总结 (part 1)

2022年6月26日 15:24

I. Fundamentals

1.Errors in finite difference approximations of derivatives (L7)

(1) Numerical differentiation

有限差分 + 泰勒展开

Forward difference:

$$f'(x) \approx \frac{f(x+h)-f(x)}{h}$$

$$f(x+h) = f(x) + hf'(x) + \frac{1}{2}h^2 f''(x) + \frac{1}{6}h^3 f'''(x) + \cdots$$

$$\frac{f(x+h)-f(x)}{h} = f'(x) + \boxed{\frac{1}{2}hf''(x)} + \cdots$$

h is small but it is not approaching 0

Leading order error $\sim O(h)$

Leading order error = $\frac{1}{2}hf''(x)$ ~O(h)

Backward difference:

$$f'(x) \approx \frac{f(x)-f(x-h)}{h}$$

$$f(x-h) = f(x) - hf'(x) + \frac{1}{2}h^2 f''(x) + \frac{1}{6}h^3 f'''(x) + \cdots$$

$$\frac{f(x)-f(x-h)}{h} = f'(x) - \frac{1}{2}hf''(x) + \cdots$$

Leading order error $\sim O(h)$

Leading order error = $-\frac{1}{2}hf''(x)$ ~O(h)

Central difference:

$$\frac{f(x+h)-f(x-h)}{2h} = f'(x) + \boxed{\frac{1}{6}h^2 f'''(x)} + \cdots$$

Leading order error $\sim O(h^2)$

$$= \frac{1}{2}\left[\frac{f(x+h)-f(x)}{h} + \frac{f(x)-f(x-h)}{h}\right]$$

$$= \frac{1}{2}\left[f'(x) + \frac{1}{2}hf''(x) + \frac{1}{3}h^2 f'''(x) + \cdots + f'(x) - \frac{1}{2}hf''(x) + \frac{1}{3}h^2 f'''(x) + \cdots\right]$$

Leading order error = $\frac{1}{6}h^2 f'''(x)$ ~$O(h^2)$

2nd order derivative using the central difference scheme:

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}\left[\frac{\partial f}{\partial x}\right] = \frac{\partial}{\partial x}\left[\frac{f(x+h)-f(x-h)}{2h}\right]$$

$$= \left[\underbrace{\frac{f(x+2h)-f(x)}{2h}}_{f'(x+h)} - \underbrace{\frac{f(x)-f(x-2h)}{2h}}_{f'(x-h)}\right]/(2h)$$

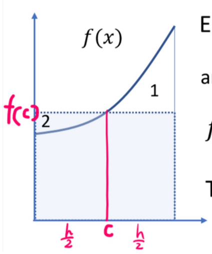
(2) Numerical integration

Rectangular rule

$$\int_{x_0}^{x_1} f(x)dx \approx f\left(\frac{x_0+x_1}{2}\right)h = f(c)h$$

$c \sim midpoint$

Error of the rectangular rule



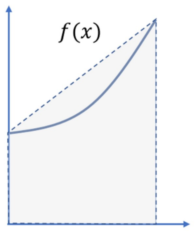
Error ~ area 1 – area 2

$$\text{area} \approx \Delta y \frac{h}{2} = \left[f'(c)\frac{h}{2} + \frac{1}{2}f''(c)\left(\frac{h}{2}\right)^2 + \frac{1}{6}f'''(c)\left(\frac{h}{2}\right)^3 + \cdots\right]\frac{h}{2}$$

$f'(c)\frac{h}{2}$ terms will cancel out

The leading error is $f''(c)\left(\frac{h}{2}\right)^2\frac{h}{2} \sim O(f''h^3)$

Trapezoidal rule



Approximating the integral using a trapezoid

$$\int_{x_0}^{x_1} f(x)dx \approx \left[\frac{1}{2}f(x_0) + \frac{1}{2}f(x_1)\right] h$$ (梯形面积=(上底+下底)*高/2)

$$error\ in\ trapezoidal\ rule \sim O(f''h^3)$$

Both rectangular rule and trapezoidal rule are linear interpolation function. Therefore, they have the similar error ~$O(f''h^3)$

Simpson's rule

$$\int_{x_1}^{x_3} f(x)dx = h\left[\frac{1}{3}f_1 + \frac{4}{3}f_2 + \frac{1}{3}f_3\right] \quad + O(h^5 f^{(4)})$$

High order interpolation using 3 points – smaller error

Bolle's rule

$$\int_{x_1}^{x_5} f(x)dx = h\left[\frac{14}{45}f_1 + \frac{64}{45}f_2 + \frac{24}{45}f_3 + \frac{64}{45}f_4 + \frac{14}{45}f_5\right] \quad + O(h^7 f^{(6)})$$

interpolation using 5 points – even smaller error

2.Lagrange Interpolation Polynomial

Definition:

$$y = \sum_{j=0}^{k} y_j l_j(x) \quad where\ l_j(x) = \prod_{\substack{0\le m\le k \\ m\ne j}} \frac{x - x_m}{x_j - x_m}$$

$$For\ x_i = 0,1,2\ and\ y(x_i) = 0,1,8$$

$$y = y_0 \frac{x-x_1}{x_0-x_1}\frac{x-x_2}{x_0-x_2} + y_1 \frac{x-x_0}{x_1-x_0}\frac{x-x_2}{x_1-x_2} + y_2 \frac{x-x_0}{x_2-x_0}\frac{x-x_1}{x_2-x_1}$$

$$= 1 \times \frac{x(x-2)}{-1} + 8 \times \frac{x(x-1)}{2}$$
$$= x(-2+3x)$$

```
>>>poly.coef
array([ 3., -2.,  0.])
```

3.Bisection method for root finding

A method for solving nonlinear equation.

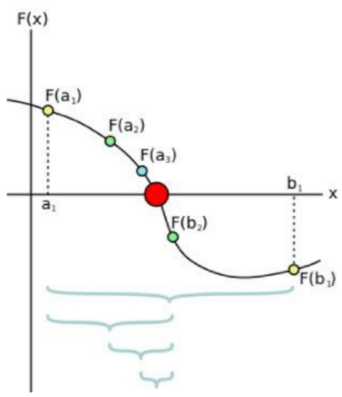
A general function - $f(x) = 0$ Mostly a nonlinear equation

How to find its root?

Bisection method

1. Search over x, find $F(a_1) > 0$ and $F(b_1) < 0$
2. Evaluate the function at $c = (a_1 + b_1)/2$, if $F(c) > 0$ then replace a_1 with c, otherwise replace b_1 with c
3. re-do step 2 until $|F(c) - 0|$ becomes very small

Just a binary search algorithm



4.Reduction of ODEs to 1st order

A general ODE $F\left(x, y, y', y'', \ldots, y^{(n-1)}\right) = y^{(n)}$

Can always be reduced to a system of 1st order ODEs

$y_i = y^{(i-1)}$ then
$$\begin{aligned} y_1' &= y_2 \\ y_2' &= y_3 \\ &\vdots \\ y_{n-1}' &= y_n \\ y_n' &= F(x, y_1, \ldots, y_n) \end{aligned}$$

A system of n 1st order ODEs

把一阶导换元. $y' = y_1' = y_2$

$y_2' = y_3 = y_1'' = y''$

$\vdots$

In vector notation $\mathbf{y}' = \mathbf{F}(x, \mathbf{y})$

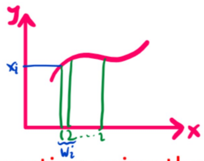
$\mathbf{y} = (y_1, \ldots, y_n)$

$\mathbf{F}(x, y_1, \ldots, y_n) = (y_2, \ldots, y_n, F(x, y_1, \ldots, y_n))$

5.Newton-Cotes formula and its relation with Lagrange interpolation

A numerical integral is basically

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n} w_i\, f(x_i)$$



x_i 应该是等距的.

Construct an interpolation function using the data points
then evaluate the integral base on the interpolation function

Replacing $f(x)$ with the Lagrange interpolation function

uniform convergence 把求和提出来

$$\int_a^b f(x)\,dx \approx \int_a^b \underbrace{L(x)}_{\text{展开}}\,dx = \int_a^b \left(\sum_{i=0}^{n} f(x_i)\, l_i(x)\right) dx = \sum_{i=0}^{n} f(x_i) \underbrace{\int_a^b l_i(x)\,dx}_{w_i}$$

积分等于权重.

$$where\ l_j(x) = \prod_{\substack{0 \le m \le k \\ m \ne j}} \frac{x - x_m}{x_j - x_m}$$

With 2 given points, the 1st order Lagrange polynomials are

$$f(x) = f(x_0) l_0(x) + f(x_1) l_1(x) \quad l_0(x) = \frac{x - x_1}{x_0 - x_1} \quad l_1(x) = \frac{x - x_0}{x_1 - x_0}$$

拉格朗日插值有闭区间公式
积分值要不要的

★ 6.Metropolis-Hastings algorithm for MCMC

纯化论算法

An algorithm to reach a desired distribution
by controlling transitions between states

To reach a distribution $P(x)$

1. Start with an initial state x_t (just randomly picked)
2. Then randomly pick a proposed new state x'
3. Calculate the ratio $a = \frac{P(x')}{P(x_t)}$, $P(x)$~*the desired distribution*
4. If $a > 1$, *then set* $x_{t+1} = x'$
   else:
   pick a random number b (from uniform distribution (0,1))

$$x_{t+1} = \begin{cases} x' \ if\ b < a \\ x_t\ if\ b \ge a \end{cases}$$

Control transition details

两个条件判断

Let the algorithm keep going for a while, then the n-m samples of $x\ (from\ x_{t+m}\ to\ x_{t+n})$ will follow the desired $P(x)$ distribution

7.A-stability analysis for ODE algorithms

A-stability of the explicit Euler method

the Euler algorithm for $y' = k \cdot y$ $\quad y_{n+1} = y_n + h \cdot f(t_n, y_n)$

$$y_{n+1} = y_n + h \cdot (ky_n) = y_n + h \cdot k \cdot y_n = (1 + h \cdot k) y_n$$

$\phi(z) = (1 + h \cdot k) = \dfrac{y_{n+1}}{y_n}$ $\qquad \phi(z) \sim$ Stability function

$\phi(z) := \dfrac{y_{n+1}}{y_n}$

一定是相邻两个的比值.
不能是 $n+2$ vs. n.

$|\phi(z)| < 1$ for the algorithm to converge to 0

For $k = -10$, the algorithm is stable only if $h < 1/5$ $\quad \left|1 + (-10) \cdot \frac{1}{5}\right| = |1-2| = |-1| \le 1$

For $k = -20$, the algorithm is stable only if $h < 1/10$

$\left|1 + (-20) \cdot \frac{1}{10}\right| = |1-2| = |-1| \le 1$

A-stability of the 2nd order Adams-Moulton method

the algorithm is $\quad y_{n+1} = y_n + \frac{1}{2} h \cdot (f(t_n, y_n) + f(t_{n+1}, y_{n+1}))$

For $y' = k \cdot y$ It becomes $y_{n+1} = y_n + \frac{1}{2} h \cdot (ky_n + ky_{n+1})$

$$\Rightarrow \quad y_{n+1} = \frac{1 + \frac{1}{2}hk}{1 - \frac{1}{2}hk} \cdot y_n \quad \Rightarrow \quad \phi(z) = \frac{1 + \frac{1}{2}z}{1 - \frac{1}{2}z}$$

$\left|\dfrac{1 + \frac{1}{2}z}{1 - \frac{1}{2}z}\right| < 1$ As long as $z < 0$ (which is always true since $k < 0$)

The stability is independent of h

8.Newton-Raphson method

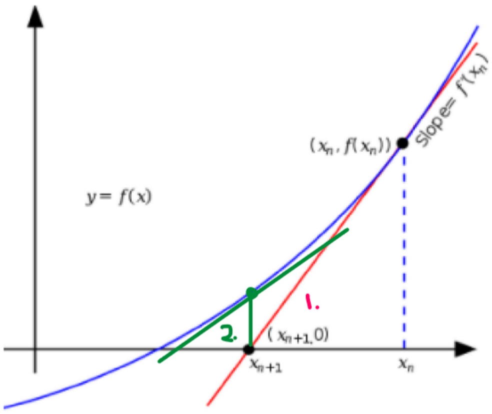
With a single starting point x_0

We know $f(x)$

So, we also know $f'(x_0)$

Find the intersection of the derivative line with the x axis

Use the intersection x coordinate as the starting point
Repeat the algorithm



$$f'(x_n) = \frac{f(x_n) - 0}{x_n - x_{n+1}} \quad \Rightarrow \quad x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Newton Raphson