7. DIFFERENTIATION AND INTEGRATION

7.1.1 Remember: what we saw during the last lesson

Nearest point, linear, cubic interpolation successively in multiple directions.

Parametric interpolation $\mathbf{r}(t)$: linear, quadratic & cubic Bézier curves

Piecewise cubic Bézier curves and non-interpolating B-splines

7.1.2 Overview: what you will learn today

Numerical error propagation, convergence and Richardson extrapolation

Differentiation using finite differences and Richardson extrapolation

Integration Newton-Cotes quadrature (mid-point, trapezoidal, Simpson)

7.2 Reminder: error propagation (NAM 1,2, H 1.2)

Error analysis Define an approximation $\widehat{x} \approx x$ — $\widehat{x} = 3.1415 \approx \pi = x$

Absolute error $e_x = \widehat{x} - x$ — $e_x = 3.1415 - \pi = -9.27 \cdot 10^{-5}$

Relative error $r_x = (\widehat{x} - x)/x = e_x/x$ — $r_x = e_x/\pi = -2.9 \cdot 10^{-5}$

Error bounds $E_x = |e_x|, R_x = |r_x|$

Keep only relevant figures and specify the main source of uncertainty.

Error propagation through an operation

With $(+,-)$ add the absolute error — $(x \pm \Delta_x) - (y \pm \Delta_y) = (x - y) \pm (\Delta_x + \Delta_y)$

With $(\cdot, \div)$ add the relative error — $(x \pm \delta_x\%)/(y \pm \delta_y\%) = (x/y) \pm (\delta_x + \delta_y)\%$

Otherwise, estimate $[\min f(x); \max f(x)]$ from a set of samples $x \in \{x_k\}$

Example EX 8.6, 8.9

Peer Teaching (2×1 minutes to think, explain to your neighbour and vote)

> **Examination.** Estimate that 100 ± 25 students will take the exam in this course; having 4 ± 1 assistants to correct them, how many copies for each?
>
> $\nwarrow 25 \pm 6$ $\nearrow 25 \pm 13$ $\searrow 25 \pm 26$ none of these

7.3 Finite Differences (NAM 1.3, H 8.6)

Formulas can be derived from Lagrange interpolations

Alternative to using Taylor expansions on a homogeneous grid $t_i = ih$:

2-points interpolation forward (t_i, y_i),(t_{i+1}, y_{i+1})

$$p(t) = y_i \frac{t-t_{i+1}}{t_i-t_{i+1}} + y_{i+1}\frac{t-t_i}{t_{i+1}-t_i} = y_i \frac{t-t_{i+1}}{-h} + y_{i+1}\frac{t-t_i}{h}$$

$$\boxed{\Rightarrow p'(t_i) = \frac{y_{i+1}-y_i}{h}}$$

2-points interpolation backward (t_{i-1}, y_{i-1}),(t_i, y_i)...

$$\boxed{\Rightarrow p'(t_i) = \frac{y_i - y_{i-1}}{h}}$$

3-points interpolation (t_{i-1}, y_{i-1}), (t_i, y_i), (t_{i+1}, y_{i+1})

$$p(t) = y_{i-1}\frac{(t-t_i)(t-t_{i+1})}{(t_{i-1}-t_i)(t_{i-1}-t_{i+1})} + y_i \frac{(t-t_{i-1})(t-t_{i+1})}{(t_i-t_{i-1})(t_i-t_{i+1})} + y_{i+1}\frac{(t-t_{i-1})(t-t_i)}{(t_{i+1}-t_{i-1})(t_{i+1}-t_i)}$$

$$p(t) = y_{i-1}\frac{(t-t_i)(t-t_{i+1})}{2h^2} + y_i\frac{(t-t_{i-1})(t-t_{i+1})}{-h^2} + y_{i+1}\frac{(t-t_{i-1})(t-t_i)}{2h^2}$$

$$p'(t) = y_{i-1}\frac{(t-t_i)+(t-t_{i+1})}{2h^2} + y_i\frac{(t-t_{i-1})+(t-t_{i+1})}{-h^2} + y_{i+1}\frac{(t-t_{i-1})+(t-t_i)}{2h^2}$$

$$\boxed{\Rightarrow p'(t_i) = \frac{y_{i+1}-y_{i-1}}{2h}}$$

$$p''(t) = y_{i-1}\frac{2}{2h^2} + y_i\frac{2}{-h^2} + y_{i+1}\frac{2}{2h^2}$$

$$\boxed{\Rightarrow p''(t) = \frac{y_{i+1}-2y_i+y_{i-1}}{h^2}}$$

etc

7.3 Convergence rate, Richardson extrapolation (NAM 3.8, H 8.7)

Convergence rate

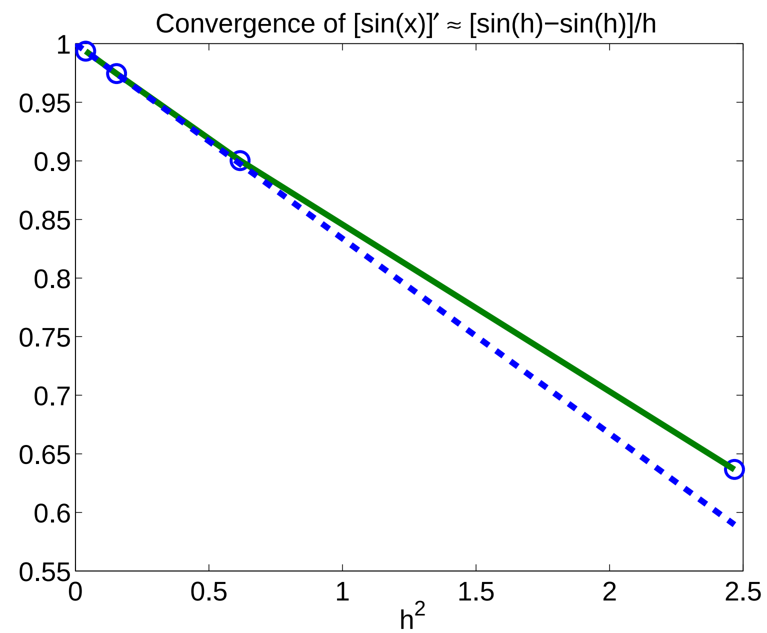
An approximation $F(h) \approx G$ often depends on a step h that can be reduced to increase the accuracy by dividing the step by an integer value Q. If $Q \to \infty$, the convergence rate $|F(h) - G| \propto h^p$ can be determined experimentally from

$$p = \ln\left(\frac{F(h) - F(h/Q)}{F(h/Q) - F(h/Q^2)}\right) \Big/ \ln(Q)$$

For example, take the centered difference

$$F(h) = \frac{\sin(h) - \sin(-h)}{2h} \approx \frac{d}{dx}\sin(x)|_{x=0}$$

```
>> n=6; k=1:n; Q=2.^k; h=pi./Q;
>> F=(sin(h)-sin(-h))./(2*h);
>> plot(h.^2,F,'o',h.^2,F), hold on;
>> plot([0 h(1)^2], [1 (F(n)-1)/h(n)^2*h(1)^2+1]
>> log( diff(F(1:n-1))./diff(F(2:n)) )/log(2)
ans =  1.8298    1.9581    1.9896    1.9974
```



which is consistent with the precision in $\mathcal{O}(h^2)$ expected for a centered finite difference scheme.

Richardson extrapolation

Assuming a leading error $\propto h^p$, the approximation is evaluated for a low resolution

$$F(h) = c_0 + c_1 h^p + \mathcal{O}(h^r)$$

and a higher resolution, where the step is divided by a integer number Q

$$F(h/Q) = c_0 + c_1 (h/Q)^p + \mathcal{O}(h^r)$$

Solve the linear system for the unknown coefficients (c_0, c_1)

$$y_1^* \equiv c_0 = F(h/Q) + \frac{F(h/Q) - F(h)}{Q^p - 1} + \mathcal{O}(h^r)$$

showing that a higher precision in $\mathcal{O}(h^r)$ can be achieved when the systematic errors $\propto h^p$ cancel out by refining the step size.

Dividing the step size further with $Q^k, k = 0, 1, 2, \ldots$ yields a series of exatrapolations $y_k^* = y_{k+1} + (y_{k+1} - y_k)/(Q^{p+k} - 1) \equiv \Delta_k / N_1$ with $N_1 = Q^p - 1$. These can in turn be used to cancel higher order errors in $\mathcal{O}(h^r)$ with $N_2 = Q^r - 1$, etc

y	Δ/N_1	$y + \Delta/N_1$	Δ/N_2	$y^* + \Delta/N_2$	Δ/N_3	$y^{**} + \Delta/N_3$
y_1						
y_2	$(y_2 - y_1)/N_1$	y_1^*				
y_3	$(y_3 - y_2)/N_1$	y_2^*	$(y_2^* - y_1^*)/N_2$	y_1^{**}		
y_4	$(y_4 - y_3)/N_1$	y_3^*	$(y_3^* - y_2^*)/N_2$	y_2^{**}	$(y_2^{**} - y_1^{**})/N_3$	y_1^{***}

For example, the approximation $F(h) = \frac{\sin(h)-\sin(-h)}{2h} \approx \frac{d}{dx}\sin(x)|_{x=0}$ yields

$$
\begin{aligned}
y_1 &= \frac{\sin(\pi/2) - \sin(-\pi/2)}{\pi} = \frac{2}{\pi} \simeq 0.6366 \\
y_2 &= \frac{\sin(\pi/4) - \sin(-\pi/4)}{\pi/2} = \frac{2\sqrt{2}}{\pi} \simeq 0.9003 \\
y_1^* &= 0.9003 + \frac{0.9003 - 0.6366}{2^2 - 1} \approx 0.9882
\end{aligned}
$$

```
>> n=4; q=2; k=1:n; Q=q.^k;
>> h=pi./Q; F=(sin(h)-sin(-h))./(2*h); % Centered differences
>> Y=zeros(n,n-1); Y(:,1)=F';
>> for p=1:n-1,
>>   D=diff(Y(:,p))./(q.^(1+p)-1);     % Start O(p+1)=O(2)
>>   Y(:,p+1)=[zeros(p,1); Y(p+1:n,p)+D(p:n-1)];
>> end
>> %       y                  y*                 y**                y***
>> %-------------------------------------------------------------------------
>> Y=Y
Y =0.63661977236758                  0                  0                  0
   0.90031631615711   0.98821516408695                  0                  0
   0.97449535840443   0.99922170582021   1.00079406892496                  0
   0.99358685114421   0.99995068205746   1.00005482151993   1.00000553835959
```

This can be compared with the theoretical value $\sin(x)'|_{x=0} = 1$ to show that the extrapolated value is obtained with 6 digits accuracy out of the original two!

7.4 Numerical Integration or Quadrature (NAM 5.1,5.2, H 8.3.1)

Newton-Cotes methods generally sum the contributions from equally spaced nodes x_i

$$I = \int_a^b f(x)dx \approx \sum_{i=1}^{N} w_i f(x_i), \qquad x_i = a + (i-1)\frac{b-a}{N-1}$$

The weights w_i are all positive and their value depends on the method being used.

Mid-point rule approximates $f(x)$ with a zero-order polynomial in the middle of $[a; b]$

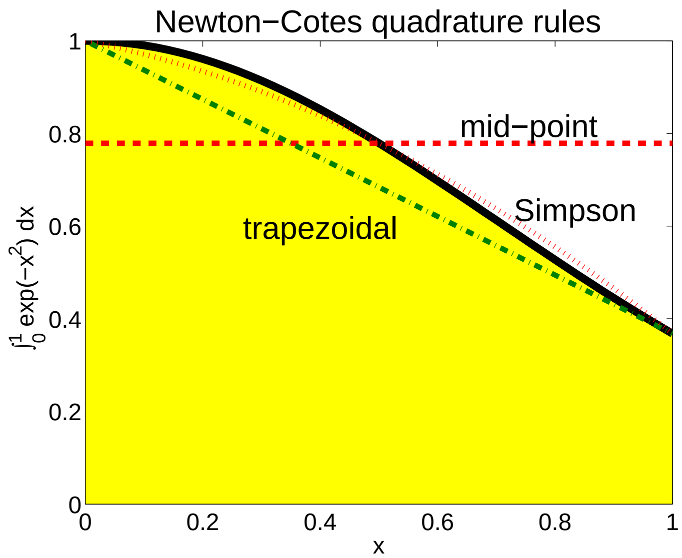
$$M = (b-a) f\left(\frac{a+b}{2}\right)$$

Trapezoidal rule approximates $f(x)$ with a 1st-order polynomial of trapezoidal shape.

$$T = \frac{b-a}{2}\left(f(a) + f(b)\right)$$

Simpson's rule takes a quadratic polynomial.

$$S = \frac{b-a}{6}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right)$$



Divide the interval of integration in smaller pieces to improve the accuracy, e.g.

$$T(h) = h\left(\sum_{i=1}^{N} f_i - \frac{f_1 + f_N}{2}\right), \qquad h = \frac{b-a}{N-1}$$

Romberg quadrature reduces the step size and then uses a Richardson extrapolation. The Simpson rule can be defined in this manner directly from the trapezoidal rule

$$S = T(h/2) + \frac{T(h/2) - T(h)}{2^2 - 1}$$

In Matlab calculate $\int_0^2 \sqrt{\frac{1}{2} + 2\exp(-x)\sin(2x^2)}\,dx$

```
function f=fq(x)                          % Function is in a separate file fq.m
  f=sqrt(0.5+2*exp(-x).*sin(2*x.^2));

>> n=40;  h=2/n; x=h*(0:n); f=fq(x); T =h*(sum(f)-0.5*(f(1)+f(n+1)));
>> n=2*n; h=2/n; x=h*(0:n); f=fq(x); T2=h*(sum(f)-0.5*(f(1)+f(n+1)));
>> n=2*n; h=2/n; x=h*(0:n); f=fq(x); T4=h*(sum(f)-0.5*(f(1)+f(n+1)));
>> S =T2+(T2-T)/3;  S2=T4+(T4-T2)/3;  B=S2+(S2-S)/15;
>> Richardson=[T 0 0; T2 S 0; T4 S2 B]

                    T                   S                   B
Richardson = 1.64200090515364                   0                   0
             1.64205315901475   1.64207057696846                   0
             1.64206616695595   1.64207050293635   1.64207049800087
```