Richardson Extrapolation

Use trapezoidal rule as an example

- subintervals: $n = 2^j = 1, 2, 4, 8, 16, \ldots.$

$$\int_a^b f(x)dx = \frac{h}{2}\left[f(x_0) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(x_n)\right] + \sum_{j=1}^{\infty} c_j h^{2j}$$

j	n	Formula
0	1	$I_0 = \frac{h}{2}\left[f(a) + f(b)\right]$
1	2	$I_1 = \frac{h}{4}\left[f(a) + 2f(x_1) + f(b)\right]$
2	4	$I_2 = \frac{h}{8}\left[f(a) + 2f(x_1) + 2f(x_2) + 2f(x_3) + f(b)\right]$
3	8	$I_3 = \frac{h}{16}\left[f(a) + 2f(x_1) + \cdots + 2f(x_7) + f(b)\right]$
$\vdots$	$\vdots$	$\vdots$
j	2^j	$I_j = \frac{h}{2^j}\left[f(a) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(b)\right]$

Richardson Extrapolation

For trapezoidal rule

$$A = \int_a^b f(x)dx = A(h) + c_1 h^2 + \cdots$$

$$\begin{cases} A = A(h) + c_1 h^2 + c_2 h^4 \cdots \\ A = A(\frac{h}{2}) + c_1(\frac{h}{2})^2 + c_2(\frac{h}{2})^4 + \cdots \end{cases}$$

$$\Rightarrow A = \frac{1}{3}\left[4A(\frac{h}{2}) - A(h)\right] - \frac{c_2}{4}h^4 + \cdots = B(h) + b_2 h^4 + \cdots$$

$$\left.\begin{cases} A = B(h) + b_2 h^4 \cdots \\ A = B(\frac{h}{2}) + b_2(\frac{h}{2})^4 + \cdots \end{cases}\right\} \Rightarrow C(h) = \frac{1}{15}\left[16B(\frac{h}{2}) - B(h)\right]$$

– k^{th} **level of extrapolation**

$$\boxed{D(h) = \frac{4^k C(h/2) - C(h)}{4^k - 1}}$$

Romberg Integration

Accelerated Trapezoid Rule

$$\boxed{I_{j,k} = \frac{4^k I_{j+1,k} - I_{j,k}}{4^k - 1}\,; \quad k = 1, 2, 3, \cdots}$$

	Trapezoid	Simpson's	Boole's		
	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$
	$O(h^2)$	$O(h^4)$	$O(h^6)$	$O(h^8)$	$O(h^{10})$
h	$I_{0,0}$	$I_{0,1}$	$I_{0,2}$	$I_{0,3}$	$I_{0,4}$
$h/2$	$I_{1,0}$	$I_{1,1}$	$I_{1,2}$	$I_{1,3}$	
$h/4$	$I_{2,0}$	$I_{2,1}$	$I_{2,2}$		
$h/8$	$I_{3,0}$	$I_{3,1}$			
$h/16$	$I_{4,0}$				
		$\dfrac{4I_{j+1,0} - I_{j,0}}{3}$	$\dfrac{16I_{j+1,1} - I_{j,1}}{15}$	$\dfrac{64I_{j+1,2} - I_{j,2}}{63}$	$\dfrac{256I_{j+1,3} - I_{j,3}}{255}$

Romberg Integration

Accelerated Trapezoid Rule

$$\boxed{I = \int_0^4 x e^{2x} dx = 5216.926477}$$

	Trapezoid	Simpson's	Boole's		
	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$
	$O(h^2)$	$O(h^4)$	$O(h^6)$	$O(h^8)$	$O(h^{10})$
$h = 4$	23847.7	8240.41	5499.68	5224.84	5216.95
$h = 2$	12142.2	5670.98	5229.14	5217.01	
$h = 1$	7288.79	5256.75	5217.20		
$h = 0.5$	5764.76	5219.68			
$h = 0.25$	5355.95				
$\varepsilon =$	-2.66%	-0.0527%	-0.0053%	-0.00168%	-0.00050%

ODE Integration

- Okay, how do we solve ODEs?
- As with quadrature, most methods are based on polynomial approximations!
- The simplest method is just the Euler method.
- We examine the Taylor Series expansion:

  $y(t+h) = y(t) + h\, y'(t) + \frac{1}{2} h^2 y''(\xi)$

- Recall that $y'(t) = f(t, y(t))$

ODE Integration

- Thus,

$$y(t+h) = y(t) + h\, f(t,y(t)) + \tfrac{1}{2}\, h^2 y''(\xi)$$

- In the Euler method we truncate after the linear term!
- Let y_k be the estimate at $y(t_k)$ and thus:

$$y^{EM}_{k+1} = y_k + h\, f(t_k,y_k)$$

- As we stated before, this rule is <u>explicit</u>, because y_{k+1} can be written as an explicit function of known quantities (t_k and y_k).

ODE Integration

- What is the error? Just subtract off the Taylor series!

$$y^{EM}_{k+1} - y(t_{k+1}) = y_k^{EM} - y(t_k)$$
$$+ h_k[f(t_k,y_k) - f(t_k,y(t_k))] - \tfrac{1}{2} h^2_k y''(\xi)$$

- The Δf term survives because there is some error from previous terms.
- Let's look at the term in brackets…

ODE Integration

$[f(t_k,y_k) - f(t_k,y(t_k))]$ Jacobian

$$= [\underbrace{f(t_k,y(t_k)) + (y_k - y(t_k))df/dy}_{\text{Expansion for f at } y_k} - f(t_k,y(t_k))]$$

$$= (y_k - y(t_k))df/dy$$

- Thus, the error at step k+1 is given by:

$$y^{EM}{}_{k+1} - y(t_{k+1}) = (y_k{}^{EM} - y(t_k))(1+h_kJ)$$
$$- \tfrac{1}{2} h_k{}^2 y''(\xi)$$

- The term $\frac{1}{2} h_k{}^2 y''(\xi)$ is the <u>local error</u> at each step.

ODE Integration

- The quantity $(1+hJ)$ is the amplification factor. Note that for all unstable ODEs this factor is greater than 1.
- The key result is that if J is very negative, then Euler's method may be unstable as well!
- If $hJ<-2$ then $|1+hJ|>1$ and our method is numerically unstable.
- Equations for which $J<<-1$ are termed stiff.

ODE Integration

- The interval of stability is given by:

$$-2 < hJ < 0$$

- If you have a stiff equation, your step size had better not be too large!
- For systems of equations, the Euler method will be stable if:

$$|1+h\lambda| < 1$$

for all eigenvalues. This is a bit more complex since λ may have an imaginary part. Thus the interval of stability is now a region. The method will be stable if all $h\lambda$ lie within a circle of radius 1 centered about $z=-1$ in the complex plane.

ODE Integration

- How do we deal with stiff problems?
- We use <u>implicit</u> methods. The simplest is the Backward Euler Method.

$$y^{BE}_{k+1} = y_k + h_k f(t_{k+1}, y_{k+1})$$

(note implicit dependence)

- A method is called <u>implicit</u> if the equation for y_{k+1} depends on a function of y_{k+1}.
- Okay what does error propagation look like in this case? We determine it the same way…

ODE Integration

- Rearranging we get:
  $y^{BE}_{k+1} - y(t_k+1) = (1/(1-hJ))(y_k - y(t_k)) + 0.5h^2 y''(\xi))$
- In this case the amplification factor is less than 1 in magnitude for all $J<0$, no matter how large h is!
- Thus, the interval of stability is:

$$hJ < 0$$

- Note that if $J>0$ the problem is still unstable.
- Implicit methods are very useful for stiff equations, but they do require the solution to a potentially nonlinear equation at every step!

Runge-Kutta Integration

- Thus far we have looked at explicit and implicit methods where the local error is $O(h^2 y'')$. We can take larger step sizes if we use higher order algorithms.
- First, let's look at <u>explicit</u> methods.
- The most common higher order methods are the Runge-Kutta techniques.
- Recall that we have the Taylor series expansion…

Runge-Kutta Integration

$$y(t_{k+1}) = y(t_k) + y'(t_k)h_k + 0.5h_k^2y''(t_k) + y'''(\xi)h_k^3/6$$

- Before in the Euler method we truncated after the linear term and had a local error which was $O(h^2)$.
- If we estimate y'', we can cause the local error to be $O(h^3)$! How can we do this?
- One approach is the 2-stage Runge-Kutta technique.

Runge-Kutta Integration

- Let:

$$K_1 = hf(t_n,y_n) \sim h\, y'(t_n)$$

$$K_2 = hf(t_n+h,y_n+K_1) \sim h\, y'(t_{n+1})$$

- Thus, $(K_2 - K_1)/h^2 \sim y''(t_n)$

  is an estimate of the second derivative.

Plugging this back into the Taylor series leads to the formula:

$$y(t_n+h) = y(t_n) + h\, y'(t_n) + 0.5h^2 y''(t_n) + O(h^3 y''')$$

Runge-Kutta Integration

- So, $y(t_{n+1}) = y_n + K_1 + 0.5(K_2 - K_1) + O(h^3 y''')$

$$= y_n + 0.5(K_1 - K_2) + O(h^3 y''')$$

2-stage R-K formula

(it's called 2-stage because there are 2 function evaluations)

- The <u>local error</u> is $O(h^3)$ the number of steps is $n \sim (b-a)/h$, thus this rule is <u>second order</u>.
- The Runge-Kutta rule is more accurate than the Euler method, but it is still explicit. It will run into trouble for very stiff equations.

Runge-Kutta Integration

- Usually codes don't use 2-stage R-K rules, but rather 4-stage rules:

$$K_1 = h\, f(t_n, y_n)$$

$$K_2 = h\, f(t_n + 0.5h, y_n + 0.5K_1)$$

$$K_3 = h\, f(t_n + 0.5h, y_n + 0.5K_2)$$

$$K_4 = h\, f(t_n + h, y_n + K_3)$$

$$y_{n+1} = y_n + (1/6)(K_1 + 2K_2 + 2K_3 + K_4)$$

- The local error is $O(h^5)$ and the overall rule is fourth order in the step size!

Runge-Kutta Integration

- It is interesting to note that the R-K rules are not unique!
- In particular, Fehlberg came up with a 6-stage R-K technique which had an error that was 5th order and then shows that 4 of these could be recombined to get a 4th order rule.
- This is very useful because the difference between the two estimates gives an error estimate which can be used in adaptive step size control.
- This Fehlberg R-K technique is implemented in the Matlab routine “ode45.m”

Q1: Which one of the three ODE problems is NONLINEAR?

A

$$\frac{d^3 y}{dt^3} = \frac{dy}{dt} + xy^2,$$

B

$$e^t \frac{d^2 y}{dt^2} + 3\frac{dy}{dt} + t^3 \sin t + ty = 0.$$

C

$$\frac{dy}{dt} = 3y + z + 1,$$

$$\frac{dz}{dt} = 4z + 3,$$

Q2: Which finite difference approx is called FORWARD difference?

A

$$f'(x_i) \cong \frac{f(x_{i+1}) - f(x_i)}{h},$$

B

$$f'(x_i) \cong \frac{f(x_i) - f(x_{i-1})}{h}.$$

C

$$f'(x_i) \cong \frac{f(x_{i+1}) - f(x_{i-1})}{2h}.$$

Q3: Which finite difference approx is EXPLICIT?

A

$$f'(x_i) \cong \frac{f(x_{i+1}) - f(x_i)}{h},$$

B

$$f'(x_i) \cong \frac{f(x_i) - f(x_{i-1})}{h}.$$

C

$$f'(x_i) \cong \frac{f(x_{i+1}) - f(x_{i-1})}{2h}.$$

Q4: Which finite difference approx is MOST ACCURATE?

A

$$f'(x_i) \cong \frac{f(x_{i+1}) - f(x_i)}{h},$$

B

$$f'(x_i) \cong \frac{f(x_i) - f(x_{i-1})}{h}.$$

C

$$f'(x_i) \cong \frac{f(x_{i+1}) - f(x_{i-1})}{2h}.$$