Lecture 16

Last time we studied iterative methods to solve linear equations

All iteration schemes ~~are~~ to solve $Ax = b$ are based on the notion of approximate inverse, by this we mean any matrix C for which $\|I - CA\| < 1$

iteration function

$$g(x) = C\bar{b} + (I - CA)\bar{x}$$
$$= x + C(\bar{b} - A\bar{x})$$

$$x^{(m+1)} = x^{(m)} + C(b - Ax^{(m)})$$

$$A = \hat{L} + \hat{D} + \hat{U}$$

$\hat{L}$: lower triangular part

$\hat{D}$: diagonal

$\hat{U}$: upper triangular part

Jacobi-iteration

$$C = \hat{D}^{-1}$$

$$x_i^{(m+1)} = \frac{b_i - \sum_{j \neq i} a_{ij} x_i^{(m)}}{a_{ii}}$$

for Gauss-Seidel iteration $C = (\hat{L} + \hat{D})^{-1}$

$$x_i^{(m+1)} = \frac{-\sum_{j<i} a_{ij} x_j^{(m+1)} - \sum_{j>i} a_{ij} x_j^{(m)} + b_i}{a_{ii}}$$

Iterative methods are usually applied to large linear systems with sparse co-efficient matrix.

Numerical methods to solve initial value differential equations

$$\begin{cases} \frac{dy}{dx} = f(x,y) \\ y(x_0) = y_0 \end{cases} \qquad x_0 \le x \le x_n$$

$$x_i = x_0 + i \cdot h$$

Cannot solve it exactly for all f

We had two methods

1) Taylor's algorithms of order k

difficult for general f if $k \geq 2$.

2. $k=1$ Euler method

$$y_{n+1} = y_n + h f(x_n, y_n)$$

Euler method does have significant error. However it is Numerically stable.

Error estimates and convergence of Euler's method

To solve the d.e.

$$y' = f(x,y)$$
$$y(x_0) = y_0$$

by Euler's method, we choose a constant step-size h and we apply the formula

$$(*) \quad y_{n+1} = y_n + h f(x_n, y_n) \qquad n = 0, 1, 2, \ldots$$

where $x_n = x_0 + nh$.

We denote the true solution of the d.e. at $x = x_n$ by $y(x_n)$ and the approximate solution obtained by applying $(*)$ as y_n.

We wish to estimate the magnitude of the discretization error e_n defined by

$$e_n = y(x_n) - y_n$$

Assuming that appropriate derivatives exist we can expand $y(x_{n+1})$ about $x = x_n$ using Taylors theorem with reminder

$$(**) \quad y(x_{n+1}) = y(x_n) + h\,y'(x_n) + \frac{h^2}{2} y''(\xi_n)$$

$$x_n \le \xi \le x_{n+1}$$

$$E_n = \frac{h^2}{2} y''(\xi_n) \longrightarrow \text{local discretization error}$$

This is the error committed in the single step from x_n to x_{n+1} assuming that y and y' are known exactly at $x = x_n$

(we ignore Round-off error in this section)

On subtracting (*) from (**) we obtain

$$e_{n+1} = e_n + h\left[f(x_n, y(x_n)) - f(x_n, y_n)\right] + \frac{h^2}{2} y''(\xi_n)$$

By M.V.T

$$f(x_n, y(x_n)) - f(x_n, y_n) = \frac{\partial f}{\partial y}(x_n, \bar{y}_n)\,(y(x_n) - y_n)$$

$$= f_y(x_n, \bar{y}_n)\, e_n$$

($\bar{y}_n$ between y_n and $y(x_n)$)

So we have

$$\boxed{e_{n+1} = e_n + h f_y(x_n, \bar{y}_n) e_n + \frac{h^2}{2} y''(\xi_n)}$$

Theorem Let y_n be the approximate solution (by Euler method) of

$$\frac{dy}{dx} = f(x,y), \qquad y(x_0) = y_0$$

If the exact solution $y(x)$ of above d.e has a continuous second derivative on the interval $[a,b]$ and if in this interval the inequalities

$$|f_y(x,y)| \le L, \qquad |y''(x)| \le Y$$

are satisfied for fixed true constants L and Y.

Then the error $e_n = y(x_n) - y_n$ of Euler's Method at a point $x_n = x_0 + nh$ is bounded as

$$|e_n| \le \frac{hY}{2L}\left(e^{(x_n - x_0)L} - 1\right)$$

Proof $e_{n+1} = e_n + h f_y(x_n, \bar{y}_n) e_n + \frac{h^2}{2} y''(\xi_n)$

$$|e_{n+1}| \le |e_n| + hL|e_n| + \frac{h^2}{2}Y$$

$$= (1+hL)|e_n| + \frac{h^2}{2}Y$$

We show by induction that the solution of the difference equation

$$\varepsilon_{n+1} = (1+hL)\varepsilon_n + \frac{h^2}{2}Y, \qquad \varepsilon_0 = 0$$

dominates $|e_n|$, i.e we will show

(*). $\varepsilon_n \ge |e_n|$ for $n = 0, 1, 2, \ldots$

Since $e_0 = \varepsilon_0 = 0$ (*) is true for $n = 0$.

Assume (*) is true for n

i.e. $\varepsilon_n \ge |e_n|$

$$\varepsilon_{n+1} = (1+hL)\varepsilon_n + \frac{h^2}{2}Y$$

$$\geq (1+hL)|e_n| + \frac{h^2}{2}Y$$

$$\geq |e_{n+1}|$$

Thus by induction $\varepsilon_n \geq |e_n| \quad \forall n = 0, 1, 2, \ldots$

$$\varepsilon_{n+1} = (1+hL)\varepsilon_n + \frac{h^2}{2}Y$$

$$\varepsilon_0 = 0$$

Soln $\quad \varepsilon_n = C(1+hL)^n - B$

where $B = \frac{hY}{2L}$ and C is a constant to be determined

$$\varepsilon_0 = 0$$

so $C = B$

$$\varepsilon_n = B(1+hL)^n - B$$

$$e^x \geq 1+x$$

so $\quad 1 + hL \leq e^{hL}$

$$(1+hL)^n \leq e^{nhL}$$

Therefore $\varepsilon_n \leq B(e^{nhL} - 1)$

$$= \frac{hY}{2L}(e^{nhL} - 1)$$

$$= \frac{hY}{2L}(e^{(x_n - x_0)L} - 1)$$

$(\because nh = x_n - x_0)$

Since $|e_n| \leq \varepsilon_n$

we have proved the theorem.

Runge-Kutta Methods

Short form R-K method

Euler's method is not very useful in practical problems because it requires a very small step size for reasonable accuracy.

Taylor algorithm of higher order is unacceptable as a general purpose

procedure because of the need to obtain higher total derivative of $y(x)$.

RK method attempt to obtain greater accuracy and at the same time avoid the need for higher derivatives by evaluating $f(x,y)$ at selected points at each sub-interval

---

RK method of order 2

We choose a formula of the following form

$$y_{n+1} = y_n + a k_1 + b k_2 \quad (**)$$

$$k_1 = h f(x_n, y_n)$$

$$k_2 = h f(x_n + \alpha h, y_n + \beta k_1)$$

a, b, α, β are constants to be determined so that $(*)$ will agree with a

Taylor algorithim of as high as order as possible

On expanding $y(x_{n+1})$ in a Taylor series through terms of order h^3 we obtain

$$(**)\quad y(x_{n+1}) = y(x_n) + hy'(x_n) + \frac{h^2}{2}y''(x_n) + \frac{h^3}{6}y'''(x_n) + O(h^4)$$

$$= y(x_n) + hf(x_n, y_n) + \frac{h^2}{2}(f_x + ff_y)_n + \frac{h^3}{6}(f_{xx} + 2ff_{xy} + f_{yy}f^2 + f_x f_y + f_y^2 f)_n + O(h^4)$$

By Taylor expresion of function of two variable we get that

$$\frac{K_2}{h} = f(x_n + \alpha h, y_n + \beta K_1)$$

$$= f(x_n, y_n) + \alpha h f_x + \beta K_1 f_y + \frac{\alpha^2 h^2}{2} f_{xx} + \alpha h \beta K_1 f_{xy} + \frac{\beta^2 K_1^2}{2} f_{yy} + O(h^3)$$

We plug value of k_2 in (*)

$$y_{n+1} = y_n + ak_1 + bk_2$$

$$= y_n + ahf(x_n, y_n) + bk_2$$

$$y_{n+1} = y_n + (a+b)hf + bh^2(\alpha f_x + \beta f f_y) + bh^3\left(\frac{\alpha^2}{2} f_{xx} + \alpha\beta f f_{xy} + \frac{\beta^2}{2} f^2 f_{yy}\right) + O(h^4)$$

Comparing above equation with (**) we get that to make corresponding power of h and h^2 to agree we must have

$$\left.\begin{aligned} a+b &= 1 \\ b\alpha &= b\beta = \frac{1}{2} \end{aligned}\right\} (***)$$

4 unknown, 3 equations. So we have one degree of freedom in (***)

Simplest solution

$$a = b = \frac{1}{2}$$

$$\alpha = \beta = 1.$$

RK method of order 2

Algorithm

$$\frac{dy}{dx} = f(x,y)$$

$$y(x_0) = y_0$$

generate approximations y_n to $y(x_0+nh)$ for h fixed and $n = 0,1,2,\ldots$ using the formula

$$y_{n+1} = y_n + \frac{1}{2}(k_1+k_2)$$

$$k_1 = hf(x_n, y_n)$$

$$k_2 = hf(x_n+h, y_n+k_1)$$

Local error of RK-method of order 2

$$y(x_{n+1}) - y_{n+1} = \frac{h^3}{12}\left(f_{xx} + 2ff_{xy} + f^2f_{yy} - 2f_xf_y - 2ff_y^2\right) + O(h^4)$$

Thus local error is $O(h^3)$ for RK-method of order 2.

Recall local error in Euler method is $O(h^2)$

Example

$$\frac{dy}{dx} = y - x^2 + 1 \qquad 0 \le x \le 2$$

$$y(0) = 0.5$$

Exact ans $\quad y(x) = (x+1)^2 - \frac{1}{2} e^x$

$h = 0.2$.

x_i	Exact $y(x_i)$	RK ord 2 y_i	Error
0	0.5	0.5	0
0.2	0.8292986	0.826600	0.0032986
0.4	1.2140877	1.20692	0.0071677
0.6	1.6489406	1.63774 24	0.0116982
0.8	2.1272295	2.1102357	0.0169938
1.0	2.6408591	2.617687 6	0.0231715
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
2.0	5.3054720	5.2330546	0.0724123

RK method of order 4

$$\frac{dy}{dx} = f(x,y) \qquad a = x_0 \le x \le b$$

$$y(x_0) = y_0$$

generate approximation y_n to $y(x_0 + nh)$ for h fixed and $n = 0, 1, 2, \ldots$ using the formula

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1 = h f(x_n, y_n)$$

$$k_2 = h f(x_n + \frac{h}{2}, y_n + \frac{1}{2}k_1)$$

$$k_3 = h f(x_n + \frac{h}{2}, y_n + \frac{1}{2}k_2)$$

$$k_4 = h f(x_n + h, y_n + k_3)$$

Local error is of $O(h^5)$

again the price we pay for the fantastic discretization error is that four function evaluations are required ~~at~~ per step.

Remark

Suppose we are solving

$$\frac{dy}{dx} = f(x,y)$$

$$y(x_0) = y_0 \qquad a = x_0 \le x \le b.$$

(*) $\begin{cases} \text{Euler's method with } h = 0.025 \\ \text{RK. (order 2)} \quad \text{"} \quad h = 0.05 \\ \text{RK (order 4)} \quad \text{"} \quad \text{"} \quad h = 0.1 \end{cases}$

Then most of the time RK-method of order 4 will give more accurate results.

Note that the # function evaluation in all three methods in (*) is the same.

Multi-step formulas

Euler's method, Taylor's method of order k and RK-methods are one-step methods. They require information about the solution at a single point $x = x_n$ from which the method proceed to obtain y at the next point $x = x_{n+1}$.

Multistep methods make use of information about the solution at more than one point

Let us assume we already obtained approximations to y at a number of equally spaced points say $x_0, x_1, \ldots, x_n$

One class of multistep methods is based on the principle of numerical integration.

If we integrate the differential equation $y' = f(x, y)$ from x_n to x_{n+1} we will have

$$\int_{x_{n+0}}^{x_{n+1}} y' dx = \int_{x_n}^{x_{n+1}} f(x, y(x)) \, dx$$

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} f(x, y(x)) \, dx.$$

To carry out the integration we now approximate $f(x, y(x))$ by a polynomial which interpolates $f(x, y(x))$ at the $m+1$ pts

$$x_n, x_{n-1}, x_{n-2}, \ldots, x_{n-m}.$$

This is conveniently done by

Newton's backward difference formula.

Digression

Newton's forward difference formula and Newton's backward difference formula

Newton's divided difference formula.

$$P_k(x) = f(x_0) + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1)$$
$$+ \cdots + f[x_0, x_1, \ldots, x_n](x - x_0) \cdots (x - x_{n-1})$$

Newton's divided-diff formula can be expressed in a simplified form when $x_0, x_1, \ldots, x_n$ are arranged consecutively with equal spacing.

Set $h = x_{i+1} - x_i \qquad i = 0, 1, 2, \ldots, n-1$

$x = x_0 + sh$

Note $x_i = x_0 + ih$

So $x - x_i = (s - i)h$

$$P_n(x) = P_n(x_0 + sh)$$

$$= f[x_0] + sh\, f[x_0, x_1] + \tfrac{1}{2} s(s-1) h^2 f[x_0, x_1, x_2]$$

$$+ \cdots + s(s-1)(s-2)\cdots(s-n+1) h^n f[x_0, \ldots, x_n]$$

We use binomial coefficients

$$\binom{s}{k} = \frac{s(s-1)\cdots(s-k+1)}{k!}$$

$$\text{So } P_n(x) = f[x_0] + \sum_{k=1}^{n} \binom{s}{k} k!\, h^k f[x_0, x_1, \ldots, x_k]$$

Forward difference operator

$$\Delta f(x_i) = f(x_{i+1}) - f(x_i)$$

$$\Delta^2 f(x_i) = \Delta(\Delta f(x_i))$$

$$= \Delta(f(x_{i+1}) - f(x_i))$$

$$= f(x_{i+2}) - 2f(x_{i+1}) + f(x_i)$$

and so on

$$f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

$$= \frac{1}{h} \Delta f(x_0)$$

$$f[x_0, x_1, x_2] = \frac{1}{2h}\left[\frac{\Delta f(x_1) - \Delta f(x_0)}{h}\right]$$

$$= \frac{1}{2h^2}\,\Delta^2 f(x_0)$$

In general

$$f[x_0, x_1, \ldots, x_k] = \frac{1}{k!\,h^k}\,\Delta^k f(x_0)$$

Thus we have

$$\boxed{P_n(x) = f(x_0) + \sum_{k=1}^{n} \binom{s}{k} \Delta^k f(x_0)}$$

This is called Newton forward difference formula

If the interpolating nodes are re-ordered from last to first as $x_n, x_{n-1}, \ldots, x_1, x_0$ we can write the interpolatory formula as

$$P_n(x) = f(x_n) + f[x_n, x_{n-1}](x - x_n) + f[x_n, x_{n-1}, x_{n-2}](x - x_n)(x - x_{n-1}) + \cdots \vdots + f[x_n, x_{n-1}, \ldots, x_0](x - x_n)(x - x_{n-1}) \cdots (x - x_1)$$

If in addition nodes are equally spaced

then $x = x_n + sh$

$$x = x_i + (s + n - i)h$$

$$P_n(x) = P_n(x_n + sh)$$

$$= f(x_n) + sh\, f[x_n, x_{n-1}] + s(s+1)h^2 f[x_n, x_{n-1}, x_{n-2}] + \cdots + s(s+1)\cdots(s+n-1)h^n f[x_n, x_{n-1}, \ldots, x_1]$$

Backwards diff operator ∇

Given a sequence $\{P_n\}_{n\geq 0}$ define the backward difference ∇P_n (read nabla P_n) by

$$\nabla P_n = P_n - P_{n-1} \quad \text{for } n \geq 1$$

$$\nabla^2 P_n = \nabla(\nabla P_n)$$

$$= \nabla(P_n - P_{n-1})$$

$$= \nabla P_n - \nabla P_{n-1}$$

$$= P_n - 2P_{n-1} + P_{n-2}$$

$$\vdots$$

$$\nabla^k P_n = \nabla(\nabla^{k-1} P_n) \quad \text{for } k \geq 2.$$

$$f[x_n, x_{n-1}] = \frac{1}{h} \nabla f(x_n)$$

$$f[x_n, x_{n-1}, x_{n-2}] = \frac{1}{2!h^2} \nabla^2 f(x_n)$$

$$\vdots$$

$$f[x_n, x_{n-1}, \ldots, x_{n-k}] = \frac{1}{k!\, h^k} \nabla^k f(x_n)$$

Consequently

$$P_n(x) = f(x_n) + s\nabla f(x_n) + \frac{s(s+1)}{2}\nabla^2 f(x_n)$$

$$+ \cdots + \frac{s(s+1)\cdots(s+(n-1))}{n!}\nabla^n f(x_n)$$

We extend binomial coeff notation

$$\binom{-s}{k} = \frac{-s(-s-1)\cdots(-s-k+1)}{k!}$$

$$= (-1)^k \frac{s(s+1)\cdots(s+k-1)}{k!}$$

So we have

$$P_n(x) = f(x_n) + (-1)\binom{-s}{1}\nabla f(x_n)$$

$$+ (-1)^2\binom{-s}{2}\nabla^2 f(x)$$

$$\vdots$$

$$+ (-1)^n\binom{-s}{n}\nabla^n f(x_n)$$

Newton's backward difference formula

$$P_n(x) = f(x_n) + \sum_{k=1}^{n} (-1)^k \binom{-s}{k}\nabla^k f(x_n)$$