97.5

BLUE BOOK

학 과	원자핵공학과
이 름	이 원 재
학 번	2003 - 12491
과 목	수치해석 기초
날 짜	2008. 12. 6.

※ 학칙을 위반하거나 학생의 본분에 어긋난 행위를 하였을 때에는 징계될 수 있습니다.

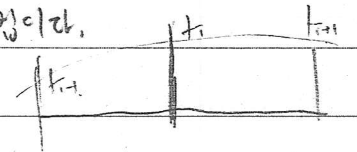
7 Pages

College of Engineering
Seoul National University

1.(a) Simpson's Method (2nd-order Interpolation)

19.8

먼저 구간을 짝수개의 작은 구간으로 나눈뒤 소구간 2개의 3점을 사용하여 2차 라그랑지 polynomial을 구한뒤, 그것을 적분하여 $I_k^{(2)}$로 삼은뒤 전체 구간에 걸쳐 I_k를 summation하는 방법이다.



$$P_2(x) = f_{i-1}\frac{(x-x_i)(x-x_{i+1})}{(x_{i-1}-x_i)(x_{i-1}-x_{i+1})} + f_i\frac{(x-x_{i-1})(x-x_{i+1})}{(x_i-x_{i-1})(x_i-x_{i+1})} + f_{i+1}\frac{(x-x_{i-1})(x-x_i)}{(x_{i+1}-x_{i-1})(x_{i+1}-x_i)}$$

x_{i-1}, x_i, x_{i+1}

$-h$, 0, h

$x_{i+1}=h$ $x_i=0$, $x_{i-1}=-h$ 이면

$$P_2(x) = f_{i-1}\frac{(x)(x-h)}{(-h)(-2h)} + f_i\frac{(x+h)(x-h)}{(h)(-h)} + f_{i+1}\frac{(x+h)(x)}{(2h)(h)}$$

$\frac{x}{h}=\tau$ 치환

$$P_2(\tau) = f_{i-1}\cdot\frac{1}{2}\tau(\tau-1) - f_i(\tau+1)(\tau-1) + f_{i+1}\cdot\frac{1}{2}\tau(\tau+1)$$

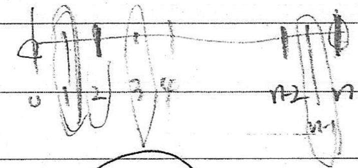
구간 내에서 적분하면 $-h<x<h$, $-1<\tau<1$ $dx=h\,d\tau$.

$$I_i^{(2)} = \int_{-h}^{h} P_2(x)\,dx = h\int_{-1}^{1}P_2(\tau)\,d\tau = h\int_{-1}^{1}\left[f_{i-1}\left(\tfrac{1}{2}\tau^2-\tfrac{1}{2}\tau\right) - f_i(\tau^2-1) + f_{i+1}\left(\tfrac{1}{2}\tau^2+\tfrac{1}{2}\tau\right)\right]$$

$$=2h\left[f_{i-1}\left(\tfrac{1}{6}\tau^3\right) - f_i\left(\tfrac{1}{3}\tau^3-\tau\right) + f_{i+1}\left(\tfrac{1}{6}\tau^3\right)\right]_0^1$$

$$=2h\left[\tfrac{1}{6}f_{i-1} + \tfrac{2}{3}f_i + \tfrac{1}{6}f_{i+1}\right] = h\left[\tfrac{1}{3}f_{i-1} + \tfrac{4}{3}f_i + \tfrac{1}{3}f_{i+1}\right].$$

전체구간 에대한 sum은



$$I = \sum_{j=1}^{n/2} I_{2j-1}^{(2)}$$

$$= \sum_{j=1}^{n/2} h\left[\tfrac{1}{3}f_{2j-2} + \tfrac{4}{3}f_{2j-1} + \tfrac{1}{3}f_{2j}\right]$$

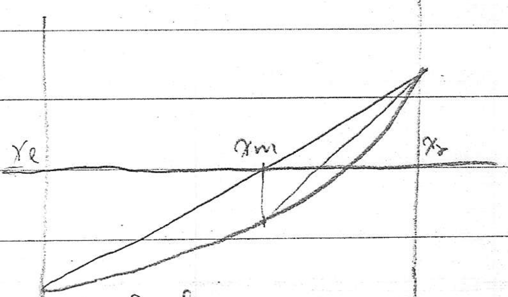
$$= \frac{h}{3}\left[f_0+f_n + 4\sum_{j=1}^{n/2}f_{2j-1} + 2\sum_{j=1}^{n/2-1}f_{2j}\right]$$

x!

11

(b) 원래 Interpolation method는, 구간 $[x_l, x_r]$ 사이에 근이 있을때

$$x_m = x_l - \frac{x_r-x_l}{f_r-f_l}f_l$$ 을 취해서

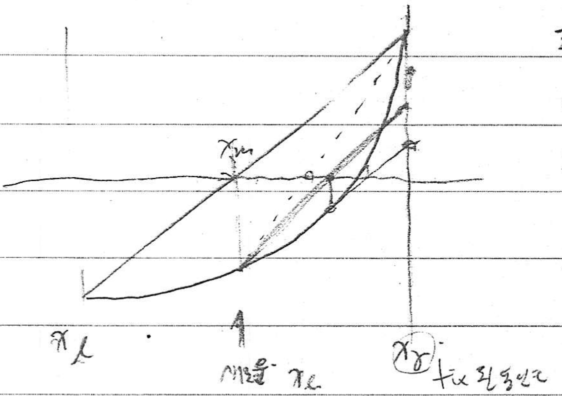


x_m이 새로운 x_l이 될지 x_r이 될지를 if($f_m\cdot f_l<0$) 으로 판단 해서 구간을 좁혀가는 방법이다.

$$(\cancel{x}-f_l) = \frac{f_r-f_l}{x_r-x_l}(x_m-x_l)$$

이 방법을 발전시켜 fix된 포인트의 높이를

반으로 줄여서 linear Interpolation을 하면 더욱 빠르게 근을 구할수 있는데
이 방법이 Modified Linear Interpolation method 이다.



좌측의 그림의경우 fix 된 포인트인 f_r 을 $f_r \leftarrow 0.5 f_r$ 한후에 linear Interpolate 하여 더욱 빨리 근에 접근 한 것을 알수있다. 만약 interpolate으로 얻은 점 x_m 이 근을 지나쳤는데 ($f_m * f_p$) 원래의 normal Interpolation 방법을 사용하면 된다. 10

(c) Implicit Euler method 는 1계 미분방정식 $y' = f(x,y)$를 풀때

$y_{n+1} = y_n + h y'_{n+1} = y_n + h f(x_{n+1}, y_{n+1})$ 의 방법을 사용하는데 반해

Explicit Euler method는 $y_{n+1} = y_n + h y_n' = y_n + h f(x_n, y_n)$ 을 사용한다.

Implicit Euler method를 사용하면 stability가 좋게 할수 있는데 그 예로서

$\frac{dy}{dt} = -\alpha y$: $y(t) = y_0 e^{-\alpha y}$ 를 생각하면. Euler method는

$y_{n+1} = y_n - \alpha h y_n \rightarrow \frac{y_{n+1}}{y_n} = 1 - \alpha h$ 로서. $h > \frac{1}{\alpha}$ 이면 unstable 하는

반면에 Implicit Euler method는 $y_{n+1} = y_n - \alpha h y_{n+1} \Rightarrow \frac{y_{n+1}}{y_n} = \frac{1}{1+\alpha h}$ 로

unconditionally stable 한 장점이 있다. 하지만 $T' = -\alpha(T^4 - T_a^4)$ 와 같을경우

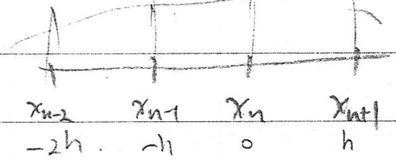
or linear system

Implicit Euler method는 $T_{n+1} = T_n - \alpha h (T_{n+1}^4 - T_a^4)$ 의 nonlinear

equation을 풀어야 한다는 단점이 있다. 오차는 local 하게 $O(h^2)$으로 같다. 9.5

(d) Adams-Bashforth-Moulton method는 Implicit Multipoint method 로서

$(x_{n-2}, f_{n-2}), (x_{n-1}, f_{n-1}), (x_n, f_n), (x_{n+1}, f_{n+1})$ 의 4점을 이용하여 3차 라그랑지 polynomial를 구한뒤



$$P_3(x) = \sum_{i=-2}^{1} f_{n+i} \prod_{\substack{j=-2 \\ j \ne i}}^{1} \frac{(x - x_{n+j})}{(x_{n+i} - x_{n+j})}$$

구간 $[x_n, x_{n+1}]$ 에서 평균기울기를 구하는 방법이다. 9 +1

$$\langle y_n' \rangle = \frac{1}{h}\int_0^h P_3(x)\,dx = \frac{1}{24}(9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2})$$

$$y_{n+1} = y_n + h\langle y_n' \rangle = y_n + \frac{h}{24}(9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2})$$

How do you ~~remember~~ ~~recall~~ remember all these?

여기서 f_{n+1} 은 $(f_{n-3}, f_{n-2}, f_{n-1}, f_n)$을 사용하여 Adams-Bashforth method로 구한다.

따라서 ~~적어도~~ 앞의 4점이 필요한데 실제로 활용할때는 초기 4점까지 single point

method를 사용하고 그 이후부터 Adams-Bashforth-Moulton method로 사용하면 된다.

Adams-bashforth 먼저, then Moulton

(e) Multistep method는 구간 내에서 기울기를 여러번 평가한뒤 가중평균하여 기울기를 얻는 방법이다. 따라서 기울기함수를 계산할 때 시간이 오래걸리는 경우에는 multipoint method (기울기 한번만 평가, Adams-Bashforth-Moulton은 두번) 에 비해 더 많은 계산을 하게 되는 단점이 있다. 하지만 multipoint method는 주로 지나간 point 들을 사용하여 extrapolation으로 구하므로 미래를 예측하기가 어려운 반면에 Multipoint Method는 앞에 올 기울기를 여러번 구해 가중평균해 얻기 때문에, 오차가 더 적다는 장점이 있다.

10

2. (a)

$$Ax = \lambda S x$$

$$\frac{1}{\lambda} x = A^{-1} S x \longrightarrow$$ 이걸 regular power method로 풀면 $\hat{\lambda} = \frac{1}{\lambda}$ 로부터 가장 작은 λ 구함

19.2

$$x^{(k)} = A^{-1} S \hat{x}^{(k-1)} \xrightarrow{k가 커지면} x^{(k)} \approx \lambda^{(k)} \hat{x}^{(k-1)}$$

$$\hat{\lambda}^{(k)} = \frac{\langle x^{(k)}, x^{(k)} \rangle}{\langle x^{(k)}, \hat{x}^{(k-1)} \rangle} \; ; \; \lambda^{(k)} = \frac{\langle x^{(k)}, \hat{x}^{(k-1)} \rangle}{\langle x^{(k)}, x^{(k)} \rangle}$$

$$\hat{x}^{(k)} = \frac{x^{(k)}}{\hat{\lambda}^{(k)}} = x^{(k)} \lambda^{(k)}$$

[초기값 설정]

while (<didn't convergent>)

$$\hat{x}^{(k-1)} = x^{(k-1)} \cdot \lambda^{(k-1)} \leftarrow \langle \text{Scaling} \rangle$$

$$x^{(k)} = A^{-1} S \hat{x}^{(k-1)} \leftarrow x^{(k)} \text{ 구하기}$$

$$\lambda^{(k)} = \frac{\langle x^{(k)}, \hat{x}^{(k-1)} \rangle}{\langle x^{(k)}, x^{(k)} \rangle}$$

10

[Convergence 확인] $\left(\frac{\| x^{(k)} - x^{(k-1)} \|}{\| x^{(k)} \|} < \varepsilon \right)$

$$\lambda^{(k-1)} \leftarrow \lambda^{(k)}$$

$$x^{(k-1)} \leftarrow x^{(k)}$$

end.

(b)

$x^{(k)} = A^{-1} S \hat{x}^{(k-1)} \longrightarrow A x^{(k)} = S \hat{x}^{(k-1)}$ 이렇게 바꾸면 matrix inverse A^{-1} 을 구하지 않아도 되는 장점이 있다. $Ax = b$ 꼴의 linear system을 직접법 (LU)이나 간접법 (Gauss-Seidel 등) 으로 풀수 있는 장점이 있다.

Inverse Power method

[초기값 할당. $\lambda_0 \leftarrow$ 초기값. $x_0 \leftarrow$ 초기벡터]

while (≪ did not convergent ≫)

$\hat{x}^{(k-1)} = x^{(k-1)} \cdot \lambda^{(k-1)}$

solve $A x^{(k)} = S \hat{x}^{(k-1)}$

$$\lambda^{(k)} = \frac{\langle x^{(k)}, \hat{x}^{(k-1)} \rangle}{\langle x^{(k)}, x^{(k)} \rangle}$$

10

[convergence 확인]

$\lambda^{(k-1)} \Leftarrow \lambda^{(k)}$; $x^{(k-1)} \Leftarrow x^{(k)}$

end

$(k-1) \to 0$

(c)

$$x^{(k)} = A \hat{x}^{(k-1)} = \frac{1}{\lambda_{(k-1)}} A x^{(k-1)} = \frac{1}{\prod \lambda_T} A^k x^{(0)}$$

$(x^{(0)} = c_1 u_1 + c_2 u_2 + \cdots + c_n u_n)$ $\qquad \lambda_1 > \lambda_2 > \lambda_3 \cdots$

$$= \frac{1}{\prod \lambda_T} \left(c_1 \lambda_1^k u_1 + c_2 \lambda_2^k u_2 + \cdots + c_n \lambda_n^k u_n \right)$$

$$= \frac{\lambda_1^k}{\prod \lambda_T} \left(c_1 u_1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k u_2 + \cdots + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^k u_n \right)$$

$k \gg 1$. $\qquad \frac{\lambda_2}{\lambda_1} \equiv \sigma$

$$= c_1 u_1 + c_2 \sigma^k u_2 \qquad \text{dominance ratio } \sigma \equiv \frac{\lambda_2}{\lambda_1} \quad \frac{\lambda_1}{\lambda_2}$$

practical 하게는 pseudo error의 비교해서 dominance ratio로 나뉘 볼 수 있다.

$$\tilde{e}^{(k+1)} \equiv x^{(k+1)} - x^{(k)} = (c_1 u_1 + c_2 \sigma^{k+1}) - c_1 u_1 - c_2 \sigma^k = c_2 (\sigma^{k+1} - \sigma^k)$$

$$\tilde{e}^{(k)} \equiv x^{(k)} - x^{(k-1)} = (c_1 u_1 + c_2 \sigma^k) - c_1 u_1 - c_2 \sigma^{k-1} = c_2 (\sigma^k - \sigma^{k-1})$$

$$\frac{\|\tilde{e}^{(k+1)}\|}{\|\tilde{e}^{(k)}\|} = \frac{|c| \|\sigma^{k+1} - \sigma^k\|}{|c| \|\sigma^k - \sigma^{k-1}\|} = \frac{|\sigma^{k-1}| \|\sigma^2 - \sigma\|}{|\sigma^{k-1}| \|\sigma - 1\|} = \sigma$$

8

(d) iteration loop 안에서 convergence 조건을 pseudo error로써 구하게 되는데 $\tilde{e}^{(k)} = x^{(k)} - x^{(k-1)}$ 실제 오차는 $e^{(k)} = x^{(k)} - x^*$ 이다.

$$\frac{e^{k+1} = x^{(k+1)} - x^*}{e^k = x^{(k)} - x^*} = \rho, \qquad x^{(k+1)} - x^* = \rho (x^{(k)} - x^*)$$

$$(1-\rho) x^* = x^{(k+1)} - \rho x^{(k)}$$

10

$$x^* = \frac{x^{(k+1)} - \rho x^{(k+1)} + \rho x^{(k+1)} - \rho x^{(k)}}{(1-\rho)} = x^{(k+1)} + \frac{\rho}{1-\rho} (x^{(k+1)} - x^{(k)})$$

따라서 false convergence를 피하기 위해서 $e^{(k)} = x^{(k)} - x^*$

$$= x^{(k)} - \left(x^{(k)} - \frac{\rho}{1-\rho} \tilde{e}^{(k)} \right) = \frac{\rho}{1-\rho} \tilde{e}^{(k)}$$ 사용

(e) 초기 벡터 $x^{(0)} = c_1 u_1 + c_2 u_2 + c_3 u_3 + \cdots + c_n u_n$ 로 위의

u_1 성분인 c_2 가 없는 경우에는 dominance ratio 가 2번째 eigen value 이

바로서 즉, $\frac{\lambda_3}{\lambda_1}$ 로 나올수 있다.

(10)

3.(a)

f_0	f_1	f_2	f_3
x_0	x_1	x_2	x_3
0	h	2h	3h

$$P_3(x) = \sum_{i=0}^{3} f_i \prod_{\substack{j=0 \\ j\neq i}}^{3} \frac{(x - x_j)}{(x_i - x_j)}$$

$$= \sum_{i=0}^{3} f_i L_3^i(x)$$

~~적분하면~~ $\int_0^{3h} P_3(x)\,dx = \int_0^{3h} \sum_{i=0}^{3} f_i L_3^i(x)\,dx = \sum_{i=0}^{3} f_i \underbrace{\int_0^{3h} L_3^i(x)\,dx}_{= w_i}$

$$L_3^0(x) = \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} = \frac{(x-h)(x-2h)(x-3h)}{(-h)(-2h)(-3h)}$$

$\frac{x}{h} = \tau$ 치고

$dx = h\,d\tau$

$$= -\frac{1}{6}(\tau-1)(\tau-2)(\tau-3) = \ell_3^0(\tau)$$

$$w_0 = \int_0^{3h} L_3^0(x)\,dx = h\int_0^3 \ell_3^0(\tau)\,d\tau$$

$(\tau^2 - 3\tau + 2)(\tau - 3)$

$\tau^3 - 3\tau^2 + 2\tau - 3\tau^2 + 9\tau - 6$

$\tau^3 - 6\tau^2 + 11\tau - 6$

$$= h\int_0^3 -\frac{1}{6}(\tau^3 - 6\tau^2 + 11\tau - 6)\,d\tau$$

$$= -\frac{h}{6}\left[\frac{1}{4}\tau^4 - 2\tau^3 + \frac{11}{2}\tau^2 - 6\tau\right]_0^3$$

$$= -\frac{h}{6}\left[\frac{81}{4} - 54 + \frac{99}{2} - 18\right]$$

$$= -\frac{h}{6}\left(-\frac{9}{4}\right) = \frac{3}{8}h$$

(10)

$3\times3\times3\times\frac{3}{4}$ 81   99×2

99 ×2 198 81 $\frac{279}{4}$ +54 +18 72 $\frac{-72\times4}{4}$ $\frac{279-288}{4}$ −112   72 ×4 288   288 279 9

(b)

f_0	f_1	f_2	f_3
t_0	t_1	t_2	t_3
−2h	−h	0	h

$$\frac{dy}{dt} = f(t, y)$$

3차

이 방법은 $\binom{t_0}{f_0}, \binom{t_1}{f_1}, \binom{t_2}{f_2}$ 를 사용해서 ~~2차~~ polynomial 얻고 구간 $[t_2\ t_3]$ 에서 적분하여

평균기울기를 얻어 y_3를 구하는 방법이다.

$P_2(t) = \sum_{i=0}^{2} f_i L_2^{(i)}(t)$ → $I = \sum f_i \boxed{\int_{t_2}^{t_3} L_2^{(i)}(t)\,dt}$

↳ weight factor for f_i

$$L_2^{(0)}(t) = \frac{(t-t_1)(t-t_2)}{(t_0-t_1)(t_0-t_2)} = \frac{(t+h)(t)}{(-h)(-2h)}$$

$\frac{t}{h} = \tau$ 치환

$dt = h\,d\tau$

$$= \frac{1}{2}(\tau+1)(\tau)$$

$$w_0 = \int_{t_2}^{t_3} L_2^{(0)}(t)\,dt = \int_0^h \frac{t(t+h)}{2h^2}\,dt$$

$$= h\int_0^1 \left(\frac{1}{2}\tau^2 + \frac{1}{2}\tau\right) d\tau = h\left[\frac{1}{6}\tau^3 + \frac{1}{4}\tau^2\right]_0^1$$

$\frac{10}{24}$

$$= h\left(\frac{1}{6} + \frac{1}{4}\right) = \frac{10}{24}h$$

(C)

f_0	f_1	f_2	f_3	
x_0	x_1	x_2	x_3	
$-3h$	$-2h$	$-h$	0	h

마찬가지로 생각하면 $x_0 \sim x_3$ 의 4개 점들로부터 3차 polynomial

→ $[0, h]$에서 적분하여 평균기울기 구함, 따라서 f_0의 weight factor는

$$w_0 = \int_0^h L_3^{(0)}(x)\,dx$$

$$L_n^{(i)}(x) = \prod_{\substack{j=0 \\ j\neq i}}^{j=n} \frac{(x-x_j)}{(x_i-x_j)} = \frac{(-1)^{n-i}}{i!\,(n-i)!}\prod_{\substack{j=0 \\ j\neq i}}^{n}(\tau-j)$$

$$\left(L_3^{(0)}(x) = \frac{(x+2h)(x+h)\,x}{(-h)(-2h)(-3h)} = -\frac{1}{6}\left[\tau(\tau+1)(\tau+2)\right] \right)$$

$\left(\frac{x}{h} = \tau\right)$

$\tau^2 + 3\tau + 2$

10

$$= -\frac{h}{6}\int_0^1 (\tau^3 + 3\tau^2 + 2\tau)\,d\tau = -\frac{h}{6}\left[\frac{1}{4}\tau^4 + \tau^3 + \tau^2\right]_0^1$$

$\frac{1}{4} + \frac{8}{4} = \frac{9}{4}$

$$= -\frac{9}{4}\cdot\frac{h}{6} = -\frac{3}{8}$$

4. (a) Gaussian Quadrature는 nonuniform한 n개의 점을 Integration할때, $\int = \sum w_i f_i$ 로서 w_i 와 x_i 를 잘구하면 $2n-1$차의 precision을 얻을수 있다는 것이다. 어떤 구간 $[a, b]$ 내의 $[x_1, x_2]$ point 에서의 $[w_1, w_2]$ weight factor를 구하는식을 찾아보면, $2\cdot2-1 = 3$ 차 까지의 polynomial 을 만족시켜야 하는데 0차에대해서 $f(x) = 1$ 이면 $f(x_1) = 1$ $f(x_2) = 1$

15, 0

$$\int_a^b f(x)\,dx = (b-a) = w_1 + w_2$$

1차에 대해서 $f(x) = x$ 이면 $f(x_1) = x_1$, $f(x_2) = x_2$.

$$\int_a^b x\,dx = \boxed{\frac{1}{2}(b^2 - a^2) = w_1 x_1 + w_2 x_2}$$

2차에 대해서 $f(x) = x^2 \rightarrow f(x_1) = x_1^2$. $f(x_2) = x_2^2$.

$$\int_a^b x^2\,dx = \boxed{\frac{1}{3}(b^3 - a^3) = w_1 x_1^2 + w_2 x_2^2}$$

3차에 대해서 $f(x) = x^3 \longrightarrow f(x_1) = x_1^3$. $f(x_2) = x_2^3$.

$$\int_a^b x^3\,dx = \boxed{\frac{1}{4}(b^4 - a^4) = w_1 x_1^3 + w_2 x_2^3}$$

10

위와 같이 4개의 구속식으로부터 4변수 x_1, x_2, w_1, w_2를 구할수 있다.

(b) 위 방법은 Nonlinear equation 이 나와서 풀기 어려워지겠는데, 다른방식으로서 어떤 임의의 2n-1차 다항식을 다음과 같이 나타내고,

$$P(x) = Q(x) P_n(x) + R(x) \quad 로$$

2n-1차 / Legendre polynomial의 n-1차까지 combination / n차 Legendre polynomial / n-1차

$f(x)$를 $P(x)$와 일치시키면 2n-1차 까지 precision을 얻을수 있게 된다, x_i가 n차 Legendre polynomial의 n개의 근중 하나라고 하면

$f(x_i) = P(x_i) = R(x_i)$ 가 만족되므로. n개의 data point

$(x_1, f_1)\ (x_2, f_2) \cdots (x_n, f_n)$ 으로부터 라그랑지 polynomial로 n-1차 다항식 $R(x)$를 구해낼수 있게 된다, $R(x) = \sum_{i=1}^{n} f_i \prod_{\substack{j=1 \\ j\neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}$

구간 [-1, 1]에서 적분한다고 할때 Legendre Polynomial의 직교성에 의해

$$\int_{-1}^{1} f(x)\,dx = \int_{-1}^{1} Q(x) P_n(x) + R(x)\,dx = \int_{-1}^{1} R(x)\,dx$$

$$= \sum_{i=1}^{n} f_i \underbrace{\int_{-1}^{1} \prod_{\substack{j=1 \\ j\neq i}}^{n} \frac{(x - x_j)}{(x_i - x_j)}\,dx}_{w_i}$$

3시 10

Weighting function 으로 w_i로 구할수 있게 된다. 2n-1차 precision을 만족시킴

2003-12491 이원재

5. (a) 15

take interval [a, b] ✓

$x_0 = \frac{b+a}{2}$; (초기 위치를 중점으로) ✓

while ([didnotconverge=true])

$x_{new} = x_{old} - \frac{f(x_{old})}{f'(x_{old})}$ ✓

$\frac{f(x+\varepsilon)-f(x-\varepsilon)}{2\varepsilon}$

$del x = x_{new} - x_{old}$

if ($del x / x_{old} < \varepsilon$) ✓

didnotconverge = false;

else

didnotconverge = true;

end.

end.

10

(b) Newton-Rhapson method

$x_k = x_{k-1} - \frac{f(x_{k})}{f'(x_{k-1})} = g(x_{k-1})$

$g(x) = x - \frac{f(x)}{f'(x)}$ ✓

$x^* = g(x^*)$

$g(x_k) - g(x^*)$

$x_{k+1} - x^* = g(x_k) - g(x^*)$

$= \frac{g(x_k) - g(x^*)}{(x_k) - (x^*)} (x_k - x^*)$ ✓

$\frac{e_{k+1}}{e_k} = \frac{x_{k+1} - x^*}{x_k - x^*} = g'(\xi(x_k)) \quad (x_k \le \xi \le x^*)$

convergence factor

$g'(x) = 1 - \frac{(f'(x))^2 - f(x) f''(x)}{(f'(x))^2}$

$= \frac{f(x) f''(x)}{(f'(x))^2}$

x^* 에서

$g'(x^*) = \frac{f(x^*) \quad f''(x^*)}{f'(x^*)^2} = 0 < 1$ ✓

즉 x^* 근처에서 항상 수렴

공차가수렴

10

(c) k번째 해까지 찾았을때 $\xi_1, \xi_2, \cdots \xi_k$.

$f(x) \leftarrow \frac{f(x)}{\prod_{i=1}^{k} (x - \xi_i)}$ ✓

로 바꿔주면 이미 찾은 해는 더이상 찾지 않게 된다.

따라서 numerical하게 $f(x)$를 구한것에 $\frac{1}{\prod_{i=1}^{k}(x-\xi_i)}$ 를 곱해준값을 함수값으로 취하면됨

10

2003-12491 이 원재

6. 15 4차 Runge-Kutta.

(a) $y' = f(x, y)$ 에 대해 ✓

초기값 (x_0, y_0) time step h

$y_1' = f(x_n, y_n) \rightarrow k_1 = h y_1' = hf(x_n, y_n)$

$y_2' = f(x_n + \alpha_1 h, y_n + \beta_1 k_1) \rightarrow k_2 = h y_2'$

$y_3' = f(x_n + \alpha_2 h, y_n + \beta_2 k_2) \rightarrow k_3 = h y_3'$

$y_4' = f(x_n + \alpha_3 h, y_n + \beta_3 k_3) \rightarrow k_4 = h y_4'$

$\hat{y}_n' = \omega_1 y_1' + \omega_2 y_2' + \omega_3 y_3' + \omega_4 y_4'$ 10

$y_{n+1} = y_n + h \hat{y}_n' = y_n + \omega_1 h y_1' + \omega_2 h y_2' + \omega_3 h y_3' + \omega_4 h y_4'$

$= y_n + \omega_1 k_1 + \omega_2 k_2 + \omega_3 k_3 + \omega_4 k_4$

6

(b) $x_n \rightarrow x$

$y_2' = f(x + \alpha_1 h, y + \beta_1 hf)$

$= f(x, y) + \frac{\partial f}{\partial x} \alpha_1 h + \frac{\partial f}{\partial y} \beta_1 hf$

$+ \frac{1}{2} f_{xx} (\alpha_1 h)^2 + f_{xy} \alpha_1 h \beta_1 hf + \frac{1}{2} f_{yy} (\beta_1 hf)^2$

$+ O(h^3)$

$y_3' = f(x + \alpha_2 h, y + \beta_2 h y_2')$

$= f(x, y) + f_x \alpha_2 h + f_y \beta_2 h y_2'$

$+ \frac{1}{2} f_{xx} (\alpha_2 h)^2 + f_{xy} \alpha_2 h \beta_2 h y_2' + \frac{1}{2} f_{yy} (\beta_2 h y_2')^2$

$+ \cdots$

$y_4' = f(x, y) + f_x \alpha_3 h + f_y \beta_3 h y_3'$

$+ \frac{1}{2} f_{xx} (\alpha_3 h)^2 + f_{xy} \alpha_3 h \beta_3 h y_3' + \frac{1}{2} f_{yy} (\beta_3 h y_3')^2 + \cdots$

$k_1 = hf$

$k_2 = hf + f_x \alpha_1 h^2 + f_y \beta_1 f h^2$

$+ \frac{1}{2} f_{xx} \alpha_1^2 h^3 + f_{xy} f \alpha_1 \beta_1 h^3$

$+ \frac{1}{2} f_{yy} f^2 \beta_1^2 h^3 + O(h^4)$

$k_3 = hf + f_x \alpha_2 h^2 + f_y \beta_2 y_2' h^2$

$+ \frac{1}{2} f_{xx} \alpha_2^2 h^3 + f_{xy} y_2' \alpha_2 \beta_2 h^3$

$+ \frac{1}{2} f_{yy} y_2'^2 \beta_2^2 h^3 + \cdots$

$k_4 = hf + f_x \alpha_3 h^2 + f_y \beta_3 y_3' h^2$

$+ \frac{1}{2} f_{xx} \alpha_3^2 h^3 + f_{xy} y_3' \alpha_3 \beta_3 h^3$

$+ \frac{1}{2} f_{yy} y_3' \beta_3^2 h^3 + \cdots$

y_{n+1} 을 위한 증분 $= \omega_1 k_1 + \omega_2 k_2 + \omega_3 k_3 + \omega_4 k_4$

$y_{n+1} = y_n + y_n' h + \frac{1}{2} y_n'' h^2 + \frac{1}{6} y_n''' h^3 + \cdots$

$= y_n + \boxed{fh + \frac{1}{2} f' h^2 + \frac{1}{6} f'' h^3 + \cdots}$

$\hookrightarrow \omega_1 k_1 + \omega_2 k_2 + \omega_3 k_3 + \omega_4 k_4$ 와 일치

$f' = f_x + f_y f$

$f'' = \frac{d}{dx}(f_x + f_y f)$ $\rightarrow (f_{yx} + f_{yy} f) f$

$= f_{xx} + f_{xy} f + \frac{d f_y}{dx} f + f_y \frac{df}{dx}$

$= f_{xx} + f_{xy} f + f_{xy} f + f_{yy} f^2 + f_x f_y + f_y^2 f$

$= f_{xx} + 2 f_{xy} f + f_{yy} f^2 + f_x f_y + f_y^2 f$

$\frac{1}{6} \times (\cdots + 2 f_{xy} f + \quad) h^3$ ✓

계수 $\frac{1}{3}$

6. (b) 계속

h^3 차의 $f_{xy}f$ 부분을 비교하면.

$$\frac{1}{3} = w_2 \alpha_1 \beta_1 + w_3 \alpha_2 \beta_2 + w_4 \alpha_3 \beta_3$$

$\alpha_i = \beta_i$ 조건 사용하면

$$\frac{1}{3} = w_2 \alpha_1^2 + w_3 \alpha_2^2 + w_4 \alpha_3^2$$

10

(c) m차 ODE의 경우

$$y^{(m)} = f(x, y, y' \cdots y^{(m-1)})$$

$$y = y_1$$

$$\boxed{\begin{aligned} \frac{dy_1}{dt} &= y_2 \\ \frac{dy_2}{dt} &= y_3 \\ &\vdots \\ \frac{dy_{m-1}}{dt} &= y_m \\ \frac{dy_m}{dt} &= f \end{aligned}}$$

로 치환하여

$$\begin{bmatrix} y_1' \\ y_2' \\ \vdots \\ y_m' \end{bmatrix} = \begin{bmatrix} 0 & 1 & & & \\ 0 & 0 & 1 & & \\ & & & \ddots & \\ & & & & 1 \\ & & f & & \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

10

$y' = A(t)\, y(t)$ 꼴로 변환할 수 있다.

여기에 Runge Kutta method를 사용하면. $f(t) \rightarrow A(t)\, y(t)$

$$k_1 = h A_n y_n$$

$$k_2 = h A_{n+0.5h} (y_n + 0.5 k_1)$$

$$k_3 = h A_{n+0.5h} (y_n + 0.5 k_2)$$

$$k_4 = h A_{n+h} (y_n + k_3)$$

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

로서 구할 수 있다.

7)

① ∇^2 을 차분화하여 matrix form으로 나타내기 (공학문제 풀때 중요하기 때문에)

② Lagrange Interpolation (모든 Interpolation의 기본적)

③ 미분방정식의 풀이와 classification. (여러가지 미분방정식을 이해하기 쉽게

- propagate problem (IVP)
- Equilibrium problem (BVP)
- Eigenvalue problem (BVP)

로 정리하여 이해할 수 있었기 때문에)