\frac{dy}{dx}\,|_{t=0} \right), \quad \text{and} \quad \beta_1 = \alpha_1 \left(\frac{dy}{dx}\,|_{t=1} \right)$$

Guidepoints guide slopes (for $[t_i, t_{i+1}] = [0, 1]$)

- Unique cubic Hermite polynomials $x(t)$, $y(t)$ with

$$\begin{pmatrix} x(0) \\ y(0) \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix}, \begin{pmatrix} x(1) \\ y(1) \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix},$$
$$\begin{pmatrix} x'(0) \\ y'(0) \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix}, \begin{pmatrix} x'(1) \\ y'(1) \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix}$$



- Guidepoints

$$(x_0 + \alpha_0, y_0 + \beta_0) \text{ and } (x_1 - \alpha_1, y_1 - \beta_1)$$

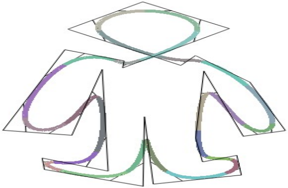
- Guidepoints guide slopes

$$\frac{dy}{dx}\Big|_{t=0} = \frac{\beta_0}{\alpha_0}, \quad \text{and} \quad \frac{dy}{dx}\Big|_{t=1} = \frac{\beta_1}{\alpha_1}.$$

- two degrees of freedom in guidepoints

$$\beta_0 = \alpha_0 \left(\frac{dy}{dx}\Big|_{t=0} \right), \quad \text{and} \quad \beta_1 = \alpha_1 \left(\frac{dy}{dx}\Big|_{t=1} \right)$$

- choice of guidepoints changes curve shapes

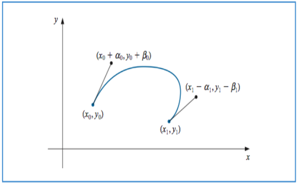


```
function [bx,by] = Bezier(x,y,alphal,betal,alphar,betar)

n = length(x);

bx = zeros(n-1,4);

by = zeros(n-1,4);

bx=[x(1:n-1),alphal(1:n-1),x(2:n)-x(1:n-1)-alphal(1:n-1),
2*(x(2:n)-x(1:n-1))-(alphal(1:n-1)+alphar(2:n))];

by=[y(1:n-1),betal(1:n-1),y(2:n)-y(1:n-1)-betal(1:n-1),
2*(y(2:n)-y(1:n-1))-(betal(1:n-1)+betar(2:n))];
```

Guidepoints guide slopes: example

- Unique cubic Hermite polynomials $x(t), y(t)$ with

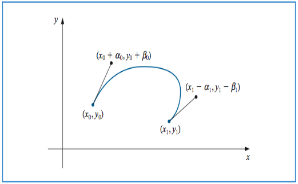
$$
\begin{aligned}
\begin{pmatrix} x(0) \\ y(0) \end{pmatrix} &= \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} x(1) \\ y(1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \\
\begin{pmatrix} x'(0) \\ y'(0) \end{pmatrix} &= \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \alpha_0 \end{pmatrix}, \\
\begin{pmatrix} x'(1) \\ y'(1) \end{pmatrix} &= \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ -\alpha_1 \end{pmatrix}
\end{aligned}
$$



Guidepoints guide slopes: example

- Unique cubic Hermite polynomials $x(t), y(t)$ with

$$
\begin{aligned}
\begin{pmatrix} x(0) \\ y(0) \end{pmatrix} &= \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} x(1) \\ y(1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \\
\begin{pmatrix} x'(0) \\ y'(0) \end{pmatrix} &= \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \alpha_0 \end{pmatrix}, \\
\begin{pmatrix} x'(1) \\ y'(1) \end{pmatrix} &= \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ -\alpha_1 \end{pmatrix}
\end{aligned}
$$



- Guidepoints guide slopes

$$
\frac{dy}{dx}\Big|_{t=0} = \frac{\beta_0}{\alpha_0} = 1, \quad \text{and} \quad \frac{dy}{dx}\Big|_{t=1} = \frac{\beta_1}{\alpha_1} = -1.
$$

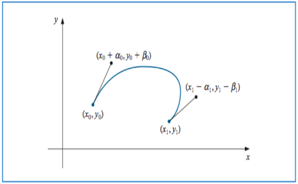
Guidepoints guide slopes: example

- Unique cubic Hermite polynomials $x(t), y(t)$ with

$$\begin{pmatrix} x(0) \\ y(0) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} x(1) \\ y(1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

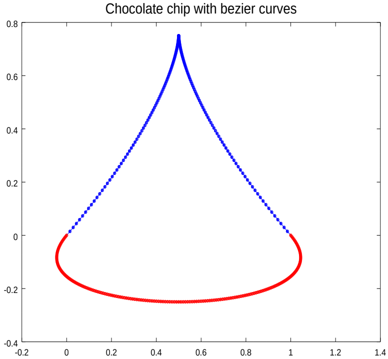
$$\begin{pmatrix} x'(0) \\ y'(0) \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \alpha_0 \end{pmatrix},$$

$$\begin{pmatrix} x'(1) \\ y'(1) \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ -\alpha_1 \end{pmatrix}$$



- Guidepoints guide slopes

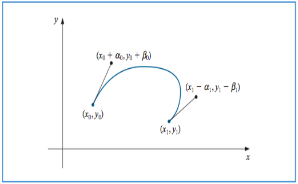
$$\frac{dy}{dx}\Big|_{t=0} = \frac{\beta_0}{\alpha_0} = 1, \quad \text{and} \quad \frac{dy}{dx}\Big|_{t=1} = \frac{\beta_1}{\alpha_1} = -1.$$



Guidepoints guide slopes: example

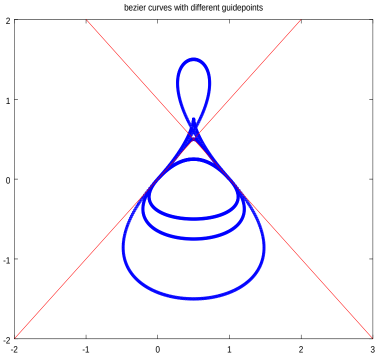
- Unique cubic Hermite polynomials $x(t), y(t)$ with

$$\begin{aligned}
\begin{pmatrix} x(0) \\ y(0) \end{pmatrix} &= \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} x(1) \\ y(1) \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \\
\begin{pmatrix} x'(0) \\ y'(0) \end{pmatrix} &= \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix} = \begin{pmatrix} \alpha_0 \\ \alpha_0 \end{pmatrix}, \\
\begin{pmatrix} x'(1) \\ y'(1) \end{pmatrix} &= \begin{pmatrix} \alpha_1 \\ \beta_1 \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ -\alpha_1 \end{pmatrix}
\end{aligned}$$



- Guidepoints guide slopes

$$\frac{dy}{dx}\,|_{t=0} = \frac{\beta_0}{\alpha_0} = 1, \quad \text{and} \quad \frac{dy}{dx}\,|_{t=1} = \frac{\beta_1}{\alpha_1} = -1.$$



Parametric Curves: Summary of approaches

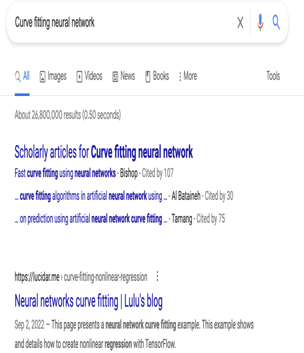
- **Given (I)**: $n+1$ data points $(x(t_0), y(t_0)), \cdots, (x(t_n), y(t_n))$.
- **Given (II)**: $n+1$ derivatives $\frac{dy}{dx}\big|_{t=t_i}$, $0 \le i \le n$.

Approaches

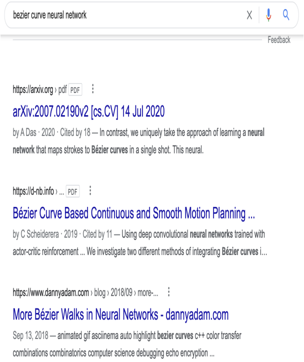
- Polynomial interpolations with $n+1$ data points $(x(t_0)), \cdots, (x(t_n))$ and $(y(t_0)), \cdots, (y(t_n))$, respectively. (likely to be unreliable)
- Splines approximations with $n+1$ data points $(x(t_0)), \cdots, (x(t_n))$ and $(y(t_0)), \cdots, (y(t_n))$, respectively. (more reliable and more computational cost)
- Bezier curves with Hermite polynomials (more reliable and less computational cost)

Neural Networks have changed the story

curve fitting



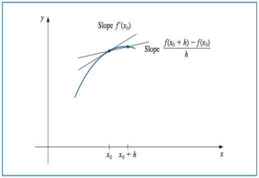
Bezier curve



§4.1 Numerical Differentiation

Derivative of given function $f(x)$ at x_0

$$
\begin{aligned}
f'(x_0) &= \lim_{h\to 0} \frac{f(x_0+h)-f(x_0)}{h} \\
&\approx \frac{f(x_0+h)-f(x_0)}{h}, \quad |h| \text{ tiny}
\end{aligned}
$$

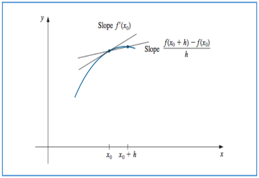


$\boxed{\text{How good is this approximation?}}$

§4.1 Numerical Differentiation

Derivative of given function $f(x)$ at x_0

$$
\begin{aligned}
f'(x_0) &= \lim_{h \to 0} \frac{f(x_0+h) - f(x_0)}{h} \\
&\approx \frac{f(x_0+h) - f(x_0)}{h}, \quad |h| \text{ tiny}
\end{aligned}
$$



$\boxed{\text{How good is this approximation?}}$

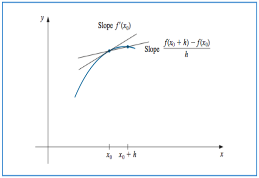
By Taylor expansion, there is a ξ between x_0 and $x_0 + h$,

$$
\begin{aligned}
f(x_0+h) &= f(x_0) + h\, f'(x_0) + \frac{1}{2} h^2\, f''(\xi), \\
\text{therefore} \quad f'(x_0) &= \frac{f(x_0+h) - f(x_0)}{h} - \frac{1}{2} h\, f''(\xi)
\end{aligned}
$$

§4.1 Numerical Differentiation

Derivative of given function $f(x)$ at x_0

$$
\begin{aligned}
f'(x_0) &= \lim_{h \to 0} \frac{f(x_0+h) - f(x_0)}{h} \\
&\approx \frac{f(x_0+h) - f(x_0)}{h}, \quad |h| \text{ tiny}
\end{aligned}
$$



$\boxed{\text{How good is this approximation?}}$

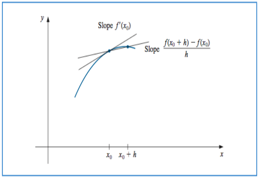
By Taylor expansion, there is a ξ between x_0 and $x_0 + h$,

$$
\begin{aligned}
f(x_0+h) &= f(x_0) + h\,f'(x_0) + \frac{1}{2}h^2\,f''(\xi), \\
\text{therefore} \quad f'(x_0) &= \frac{f(x_0+h) - f(x_0)}{h} - \frac{1}{2}h\,f''(\xi) \\
&\approx \frac{f(x_0+h) - f(x_0)}{h}.
\end{aligned}
$$

§4.1 Numerical Differentiation

Derivative of given function $f(x)$ at x_0

$$
\begin{aligned}
f'(x_0) &= \lim_{h \to 0} \frac{f(x_0+h) - f(x_0)}{h} \\
&\approx \frac{f(x_0+h) - f(x_0)}{h}, \quad |h| \text{ tiny}
\end{aligned}
$$



$$\boxed{\text{How good is this approximation?}}$$

By Taylor expansion, there is a ξ between x_0 and $x_0 + h$,

$$
\begin{aligned}
f(x_0+h) &= f(x_0) + h\, f'(x_0) + \frac{1}{2} h^2\, f''(\xi), \\
\text{therefore} \quad f'(x_0) &= \frac{f(x_0+h) - f(x_0)}{h} - \frac{1}{2} h\, f''(\xi) \\
&\approx \frac{f(x_0+h) - f(x_0)}{h}.
\end{aligned}
$$

- **forward-difference formula** for $h > 0$,
- **backward-difference formula** for $h < 0$.

Numerical Differentiation, example

Example: Use the forward-difference formula to approximate the derivative of $f(x) = \mathbf{ln}(x)$ at $x_0 = 1.8$, using $h = 0.1, 0.05, 0.01$.

Because $f''(\xi) = -1/\xi^2$ for $\xi \in [x_0, x_0 + h] \subset [1.8, 1.9]$, it follows that the approximation error

$$\frac{1}{2}\left|h\, f''(\xi)\right| \leq \frac{1}{2}\left|\frac{h}{1.8^2}\right|.$$

Numerical Differentiation, example

Example: Use the forward-difference formula to approximate the derivative of $f(x) = \mathbf{ln}(x)$ at $x_0 = 1.8$, using $h = 0.1, 0.05, 0.01$.

Because $f''(\xi) = -1/\xi^2$ for $\xi \in [x_0, x_0 + h] \subset [1.8, 1.9]$, it follows that the approximation error

$$\frac{1}{2}\left|h\, f''(\xi)\right| \leq \frac{1}{2}\left|\frac{h}{1.8^2}\right|.$$

h	$f(1.8+h)$	$\frac{f(1.8+h)-f(1.8)}{h}$	$\frac{\lvert h\rvert}{2(1.8)^2}$
0.1	0.64185389	0.5406722	0.0154321
0.05	0.61518564	0.5479795	0.0077160
0.01	0.59332685	0.5540180	0.0015432

Numerical Differentiation, via polynomial interpolation

Suppose that $\{x_0, x_1, \cdots, x_n\}$ are $n+1$ distinct numbers,

$$f(x) = \left(\sum_{j=0}^{n} f(x_j) L_j(x)\right) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{i=0}^{n} (x - x_i), \quad L_j(x) = \prod_{i \neq j} \frac{(x - x_i)}{(x_j - x_i)},$$

for some $\xi(x)$.

Numerical Differentiation, via polynomial interpolation

Suppose that $\{x_0, x_1, \cdots, x_n\}$ are $n+1$ distinct numbers,

$$f(x) = \left(\sum_{j=0}^{n} f(x_j) L_j(x)\right) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{i=0}^{n} (x - x_i), \quad L_j(x) = \prod_{i \neq j} \frac{(x - x_i)}{(x_j - x_i)},$$

for some $\xi(x)$. So

$$\begin{aligned} f'(x) &= \left(\sum_{j=0}^{n} f(x_j) L_j'(x)\right) + \left(\frac{f^{(n+1)}(\xi(x))}{(n+1)!}\right) \frac{d}{dx} \left(\prod_{i=0}^{n} (x - x_i)\right) \\ &\quad + \frac{d}{dx} \left(\frac{f^{(n+1)}(\xi(x))}{(n+1)!}\right) \left(\prod_{i=0}^{n} (x - x_i)\right). \end{aligned}$$

$$\boxed{\text{Last term messy, but } = 0 \text{ for } x = x_k, k = 0, 1, \cdots, n}$$

Numerical Differentiation, via polynomial interpolation

$$
\begin{aligned}
f'(x_k) &= \left(\sum_{j=0}^{n} f(x_j)L_j'(x_k)\right) + \left(\frac{f^{(n+1)}(\xi(x_k))}{(n+1)!}\right)\left(\prod_{i\neq k}(x_k - x_i)\right) \\
&\approx \sum_{j=0}^{n} f(x_j)L_j'(x_k).
\end{aligned}
$$

$(n+1)$-**point formula**, works (in theory) for any nodal choices.

2-point formulas ($n = 1$), $j = 0, 1$

$$
\begin{aligned}
L_0(x) &= \frac{x - x_1}{x_0 - x_1}, \quad L_0'(x) = \frac{1}{x_0 - x_1}, \\
L_1(x) &= \frac{x - x_0}{x_1 - x_0}, \quad L_1'(x) = \frac{1}{x_1 - x_0} = -L_0'(x)
\end{aligned}
$$

2-point formulas ($n = 1$), $j = 0, 1$

$$\begin{aligned} L_0(x) &= \frac{x - x_1}{x_0 - x_1}, \quad L_0'(x) = \frac{1}{x_0 - x_1}, \\ L_1(x) &= \frac{x - x_0}{x_1 - x_0}, \quad L_1'(x) = \frac{1}{x_1 - x_0} = -L_0'(x) \end{aligned}$$

So
$$\begin{aligned} f'(x_0) &\approx f(x_0)\, L_0'(x_0) + f(x_1)\, L_1'(x_0) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \\ f'(x_1) &\approx f(x_0)\, L_0'(x_1) + f(x_1)\, L_1'(x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \end{aligned}$$

Numerical Differentiation, via polynomial interpolation

$$
\begin{aligned}
f'(x_k) &= \left(\sum_{j=0}^{n} f(x_j) L_j'(x_k)\right) + \left(\frac{f^{(n+1)}(\xi(x_k))}{(n+1)!}\right)\left(\prod_{i\neq k}(x_k - x_i)\right) \\
&\approx \sum_{j=0}^{n} f(x_j) L_j'(x_k),
\end{aligned}
$$

$$
\text{where} \quad L_j'(x_k) = \begin{cases} \frac{1}{x_j - x_k} \prod_{i\neq j,k} \frac{(x_k - x_i)}{(x_j - x_i)}, & \text{for } k \neq j, \\ \sum_{i\neq j} \frac{1}{x_j - x_i} & \text{for } k = j. \end{cases}
$$

$(n+1)$-**point formula**, works for any nodal choices.

3-point formulas ($n = 2$), equi-spaced nodes

Choose $x_1 = x_0 + h, x_2 = x_0 + 2h$, with $h \neq 0$.

$$
\begin{aligned}
f'(x_0) &= \frac{1}{2\,h}\left(-3\,f(x_0) + 4\,f(x_1) - f(x_2)\right) + \frac{h^2}{3} f^{(3)} f(\xi_0) \\
&= \frac{1}{2\,h}\left(-3\,f(x_0) + 4\,f(x_0 + h) - f(x_0 + 2\,h)\right) + \frac{h^2}{3} f^{(3)} f(\xi_0) \\
f'(x_0 + h) &= \frac{1}{2\,h}\left(-f(x_0) + f(x_0 + 2\,h)\right) - \frac{h^2}{6} f^{(3)} f(\xi_1) \\
f'(x_0 + 2\,h) &= \frac{1}{2\,h}\left(f(x_0) - 4\,f(x_0 + h) + 3\,f(x_0 + 2\,h)\right) + \frac{h^2}{3} f^{(3)} f(\xi_2)
\end{aligned}
$$

Second order derivatives, equi-spaced points

$$
\begin{aligned}
f(x_0 + h) &= f(x_0) + f'(x_0)\, h + \frac{1}{2} f''(x_0)\, h^2 + \frac{1}{6} f'''(x_0)\, h^3 \\
&\quad + \frac{1}{24} f^{(4)}(\xi_+)\, h^4, \\
f(x_0 - h) &= f(x_0) - f'(x_0)\, h + \frac{1}{2} f''(x_0)\, h^2 - \frac{1}{6} f'''(x_0)\, h^3 \\
&\quad + \frac{1}{24} f^{(4)}(\xi_-)\, h^4.
\end{aligned}
$$

Second order derivatives, equi-spaced points

$$\begin{aligned} f(x_0+h) &= f(x_0)+f'(x_0)\,h+\frac{1}{2}f''(x_0)\,h^2+\frac{1}{6}f'''(x_0)\,h^3 \\ &\quad +\frac{1}{24}f^{(4)}(\xi_+)\,h^4, \\ f(x_0-h) &= f(x_0)-f'(x_0)\,h+\frac{1}{2}f''(x_0)\,h^2-\frac{1}{6}f'''(x_0)\,h^3 \\ &\quad +\frac{1}{24}f^{(4)}(\xi_-)\,h^4. \end{aligned}$$

Adding up, terms with difference signs cancel,

$$\begin{aligned} f(x_0+h)+f(x_0-h) &= 2f(x_0)+f''(x_0)\,h^2+\frac{h^4}{24}\left(f^{(4)}(\xi_+)+f^{(4)}(\xi_-)\right) \\ &= 2f(x_0)+f''(x_0)\,h^2+\frac{2h^4}{24}f^{(4)}(\xi). \end{aligned}$$

Second order derivatives, equi-spaced points

$$
\begin{aligned}
f(x_0+h) &= f(x_0) + f'(x_0)\,h + \frac{1}{2}f''(x_0)\,h^2 + \frac{1}{6}f'''(x_0)\,h^3 \\
&\quad + \frac{1}{24}f^{(4)}(\xi_+)\,h^4, \\
f(x_0-h) &= f(x_0) - f'(x_0)\,h + \frac{1}{2}f''(x_0)\,h^2 - \frac{1}{6}f'''(x_0)\,h^3 \\
&\quad + \frac{1}{24}f^{(4)}(\xi_-)\,h^4.
\end{aligned}
$$

Adding up, terms with difference signs cancel,

$$
\begin{aligned}
f(x_0+h) + f(x_0-h) &= 2f(x_0) + f''(x_0)\,h^2 + \frac{h^4}{24}\Big(f^{(4)}(\xi_+) + f^{(4)}(\xi_-)\Big) \\
&= 2f(x_0) + f''(x_0)\,h^2 + \frac{2h^4}{24}f^{(4)}(\xi).
\end{aligned}
$$

Therefore

$$
f''(x_0) = \frac{f(x_0+h) + f(x_0-h) - 2f(x_0)}{h^2} - \frac{h^2}{12}f^{(4)}(\xi).
$$

Round-Off Error Instability: example

Three-point midpoint formula

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi).$$

- every computation incurs round-off error.
- division by $2h$ magnifies round-off error
- assume round-off error model

$$\begin{aligned} f(x_0+h) &= \widehat{f}(x_0+h) + e(x_0+h) \\ f(x_0-h) &= \widehat{f}(x_0-h) + e(x_0-h) \end{aligned}$$

for $|e(x_0+h)| \le \epsilon, \ |e(x_0-h)| \le \epsilon$

- no other round-off errors

Round-Off Error Instability: example

Three-point midpoint formula

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi).$$

- every computation incurs round-off error.
- division by $2h$ magnifies round-off error
- assume round-off error model

$$\begin{aligned} f(x_0+h) &= \widehat{f}(x_0+h) + e(x_0+h) \\ f(x_0-h) &= \widehat{f}(x_0-h) + e(x_0-h) \end{aligned}$$

for $|e(x_0+h)| \le \epsilon,\ |e(x_0-h)| \le \epsilon$

- no other round-off errors

It follows

$$\begin{aligned} & f'(x_0) - \frac{\widehat{f}(x_0+h) - \widehat{f}(x_0-h)}{2h} \\ & \quad = \frac{e(x_0+h) - e(x_0-h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi) \\ & \left| f'(x_0) - \frac{\widehat{f}(x_0+h) - \widehat{f}(x_0-h)}{2h} \right| \\ & \quad \le \left| \frac{e(x_0+h) - e(x_0-h)}{2h} \right| + \frac{h^2}{6} |f^{(3)}(\xi)| \end{aligned}$$

Round-Off Error Instability: example

Three-point midpoint formula

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi).$$

- every computation incurs round-off error.
- division by $2h$ magnifies round-off error
- assume round-off error model

$$\begin{aligned} f(x_0+h) &= \widehat{f}(x_0+h) + e(x_0+h) \\ f(x_0-h) &= \widehat{f}(x_0-h) + e(x_0-h) \end{aligned}$$

for $|e(x_0+h)| \le \epsilon, \ |e(x_0-h)| \le \epsilon$

- no other round-off errors

It follows

$$\begin{aligned} & f'(x_0) - \frac{\widehat{f}(x_0+h) - \widehat{f}(x_0-h)}{2h} \\ & \quad = \frac{e(x_0+h) - e(x_0-h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi) \\ & \left| f'(x_0) - \frac{\widehat{f}(x_0+h) - \widehat{f}(x_0-h)}{2h} \right| \\ & \quad \le \left| \frac{e(x_0+h) - e(x_0-h)}{2h} \right| + \frac{h^2}{6} |f^{(3)}(\xi)| \end{aligned}$$

- assume an upper bound: $|f^{(3)}(\xi)| \le M$

$$\left| f'(x_0) - \frac{\widehat{f}(x_0+h) - \widehat{f}(x_0-h)}{2h} \right| \le \frac{\epsilon}{h} + \frac{M\,h^2}{6} \stackrel{def}{=} e(h).$$

Round-Off Error Instability: example

Three-point midpoint formula

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi).$$

- every computation incurs round-off error.
- division by $2h$ magnifies round-off error
- assume round-off error model

$$\begin{aligned} f(x_0+h) &= \widehat{f}(x_0+h) + e(x_0+h) \\ f(x_0-h) &= \widehat{f}(x_0-h) + e(x_0-h) \end{aligned}$$

for $|e(x_0+h)| \le \epsilon, \ |e(x_0-h)| \le \epsilon$

- no other round-off errors

It follows

$$f'(x_0) - \frac{\widehat{f}(x_0+h) - \widehat{f}(x_0-h)}{2h} = \frac{e(x_0+h) - e(x_0-h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi)$$

$$\left| f'(x_0) - \frac{\widehat{f}(x_0+h) - \widehat{f}(x_0-h)}{2h} \right| \le \left| \frac{e(x_0+h) - e(x_0-h)}{2h} \right| + \frac{h^2}{6} |f^{(3)}(\xi)|$$

- assume an upper bound: $|f^{(3)}(\xi)| \le M$

$$\left| f'(x_0) - \frac{\widehat{f}(x_0+h) - \widehat{f}(x_0-h)}{2h} \right| \le \frac{\epsilon}{h} + \frac{M\,h^2}{6} \stackrel{def}{=} e(h).$$

- $e(h)$ too big as $h \longrightarrow 0^+$

Round-Off Error Instability: optimal h choice

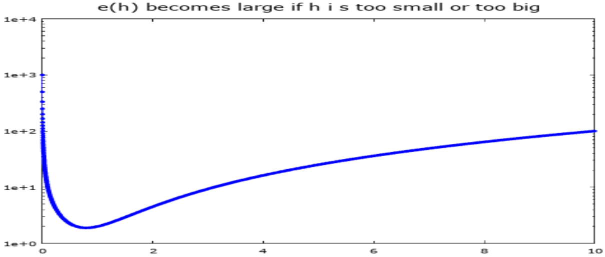
$$e(h) = \frac{\epsilon}{h} + \frac{M\,h^2}{6}$$

is smallest at

$$h_{\mathbf{min}} = \left(\frac{3\epsilon}{M}\right)^{\frac{1}{3}} = O\left(\epsilon^{\frac{1}{3}}\right),$$

with

$$e(h_{\mathbf{min}}) = \frac{1}{2}\left(9\,M\,\epsilon^2\right)^{\frac{1}{3}} = O\left(\epsilon^{\frac{2}{3}}\right).$$



§4.2 Richardson's Extrapolation

- **Given (I)**: A formula $N_1(h)$ that approximates an unknown constant M for any $h \neq 0$.

§4.2 Richardson's Extrapolation

- **Given (I)**: A formula $N_1(h)$ that approximates an unknown constant M for any $h \neq 0$.
  $\left(\text{Example: } f'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h}\right)$

§4.2 Richardson's Extrapolation

- **Given (I)**: A formula $N_1(h)$ that approximates an unknown constant M for any $h \neq 0$.
  $\left(\text{Example: } f'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h}\right)$
- **Given (II)**: Truncation error satisfies power series for $h \neq 0$

$$M - N_1(h) = K_1 h + K_2 h^2 + K_3 h^3 + \cdots = O\left(h\right), \qquad (1)$$

with (unknown) constants $K1, K_2, K_3, \cdots$.

§4.2 Richardson's Extrapolation

- **Given (I)**: A formula $N_1(h)$ that approximates an unknown constant M for any $h \neq 0$.
  $\left(\text{Example: } f'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h}\right)$
- **Given (II)**: Truncation error satisfies power series for $h \neq 0$

$$M - N_1(h) = K_1 h + K_2 h^2 + K_3 h^3 + \cdots = O(h), \qquad (1)$$

with (unknown) constants $K1, K_2, K_3, \cdots$.

$$\left(f'(x_0) - \frac{f(x_0+h)-f(x_0)}{h} = -\frac{1}{2}\, h\, f''(x_0) - \cdots - \frac{1}{n!}\, h^{n-1}\, f^{(n)}(x_0) - \cdots\right)$$

$\boxed{\textbf{Goal}\text{: Generate higher order approximations}}$

- **Key**: Equation (1) works for *any* $h \neq 0$, including $\dfrac{h}{2}$.

Extrapolation, Step I

$$
\begin{aligned}
M - N_1(h) &= K_1 h + K_2 h^2 + K_3 h^3 + \cdots , && (1)\\
M - N_1(\frac{h}{2}) &= K_1(\frac{h}{2}) + K_2(\frac{h}{2})^2 + K_3(\frac{h}{2})^3 + \cdots . && (2)
\end{aligned}
$$

(2)$\times$2$-$ (1):

$$
\begin{aligned}
M - N_2(h) &= -\frac{K_2}{2} h^2 - \frac{3K_3}{4} h^3 - \cdots - (1 - 2^{-(t-1)}) K_t h^t - \cdots , \\
&\overset{def}{=} \widehat{K}_2 h^2 + \widehat{K}_3 h^3 + \cdots \widehat{K}_t h^t + \cdots = O\left(h^2\right) , && (3)
\end{aligned}
$$

$$
\text{where} \quad N_2(h) = N_1(\frac{h}{2}) + \left(N_1(\frac{h}{2}) - N_1(h) \right) .
$$

$\boxed{\text{Equation (3) again power series, but now 2nd order.}}$

Extrapolation, Step II

$$\begin{aligned} M - N_2(h) &= \widehat{K}_2 h^2 + \widehat{K}_3 h^3 + \cdots \widehat{K}_t h^t + \cdots, & (3) \\ M - N_2(\frac{h}{2}) &= \widehat{K}_2 (\frac{h}{2})^2 + \widehat{K}_3 (\frac{h}{2})^3 + + \cdots. & (4) \end{aligned}$$

$\dfrac{(4)\times 2^2 - (3)}{2^2 - 1}$:

$$M - N_3(h) = -\frac{\widehat{K}_3}{6} h^3 - \cdots - \frac{1 - 2^{-(t-2)}}{3} \widehat{K}_t h^t - \cdots, \quad (5)$$

$$\text{where} \quad N_3(h) \stackrel{def}{=} N_2(\frac{h}{2}) + \frac{N_2(\frac{h}{2}) - N_2(h)}{3}.$$

$\boxed{\text{Equation (5) is one more power series, but 3rd order.}}$

Extrapolation, Step III

- Assume

$$M - N_j(h) = \widehat{K}_j h^j + \widehat{K}_{j+1} h^{j+1} + \cdots \widehat{K}_t h^t + \cdots ,$$

- replace h by $h/2$:

$$M - N_j(\frac{h}{2}) = \widehat{K}_j (\frac{h}{2})^j + \widehat{K}_{j+1} (\frac{h}{2})^{j+1} + \cdots \widehat{K}_t (\frac{h}{2})^t + \cdots .$$

$\dfrac{\text{second equation } \times 2^j - \text{ first equation}}{2^j - 1}$:

$$M - N_{j+1}(h) \quad = \quad -\frac{\widehat{K}_{j+1}}{2(2^j - 1)} h^{j+1} - \cdots = O\left(h^{j+1}\right),$$

$$\text{where} \quad N_{j+1}(h) \stackrel{def}{=} N_j(\frac{h}{2}) + \frac{N_j(\frac{h}{2}) - N_j(h)}{2^j - 1}.$$

Richardson's Extrapolation Table

$$
\begin{array}{cccc}
O(h) & O(h^2) & O(h^3) & O(h^4) \\
\hline
N_1(h) \searrow \atop \rightarrow & & & \\
N_1(\frac{h}{2}) \searrow \atop \rightarrow & N_2(h) \searrow \atop \rightarrow & & \\
N_1(\frac{h}{4}) \searrow & N_2(\frac{h}{2}) \searrow & N_3(h) \searrow & \\
N_1(\frac{h}{8}) \rightarrow & N_2(\frac{h}{4}) \rightarrow & N_3(\frac{h}{2}) \rightarrow & N_4(h)
\end{array}
$$

Richardson's Extrapolation: even power series

- **Given (I)**: A formula $N_1(h)$ that approximates an unknown constant M for any $h \neq 0$.
- **Given (II)**: Truncation error satisfies even power series

$$M - N_1(h) = K_1 h^2 + K_2 h^4 + K_3 h^6 + \cdots = O\left(h^2\right), \qquad (1)$$

with (unknown) constants $K1, K_2, K_3, \cdots$.

$\boxed{\textbf{Goal}\text{: Generate higher order approximations}}$

- **Key**: Equation (1) works for *any* h, including $\frac{h}{2}$.

Even power extrapolation, Step I

$$
\begin{aligned}
M - N_1(h) &= K_1 h^2 + K_2 h^4 + K_3 h^6 + \cdots, & (1)\\
M - N_1(\frac{h}{2}) &= K_1 (\frac{h}{2})^2 + K_2 (\frac{h}{2})^4 + K_3 (\frac{h}{2})^6 + \cdots. & (2)
\end{aligned}
$$

$\dfrac{(2)\times 2^2 - (1)}{2^2 - 1}$:

$$
\begin{aligned}
M - N_2(h) &= -\frac{K_2}{4} h^4 - \frac{5K_3}{16} h^6 - \cdots - \frac{1 - 2^{-2(t-1)}}{3} K_t h^{2t} - \cdots,\\
&\stackrel{def}{=} \widehat{K}_2 h^4 + \widehat{K}_3 h^6 + \cdots \widehat{K}_t h^{2t} + \cdots = O\left(h^4\right), & (3)
\end{aligned}
$$

$$
\text{where} \quad N_2(h) = N_1(\frac{h}{2}) + \frac{N_1(\frac{h}{2}) - N_1(h)}{3}.
$$

$\boxed{\text{Equation (3) again even power series, but now 4th order.}}$

Even extrapolation, Step II

$$
\begin{aligned}
M - N_2(h) &= \widehat{K}_2 h^4 + \widehat{K}_3 h^6 + \cdots \widehat{K}_t h^{2t} + \cdots, & (3)\\
M - N_2(\frac{h}{2}) &= \widehat{K}_2(\frac{h}{2})^4 + \widehat{K}_3(\frac{h}{2})^6 + \cdots. & (4)
\end{aligned}
$$

$\dfrac{(4)\times 2^4 - (3)}{2^4 - 1}$:

$$
M - N_3(h) \;=\; -\frac{\widehat{K}_3}{20} h^6 - \cdots - \frac{1 - 2^{-2(t-2)}}{15} \widehat{K}_t h^{2t} - \cdots,
$$

$$
\text{where} \quad N_3(h) \stackrel{def}{=} N_2(\frac{h}{2}) + \frac{N_2(\frac{h}{2}) - N_2(h)}{15}.
$$

$\boxed{\text{One more even power series, but now 6-th order.}}$

Even extrapolation, Step III

- Assume

$$M - N_j(h) = \widehat{K}_j h^{2j} + \widehat{K}_{j+1} h^{2(j+1)} + \cdots \widehat{K}_t h^{2t} + \cdots ,$$

- replace h by $h/2$:

$$M - N_j(\frac{h}{2}) = \widehat{K}_j(\frac{h}{2})^{2j} + \widehat{K}_{j+1}(\frac{h}{2})^{2(j+1)} + \cdots \widehat{K}_t(\frac{h}{2})^{2t} + \cdots .$$

$\dfrac{\text{second equation } \times 4^j - \text{ first equation}}{4^j - 1}$:

$$M - N_{j+1}(h) \quad = \quad -\frac{3\widehat{K}_{j+1}}{4(4^j - 1)} h^{2(j+1)} - \cdots = O\left(h^{2(j+1)}\right),$$

$$\text{where} \quad N_{j+1}(h) \stackrel{def}{=} N_j(\frac{h}{2}) + \frac{N_j(\frac{h}{2}) - N_j(h)}{4^j - 1}.$$

Richardson's Even Extrapolation Table

$$
\begin{array}{cccc}
O(h^2) & O(h^4) & O(h^6) & O(h^8) \\
\hline
N_1(h) \searrow & & & \\
\rightarrow & & & \\
N_1(\frac{h}{2}) \searrow & N_2(h) \searrow & & \\
\rightarrow & \rightarrow & & \\
N_1(\frac{h}{4}) \searrow & N_2(\frac{h}{2}) \searrow & N_3(h) \searrow & \\
N_1(\frac{h}{8}) \rightarrow & N_2(\frac{h}{4}) \rightarrow & N_3(\frac{h}{2}) \rightarrow & N_4(h)
\end{array}
$$

Even extrapolation, example

Consider Taylor expansion for a given function $f(x)$ with $h > 0$:

$$\begin{aligned} f(x+h) &= f(x) + f'(x)\,h + \frac{1}{2}f''(x)\,h^2 + \cdots + \frac{1}{n!}f^{(n)}(x)\,h^n + \cdots\,, \\ f(x-h) &= f(x) - f'(x)\,h + \frac{1}{2}f''(x)\,h^2 + \cdots + \frac{1}{n!}f^{(n)}(x)\,(-h)^n + \cdots\,. \end{aligned}$$

Take the difference:

$$\frac{f(x+h)-f(x-h)}{2h} = f'(x) + \cdots + \frac{1}{(2t+1)!}f^{(2t+1)}(x)\,h^{2t} + \cdots\,.$$

This is an even power series with 2nd order approximation

$$f'(x), \approx N_1(h) \stackrel{def}{=} \frac{f(x+h)-f(x-h)}{2h}.$$

Even extrapolation, example

Consider Taylor expansion for a given function $f(x)$ with $h > 0$:

$$\begin{aligned} f(x+h) &= f(x) + f'(x)\,h + \frac{1}{2}f''(x)\,h^2 + \cdots + \frac{1}{n!}f^{(n)}(x)\,h^n + \cdots\ , \\ f(x-h) &= f(x) - f'(x)\,h + \frac{1}{2}f''(x)\,h^2 + \cdots + \frac{1}{n!}f^{(n)}(x)\ (-h)^n + \cdots\ . \end{aligned}$$

Take the difference:

$$\frac{f(x+h) - f(x-h)}{2h} = f'(x) + \cdots + \frac{1}{(2t+1)!}f^{(2t+1)}(x)\,h^{2t} + \cdots\ .$$

This is an even power series with 2nd order approximation

$$f'(x), \approx N_1(h) \stackrel{def}{=} \frac{f(x+h) - f(x-h)}{2h}.$$

4th order approximation

$$\begin{aligned} f'(x) &\approx N_1(h) + \frac{N_1(h) - N_1(2h)}{3} \\ &= \frac{f(x-2h) - 8f(x-h) + 8f(x+h) - f(x+2h)}{12h}. \end{aligned}$$

Even extrapolation, example

Approximate $f'(2.0)$ with central difference $N_1(0.1)$ and extrapolation $N_2(0.1)$ for $f(x) = xe^x$.

Solution: $f'(x) = (1+x)e^x$, therefore $f'(2.0) = 3e^2 \approx 22.167$.

- $$N_1(0.1) = \frac{f(2.1) - f(1.9)}{2 \times 0.1} \approx 22.229.$$
- $$\begin{aligned} N_1(0.2) &= \frac{f(2.2) - f(1.8)}{2 \times 0.2} \approx 22.414. \\ N_2(0.2) &= N_1(h) + \frac{N_1(h) - N_1(2h)}{3} \approx 22.167. \end{aligned}$$

Midterm Exam: Oct. 12, in class

- covers material up to and include §4.2 (six questions total)
  - one question each from chapters 1 and 4
  - two questions each from chapters 2 and 3

Midterm Exam: Oct. 12, in class

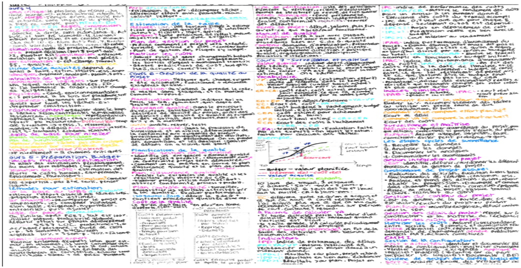
- covers material up to and include §4.2 (six questions total)
  - one question each from chapters 1 and 4
  - two questions each from chapters 2 and 3
- DSP students will get DSP recommended extra time
- Sample exam has been provided.
  Solutions available over the weekend

Midterm Exam: Oct. 12, in class

- covers material up to and include §4.2 (six questions total)
  - one question each from chapters 1 and 4
  - two questions each from chapters 2 and 3
- DSP students will get DSP recommended extra time
- Sample exam has been provided.
  Solutions available over the weekend
- one page cheat sheet on one side only

Midterm Exam: Oct. 12, in class

- covers material up to and include §4.2 (six questions total)
  - one question each from chapters 1 and 4
  - two questions each from chapters 2 and 3
- DSP students will get DSP recommended extra time
- Sample exam has been provided.
  Solutions available over the weekend
- one page cheat sheet on one side only
- You can skip the exam, but this is NOT encouraged
  - Final worth 50 (as opposed to 30) points if you do skip.
  - If you submit the exam, it WILL count.



Sample exam problem

- How many multiplications and additions are required to determine a sum of the form

$$S \stackrel{def}{=} \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_i \beta_j \qquad (1)$$

- Modify the sum in part (1) to an equivalent form that reduces the number of computations.

Sample exam problem

- How many multiplications and additions are required to determine a sum of the form

$$S \stackrel{def}{=} \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_i \, \beta_j \qquad (1)$$

- Modify the sum in part (1) to an equivalent form that reduces the number of computations.

SOLUTION:

- it takes $m\,n$ multiplications and $m\,n$ additions in (1).
- Rewrite

$$S \stackrel{def}{=} \left(\sum_{i=1}^{n} \alpha_i\right) \left(\sum_{j=1}^{m} \beta_j\right)$$

it takes 1 multiplication and $m + n$ additions.

Sample exam problem

The following two methods are proposed to compute $7^{1/5}$. Discuss their orders of convergence, assuming $p_0 = 1$.

1. $p_{n+1} = p_n - \frac{p_n^5 - 7}{5\, p_n^4}$
2. $p_{n+1} = p_n - \frac{p_n^5 - 7}{100}$

Sample exam problem

The following two methods are proposed to compute $7^{1/5}$.
Discuss their orders of convergence, assuming $p_0 = 1$.

1. $p_{n+1} = p_n - \frac{p_n^5 - 7}{5\, p_n^4}$
2. $p_{n+1} = p_n - \frac{p_n^5 - 7}{100}$

SOLUTION:

1. Newton's method, quadratic convergence.
2. fixed point iteration, linear convergence.

Sample exam problem

A quadratic spline interpolating function S defined with the nodes $x_0 < x_1 < x_2$ is such that S is a quadratic polynomial on each of the intervals $[x_0, x_1]$ and $[x_1, x_2]$, respectively. Assume that $S(x) \in C^2[x_0, x_2]$. Show that S must be a quadratic polynomial on the entire interval $[x_0, x_2]$.

Sample exam problem

A quadratic spline interpolating function S defined with the nodes $x_0 < x_1 < x_2$ is such that S is a quadratic polynomial on each of the intervals $[x_0, x_1]$ and $[x_1, x_2]$, respectively. Assume that $S(x) \in C^2[x_0, x_2]$. Show that S must be a quadratic polynomial on the entire interval $[x_0, x_2]$.

SOLUTION: Parameterize S as

$$S(x) = \begin{cases} a_0 + b_0\,(x - x_1) + c_0\,(x - x_1)^2, & \text{if } x \in [x_0, x_1], \\ a_1 + b_1\,(x - x_1) + c_1\,(x - x_1)^2, & \text{if } x \in [x_1, x_2]. \end{cases}$$

The condition that $S(x) \in C^2[x_0, x_2]$ implies that

$$S(x)\Big|_{x=x_1^-} = S(x)\Big|_{x=x_1^+},\ S(x)'\Big|_{x=x_1^-} = S(x)'\Big|_{x=x_1^+},\ S(x)''\Big|_{x=x_1^-} = S(x)''\Big|_{x=x_1^+}$$

This leads to

$$a_0 = a_1, \quad b_0 = b_1, \quad c_0 = c_1$$

§4.3 Numerical Integration: general idea

- **Goal**: Numerical method /quadrature for approximating $\int_a^b f(x)dx$.

§4.3 Numerical Integration: general idea

- **Goal**: Numerical method /quadrature for approximating $\int_a^b f(x)dx$.
- **Approach**: Replacing $f(x)$ by a polynomial.
  - **Choose**: $n+1$ points $a \le x_0 < x_1 < \cdots < x_n \le b$.
  - **Interpolation**:

$$f(x) = P(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{j=0}^{n} (x - x_j), \quad P(x) = \sum_{j=0}^{n} f(x_j) L_j(x).$$

  - **Approximate Integration**:

$$\begin{aligned}
\int_a^b f(x)dx &= \left(\int_a^b P(x)dx \right) + \frac{1}{(n+1)!} \int_a^b f^{(n+1)}(\xi(x)) \prod_{j=0}^{n} (x - x_j)\ dx \\
&= \left(\sum_{j=0}^{n} a_j f(x_j) \right) + \frac{1}{(n+1)!} \int_a^b f^{(n+1)}(\xi(x)) \prod_{j=0}^{n} (x - x_j)\ dx \\
&\approx \sum_{j=0}^{n} a_j f(x_j),
\end{aligned}$$

§4.3 Numerical Integration: general idea

- **Goal**: Numerical method /quadrature for approximating $\int_a^b f(x)dx$.
- **Approach**: Replacing $f(x)$ by a polynomial.
  - **Choose**: $n+1$ points $a \leq x_0 < x_1 < \cdots < x_n \leq b$.
  - **Interpolation**:

$$f(x) = P(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!} \prod_{j=0}^{n} (x - x_j), \quad P(x) = \sum_{j=0}^{n} f(x_j) L_j(x).$$

  - **Approximate Integration**:

$$\begin{aligned}
\int_a^b f(x)dx &= \left(\int_a^b P(x)dx \right) + \frac{1}{(n+1)!} \int_a^b f^{(n+1)}(\xi(x)) \prod_{j=0}^{n} (x - x_j)\; dx \\
&= \left(\sum_{j=0}^{n} a_j\, f(x_j) \right) + \frac{1}{(n+1)!} \int_a^b f^{(n+1)}(\xi(x)) \prod_{j=0}^{n} (x - x_j)\; dx \\
&\approx \sum_{j=0}^{n} a_j\, f(x_j), \quad \text{with} \quad a_j = \int_a^b L_j(x)dx = \int_a^b \prod_{k \neq j} \frac{(x - x_k)}{(x_j - x_k)} dx.
\end{aligned}$$

The Trapezoidal Rule: $n = 1$, $x_0 = a, x_1 = b, h = b - a$.

- **Linear Interpolation**:

$$
\begin{aligned}
P_1(x) &= \frac{(x - x_1)}{(x_0 - x_1)} f(x_0) + \frac{(x - x_0)}{(x_1 - x_0)} f(x_1), \\
f(x) &= P_1(x) + \frac{1}{2} f''(\xi(x))(x - x_0)(x - x_1).
\end{aligned}
$$

The Trapezoidal Rule: $n = 1$, $x_0 = a, x_1 = b, h = b - a$.

- **Linear Interpolation**:

$$
\begin{aligned}
P_1(x) &= \frac{(x - x_1)}{(x_0 - x_1)} f(x_0) + \frac{(x - x_0)}{(x_1 - x_0)} f(x_1), \\
f(x) &= P_1(x) + \frac{1}{2} f''(\xi(x))(x - x_0)(x - x_1).
\end{aligned}
$$

- **Quadrature**:

$$
\begin{aligned}
\int_a^b P_1(x) dx &= \int_a^b \left(\frac{(x - x_1)}{(x_0 - x_1)} f(x_0) + \frac{(x - x_0)}{(x_1 - x_0)} f(x_1) \right) dx \\
&= \frac{1}{2} \left(\frac{(x - x_1)^2}{(x_0 - x_1)} f(x_0) + \frac{(x - x_0)^2}{(x_1 - x_0)} f(x_1) \right)_{x_0}^{x_1} = \frac{h}{2} \left(f(x_0) + f(x_1) \right).
\end{aligned}
$$

The Trapezoidal Rule: $n = 1$, $x_0 = a, x_1 = b, h = b - a$.

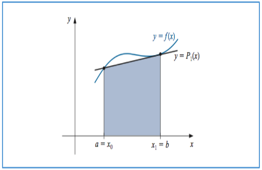
- **Linear Interpolation**:

$$
\begin{aligned}
P_1(x) &= \frac{(x-x_1)}{(x_0-x_1)} f(x_0) + \frac{(x-x_0)}{(x_1-x_0)} f(x_1), \\
f(x) &= P_1(x) + \frac{1}{2} f''(\xi(x))(x-x_0)(x-x_1).
\end{aligned}
$$

- **Quadrature**:

$$
\begin{aligned}
\int_a^b P_1(x)dx &= \int_a^b \left(\frac{(x-x_1)}{(x_0-x_1)} f(x_0) + \frac{(x-x_0)}{(x_1-x_0)} f(x_1) \right) dx \\
&= \frac{1}{2} \left(\frac{(x-x_1)^2}{(x_0-x_1)} f(x_0) + \frac{(x-x_0)^2}{(x_1-x_0)} f(x_1) \right) \Bigg|_{x_0}^{x_1} = \frac{h}{2} \left(f(x_0) + f(x_1) \right).
\end{aligned}
$$

$$
\begin{aligned}
\textbf{error} &= \int_a^b \frac{1}{2} f''(\xi(x))(x-x_0)(x-x_1)dx \\
&= \frac{f''(\xi)}{2} \int_a^b (x-x_0)(x-x_1)dx \\
&= -\frac{f''(\xi)}{12}(b-a)^3
\end{aligned}
$$



Simpson's Rule: $n = 2$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Quadratic Interpolation**:

$$P_2(x_j) = f(x_j), \quad j = 0, 1, 2.$$

$$\begin{aligned} P_2(x) &= \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} f(x_0) + \frac{(x-x_1)(x-x_0)}{(x_2-x_1)(x_2-x_0)} f(x_2) \\ &\quad + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f(x_1). \end{aligned}$$

- **Interpolation error**:

$$f(x) = P_2(x) + \frac{1}{3!} f^{(4)}(\xi(x))(x-x_0)(x-x_1)(x-x_2).$$

- **Simpson's Rule and error**:

$$\int_a^b f(x)dx = \int_a^b P_2(x)dx + \int_a^b \left(\frac{1}{3!} f^{(4)}(\xi(x))(x-x_0)(x-x_1)(x-x_2) \right) dx$$

Simpson's Rule: $n = 2$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Quadratic Interpolation**:

$$P_2(x_j) = f(x_j), \quad j = 0, 1, 2.$$

$$\begin{aligned} P_2(x) &= \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} f(x_0) + \frac{(x-x_1)(x-x_0)}{(x_2-x_1)(x_2-x_0)} f(x_2) \\ &\quad + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f(x_1). \end{aligned}$$

- **Interpolation error**:

$$f(x) = P_2(x) + \frac{1}{3!} f^{(4)}(\xi(x))(x-x_0)(x-x_1)(x-x_2).$$

- **Simpson's Rule and error**:

$$\begin{aligned} \int_a^b f(x)dx &= \int_a^b P_2(x)dx + \int_a^b \left(\frac{1}{3!} f^{(4)}(\xi(x))(x-x_0)(x-x_1)(x-x_2) \right) dx \\ &\approx \int_a^b P_2(x)dx = \frac{h}{3} \left(f(x_0) + 4f(x_1) + f(x_2) \right) \end{aligned}$$

Simpson's Rule: $n = 2$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Quadrature Rule**

$$\int_a^b f(x)dx \approx \frac{h}{3}\left(f(x_0) + 4f(x_1) + f(x_2)\right)$$

Simpson's Rule: $n = 2$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Quadrature Rule**

$$\int_a^b f(x)dx \approx \frac{h}{3}\left(f(x_0) + 4f(x_1) + f(x_2)\right)$$

- **Quadrature Error**:

$$\begin{aligned}
& \frac{1}{3!}\int_a^b f^{(3)}(\xi(x))(x-x_0)(x-x_1)(x-x_2)\,dx \\
\underline{\underline{\textbf{maybe}}} \quad & \frac{f^{(3)}(\xi)}{6}\int_a^b (x-x_0)(x-x_1)(x-x_2)\,dx \\
\underline{\underline{\textbf{???}}} \quad & 0.
\end{aligned}$$

Simpson's Rule: $n = 2$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Quadrature Rule**

$$\int_a^b f(x)dx \approx \frac{h}{3}\left(f(x_0) + 4f(x_1) + f(x_2)\right)$$

- **Quadrature Error**:

$$\begin{aligned} & \frac{1}{3!}\int_a^b f^{(3)}(\xi(x))(x-x_0)(x-x_1)(x-x_2)\,dx \\ \underline{\underline{\textbf{maybe}}} \quad & \frac{f^{(3)}(\xi)}{6}\int_a^b (x-x_0)(x-x_1)(x-x_2)\,dx \\ \underline{\underline{\textbf{???}}} \quad & 0. \end{aligned}$$

$$\boxed{\text{Error estimate wrong. Need approach better than in book.}}$$

Simpson's Rule: $n = 3$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Cubic Interpolation** with double node in x_1:

$$P_3(x_j) = f(x_j), \quad j = 0, 1, 2; \quad P_3'(x_1) = f'(x_1).$$

$$\begin{aligned} P_3(x) \;\; = \;\; & \frac{(x-x_1)^2(x-x_2)}{(x_0-x_1)^2(x_0-x_2)} f(x_0) + \frac{(x-x_1)^2(x-x_0)}{(x_2-x_1)^2(x_2-x_0)} f(x_2) \\ & + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \left(1 - \frac{(x-x_1)(2x_1-x_0-x_2)}{(x_1-x_0)(x_1-x_2)}\right) f(x_1) \\ & + \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f'(x_1). \end{aligned}$$

Simpson's Rule: $n = 3$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Cubic Interpolation** with double node in x_1:

$$P_3(x_j) = f(x_j), \quad j = 0, 1, 2; \quad P_3'(x_1) = f'(x_1).$$

$$\begin{aligned} P_3(x) &= \frac{(x-x_1)^2(x-x_2)}{(x_0-x_1)^2(x_0-x_2)} f(x_0) + \frac{(x-x_1)^2(x-x_0)}{(x_2-x_1)^2(x_2-x_0)} f(x_2) \\ &\quad + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \left(1 - \frac{(x-x_1)(2x_1-x_0-x_2)}{(x_1-x_0)(x_1-x_2)}\right) f(x_1) \\ &\quad + \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f'(x_1). \end{aligned}$$

- **Interpolation error:**

$$f(x) = P_3(x) + \frac{1}{4!} f^{(4)}(\xi(x))(x-x_0)(x-x_1)^2(x-x_2)$$

Simpson's Rule: $n = 3$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Quadrature Rule**

$$\int_a^b P_3(x)dx = f(x_0)\int_a^b \frac{(x-x_1)^2(x-x_2)}{(x_0-x_1)^2(x_0-x_2)}\,dx + f(x_2)\int_a^b \frac{(x-x_1)^2(x-x_0)}{(x_2-x_1)^2(x_2-x_0)}\,dx$$

$$+f(x_1)\int_a^b \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\left(1 - \frac{(x-x_1)(2x_1-x_0-x_2)}{(x_1-x_0)(x_1-x_2)}\right)\,dx$$

$$+f'(x_1)\int_a^b \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\,dx$$

Simpson's Rule: $n = 3$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Quadrature Rule**

$$\int_a^b P_3(x)dx = f(x_0)\int_a^b \frac{(x-x_1)^2(x-x_2)}{(x_0-x_1)^2(x_0-x_2)}\,dx + f(x_2)\int_a^b \frac{(x-x_1)^2(x-x_0)}{(x_2-x_1)^2(x_2-x_0)}\,dx$$

$$+f(x_1)\int_a^b \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\left(1 - \frac{(x-x_1)(2x_1-x_0-x_2)}{(x_1-x_0)(x_1-x_2)}\right)\,dx$$

$$+f'(x_1)\int_a^b \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\,dx$$

$$\stackrel{!!!}{=\!=} \frac{h}{3}\left(f(x_0) + 4f(x_1) + f(x_2)\right).$$

Simpson's Rule: $n = 3$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

- **Quadrature Rule**

$$\int_a^b P_3(x)dx = f(x_0)\int_a^b \frac{(x-x_1)^2(x-x_2)}{(x_0-x_1)^2(x_0-x_2)}\,dx + f(x_2)\int_a^b \frac{(x-x_1)^2(x-x_0)}{(x_2-x_1)^2(x_2-x_0)}\,dx$$

$$+f(x_1)\int_a^b \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\left(1-\frac{(x-x_1)(2x_1-x_0-x_2)}{(x_1-x_0)(x_1-x_2)}\right)\,dx$$

$$+f'(x_1)\int_a^b \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\,dx$$

$$\overset{!!!}{=\!=} \frac{h}{3}\left(f(x_0)+4f(x_1)+f(x_2)\right).$$

$\boxed{\text{Stroke of luck: } f'(x_1) \text{ does not end up in quadrature}}$

Simpson's Rule: $n = 3,\ x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b,\ h = \frac{b-a}{2}$.

- **Quadrature Rule**

$$\int_a^b P_3(x)dx = f(x_0)\int_a^b \frac{(x-x_1)^2(x-x_2)}{(x_0-x_1)^2(x_0-x_2)}\,dx + f(x_2)\int_a^b \frac{(x-x_1)^2(x-x_0)}{(x_2-x_1)^2(x_2-x_0)}\,dx$$

$$+f(x_1)\int_a^b \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\left(1 - \frac{(x-x_1)(2x_1-x_0-x_2)}{(x_1-x_0)(x_1-x_2)}\right)\,dx$$

$$+f'(x_1)\int_a^b \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\,dx$$

$$\overset{!!!}{=\!=} \frac{h}{3}\left(f(x_0) + 4f(x_1) + f(x_2)\right).$$

Stroke of luck: $f'(x_1)$ does not end up in quadrature

$$\textbf{Quadrature Error} = \frac{1}{4!}\int_a^b f^{(4)}(\xi(x))(x-x_0)(x-x_1)^2(x-x_2)\,dx$$

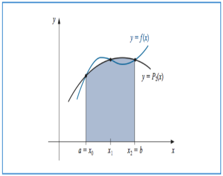
$$= \frac{f^{(4)}(\xi)}{4!}\int_a^b (x-x_0)(x-x_1)^2(x-x_2)\,dx = -\frac{f^{(4)}(\xi)}{90}\,h^5$$

Simpson's Rule: $n = 3$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

$$\int_a^b f(x)dx = \frac{h}{3}\left(f(x_0) + 4f(x_1) + f(x_2)\right) - \frac{f^{(4)}(\xi)}{90}\, h^5.$$

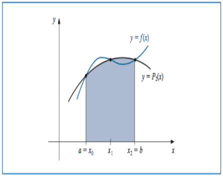
Simpson's Rule: $n = 3$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

$$\int_a^b f(x)dx = \frac{h}{3}\left(f(x_0) + 4f(x_1) + f(x_2)\right) - \frac{f^{(4)}(\xi)}{90}h^5.$$



Simpson's Rule: $n = 3$, $x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b$, $h = \frac{b-a}{2}$.

$$\int_a^b f(x)dx = \frac{h}{3}\left(f(x_0) + 4f(x_1) + f(x_2)\right) - \frac{f^{(4)}(\xi)}{90}h^5.$$



Book wrong: $f'(x_1) \neq P'(x_1)$

Correct plot: $f'(x_1) = P'(x_1)$

Example: approximate $\int_0^2 f(x)dx$: Simpson wins

	(a)	(b)	(c)	(d)	(e)	(f)
$f(x)$	x^2	x^4	$(x+1)^{-1}$	$\sqrt{1+x^2}$	$\sin x$	e^x
Exact value	2.667	6.400	1.099	2.958	1.416	6.389
Trapezoidal	4.000	16.000	1.333	3.326	0.909	8.389
Simpson's	2.667	6.667	1.111	2.964	1.425	6.421

Example: approximate $\int_0^2 f(x)dx$: Simpson wins

	(a)	(b)	(c)	(d)	(e)	(f)
$f(x)$	x^2	x^4	$(x+1)^{-1}$	$\sqrt{1+x^2}$	$\sin x$	e^x
Exact value	2.667	6.400	1.099	2.958	1.416	6.389
Trapezoidal	4.000	16.000	1.333	3.326	0.909	8.389
Simpson's	2.667	6.667	1.111	2.964	1.425	6.421

- DEFINITION — **Degree of precision (DoP)**: integer n such that quadrature formula is exact for $f(x) = x^k$, for each $k = 0, 1, \cdots, n$ but inexact for $f(x) = x^{n+1}$

Example: approximate $\int_0^2 f(x)dx$: Simpson wins

	(a)	(b)	(c)	(d)	(e)	(f)
$f(x)$	x^2	x^4	$(x+1)^{-1}$	$\sqrt{1+x^2}$	$\sin x$	e^x
Exact value	2.667	6.400	1.099	2.958	1.416	6.389
Trapezoidal	4.000	16.000	1.333	3.326	0.909	8.389
Simpson's	2.667	6.667	1.111	2.964	1.425	6.421

- DEFINITION — **Degree of precision (DoP)**: integer n such that quadrature formula is exact for $f(x) = x^k$, for each $k = 0, 1, \cdots, n$ but inexact for $f(x) = x^{n+1}$
- **Theorem**: quadrature formula is exact for $\boxed{\text{all polynomials}}$ of degree at most n.

Example: approximate $\int_0^2 f(x)dx$: Simpson wins

	(a)	(b)	(c)	(d)	(e)	(f)
$f(x)$	x^2	x^4	$(x+1)^{-1}$	$\sqrt{1+x^2}$	$\sin x$	e^x
Exact value	2.667	6.400	1.099	2.958	1.416	6.389
Trapezoidal	4.000	16.000	1.333	3.326	0.909	8.389
Simpson's	2.667	6.667	1.111	2.964	1.425	6.421

- DEFINITION — **Degree of precision (DoP)**: integer n such that quadrature formula is <u>exact</u> for $f(x) = x^k$, for each $k = 0, 1, \cdots, n$ but <u>inexact</u> for $f(x) = x^{n+1}$
- **Theorem**: quadrature formula is exact for $\boxed{\text{all polynomials}}$ of degree at most n.
- **Simplification**: only need to verify exactness on interval $[0, 1]$.

Example: approximate $\int_0^2 f(x)dx$: Simpson wins

	(a)	(b)	(c)	(d)	(e)	(f)
$f(x)$	x^2	x^4	$(x+1)^{-1}$	$\sqrt{1+x^2}$	$\sin x$	e^x
Exact value	2.667	6.400	1.099	2.958	1.416	6.389
Trapezoidal	4.000	16.000	1.333	3.326	0.909	8.389
Simpson's	2.667	6.667	1.111	2.964	1.425	6.421

- Definition — **Degree of precision (DoP)**: integer n such that quadrature formula is exact for $f(x) = x^k$, for each $k = 0, 1, \cdots, n$ but inexact for $f(x) = x^{n+1}$
- **Theorem**: quadrature formula is exact for $\boxed{\text{all polynomials}}$ of degree at most n.
- **Simplification**: only need to verify exactness on interval $[0, 1]$.

  DoP $= 1$ for Trapezoidal Rule, **DoP** $= 3$ for Simpson.