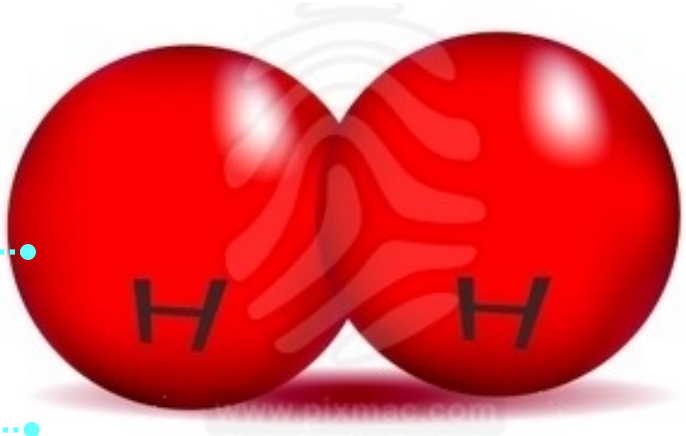
第一章
基本数学运算

主讲教师：蒋臣威

西安交通大学　　理学院

XI'AN JIAOTONG UNIVERSITY

本章内容



1. 数值微分
2. 数值积分
3. 方程求根
4. 分子振动的半经典量子化

1. 数值微分

为什么要学习数值微分？

我们碰到的函数往往没有解析形式，例如可能是如下数表形式
如利用计算所得势能曲线上的点求梯度力。

x	0.1	0.2	0.3	0.4	0.5
$f(x)$	4	4.5	6	8	8.5

这就需要借助于数值微分。
当函数f(x)过于复杂时，也可借助数值微分。

更重要的，数值微分是其它很多数值方法的基础

微积分中，关于导数的定义如下：

$$f'(x)=\lim_{h\to 0}\frac{f(x+h)-f(x)}{h}=\lim_{h\to 0}\frac{f(x)-f(x-h)}{h}=\lim_{h\to 0}\frac{f(x+h)-f(x-h)}{2h}$$

自然，而又简单的方法就是，取极限的近似值，即差商（差分）

向前差分

$$f'(x_0)\approx\frac{f(x_0+h)-f(x_0)}{h}$$

由Taylor展开

$$f(x_0+h)=f(x_0)+hf'(x_0)+\frac{h^2}{2!}f''(x_0)+\cdots$$

$$=f(x_0)+hf'(x_0)+\frac{h^2}{2!}f''(\xi),x_0\le\xi\le x_0+h$$

$$\Rightarrow f'(x_0)=\frac{f(x_0+h)-f(x_0)}{h}-\frac{h}{2!}f''(\xi),x_0\le\xi\le x_0+h$$

因此，有误差

$$R(x) = f'(x_0) - \frac{f(x_0+h) - f(x_0)}{h} = -\frac{h}{2!} f''(\xi) = O(h)$$

向后差分

$$f'(x_0) \approx \frac{f(x_0) - f(x_0-h)}{h}$$

由Taylor展开

$$f(x_0-h) = f(x_0) - hf'(x_0) + \frac{h^2}{2!} f''(x_0) - \cdots$$

$$= f(x_0) - hf'(x_0) + \frac{h^2}{2!} f''(\xi), x_0 \le \xi \le x_0 + h$$

$$\Rightarrow f'(x_0) = \frac{f(x_0) - f(x_0-h)}{h} + \frac{h}{2} f''(\xi), x_0 \le \xi \le x_0 + h$$

因此，有误差

$$R(x)=f'(x_0)-\frac{f(x_0)-f(x_0-h)}{h}=\frac{h}{2!}f''(\xi)=O(h)$$

中心差分

$$f'(x_0)\approx\frac{f(x_0+h)-f(x_0-h)}{2h}$$

由Taylor展开

$$f(x_0+h)=f(x_0)+hf'(x_0)+\frac{h^2}{2!}f''(x_0)+\frac{h^3}{3!}f'''(x_0)+\cdots$$

$$=f(x_0)+hf'(x_0)+\frac{h^2}{2!}f''(x_0)+\frac{h^3}{3!}f'''(\xi_1),x_0\le\xi_1\le x_0+h$$

$$f(x_0-h)=f(x_0)-hf'(x_0)+\frac{h^2}{2!}f''(x_0)-\frac{h^3}{3!}f'''(x_0)+\cdots$$

$$=f(x_0)-hf'(x_0)+\frac{h^2}{2!}f''(x_0)-\frac{h^3}{3!}f'''(\xi_2),x_0-h\le\xi_2\le x_0$$

上面两式相减得：

$$f(x_0+h)-f(x_0-h)=2hf'(x_0)+\frac{h^3}{3!}f'''(\xi_1)+\frac{h^3}{3!}f'''(\xi_2)$$

$$f'(x_0)=\frac{f(x_0+h)-f(x_0-h)}{2h}-[\frac{h^2}{12}f'''(\xi_1)+\frac{h^2}{12}f'''(\xi_2)]$$

$$\approx\frac{f(x_0+h)-f(x_0-h)}{2h}$$

因此，有误差

$$R(x)=f'(x_0)-\frac{f(x_0+h)-f(x_0-h)}{2h}$$

$$=-\frac{h^2}{12}[f'''(\xi_1)+f'''(\xi_2)]=-\frac{h^2}{6}f'''(\xi)=O(h^2)$$

五点公式

$$f_{\pm 1} \equiv f(x_0 \pm h) = f(x_0) \pm hf'(x_0) + \frac{h^2}{2} f''(x_0) \pm \frac{h^3}{6} f'''(x_0) + O(h^4)$$

$$f_{\pm 2} \equiv f(x_0 \pm 2h) = f(x_0) \pm 2hf'(x_0) + 2h^2 f''(x_0) \pm \frac{4}{3} h^3 f'''(x_0) + O(h^4)$$

分别相减

$$f_{+1} - f_{-1} = 2hf'(x_0) + \frac{h^3}{3} f'''(x_0) + O(h^5)$$

$$f_{+2} - f_{-2} = 4hf'(x_0) + \frac{8}{3} h^3 f'''(x_0) + O(h^5)$$

$$8(f_{+1} - f_{-1}) - (f_{+2} - f_{-2}) = 12hf'(x_0) + O(h^5)$$

$$f'(x_0) = \frac{1}{12h}[8(f_{+1} - f_{-1}) - (f_{+2} - f_{-2})] + O(h^4)$$ 五点公式

虽然精度更大，但是计算量也更大！通常用三点公式已足够。

表一. 不同方法计算 $d(\sin x)/dx\,|_{x=1} = 0.540302$ 的误差

h	Symmetric 3-point	Forward 2-point	Backward 2-point	Symmetric 5-point
0.50000	0.022233	0.228254	−0.183789	0.001092
0.20000	0.003595	0.087461	−0.080272	0.000028
0.10000	0.000899	0.042938	−0.041139	0.000001
0.05000	0.000225	0.021258	−0.020808	0.000000
0.02000	0.000037	0.008453	−0.008380	0.000001
0.01000	0.000010	0.004224	−0.004204	0.000002
0.00500	0.000010	0.002108	−0.002088	0.000006
0.00200	−0.000014	0.000820	−0.000848	−0.000017
0.00100	−0.000014	0.000403	−0.000431	−0.000019
0.00050	0.000105	0.000403	−0.000193	0.000115
0.00020	−0.000163	−0.000014	−0.000312	−0.000188
0.00010	−0.000312	−0.000312	−0.000312	−0.000411
0.00005	0.000284	0.001476	−0.000908	0.000681
0.				000873
0.				000880

见Matlab程序
chap1_sinx_differential.m

注意

- 不同公式的精度：五点>三点>两点
- 注意误差随步长 h 的减小先减小再增大

舍入误差

微分公式涉及两个很接近的 f 值**相减**。当步长过小时，计算机的舍入误差会使导数计算不准确。

合适步长

设D(h),D(h/2)分别为步长为h,h/2的差商公式。则

$$\left|D(h)-D(\frac{h}{2})\right|<\varepsilon$$

时的步长h/2就是合适的步长

高阶导数

二阶导数

$$f_1 \equiv f(x_0+h) = f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{6}f'''(x_0) + O(h^4)$$

$$f_{-1} \equiv f(x_0-h) = f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(x_0) - \frac{h^3}{6}f'''(x_0) + O(h^4)$$

$$\Rightarrow \quad f''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2} + O(h^2)$$

这也容易从二阶导数的定义直接得出（利用一阶导数向后差分公式）

$$f''(x_0) = \frac{f'(x_0+h) - f'(x_0)}{h} = \frac{[f(x_0+h) - f(x_0)]/h - [f(x_0) - f(x_0-h)]/h}{h}$$

$$= \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2}$$

更高阶的导数以此类推

尝试给出一个三阶导数的公式

例题1 给定函数 $f(x)=e^{-x/5}$ 在节点 $x_k=1.74+kh$, $(h=0.02, k=0,1,\ldots,5)$ 上的函数值，求函数 $f(x)$ 在内节点处的一阶和二阶微商值，与解析结果比较确定误差大小。

节点坐标	1.76	1.78	1.80	1.82
一阶微商（数值）	-0.1407	-0.1401	-0.1395	-0.1390
一阶微商（误差）	-3.7508e-7	-3.7359e-7	-3.7209e-7	-3.7061e-7
二阶微商（数值）	0.0281	0.0280	0.0279	0.0278
二阶微商（误差）	3.7509e-8	3.7358e-8	3.7210e-8	3.7061e-8

见Matlab程序
chap1_example_1_differential.m
chap1_example_1_differential_revised.m

XI'AN JIAOTONG UNIVERSITY

Matlab中计算数值微分的命令

在Matlab中，没有直接求数值微分的命令，只有差分计算公式diff(f)与gradient(f)

- df=diff(f);求一元函数f(x)的两点向前差分

$$df = f(2:n) - f(1:n-1)$$ ，那么数值微分为df/dx;

- gf=gradient(f); 求一元函数f(x)的三点中心差分

$$gf(2:n-1) = \frac{f(3:n) - f(1:n-2)}{2}$$ ，那么数值微分为gf/gx;

对df而言，当f是向量时，df=f(2:n)-f(1:n-1)；df的长度比f的长度少一个元素；也就是说不求最后一个点的差分。对gradient而言，当f是向量时，gf的非端点gf(2:n-1)=(f(3:n)-f(1:n-2))/2;而首端gf(1)=f(2)-f(1),末端gf(n)=f(n)-f(n-1);注意：gf的长度与f相同，内节点采用中心差分，两端采用向前或向后差分；

见Matlab程序
chap1_example_1_differential_Matlab.m

2.数值积分

为什么要数值积分？

牛顿-莱布尼兹公式

$$I = \int_a^b f(x)dx = F(b) - F(a)$$

但是这要求

- 被积函数 **f(x)** 有解析表达式
- **f(x)** 的原函数 **F(x)** 为初等函数

我们面临的的问题

1) f(x)没有解析表达式

x	0.1	0.2	0.3	0.4	0.5
$f(x)$	4	4.5	6	8	8.5

2) f(x)有解析表达式，但原函数不是初等函数，例如

$$\int_a^b e^{-x^2}dx \qquad \int_a^b \frac{sinx}{x}dx$$

它们的原函数都不是初等函数

3）$f(x)$为形式简单的初等函数，但其原函数表达式复杂；如：

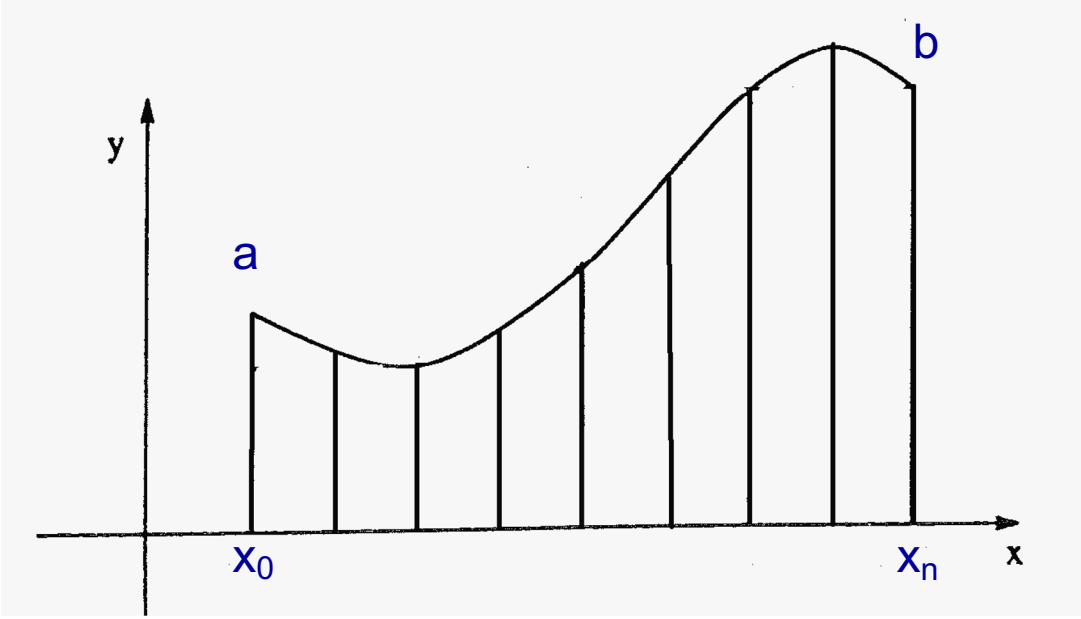
$$f(x)=x^2\sqrt{2x^2+3}$$

$$F(x)=\frac{1}{4}x^2\sqrt{2x^2+3}+\frac{3}{16}x\sqrt{2x^2+3}-\frac{9}{16\sqrt{2}}\ln(\sqrt{2}x+x^2\sqrt{2x^2+3})$$

4） $f(x)$的表达式结构复杂，求原函数困难；

数值积分的出发点

一维定积分的几何意义是曲边梯形的面积。从积分定义可知，定积分的基本分析方法是四步，即分割、近似、求和、取极限。



将区间 **[a,b]** 分割为 **n** 等份，每个小区间的宽度为

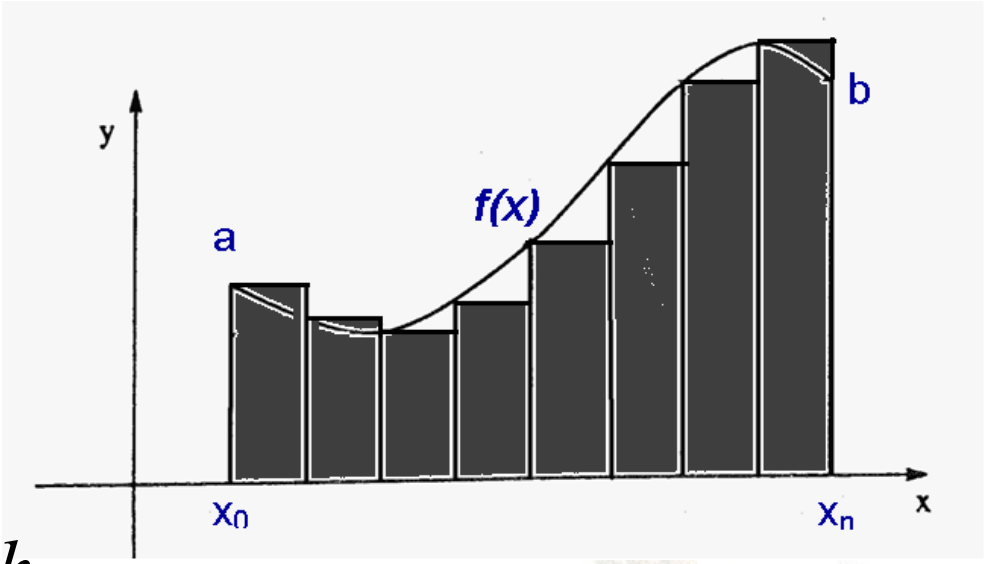
h=(b-a)/n

积分转化为求和 $$\int_a^b f(x)dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x)dx$$

矩形积分公式

将每个小的积分区间上的积分结果

$$\int_{x_i}^{x_{i+1}} f(x)dx$$



近似取为 $f(x_i)\Delta x = f(x_i)h$

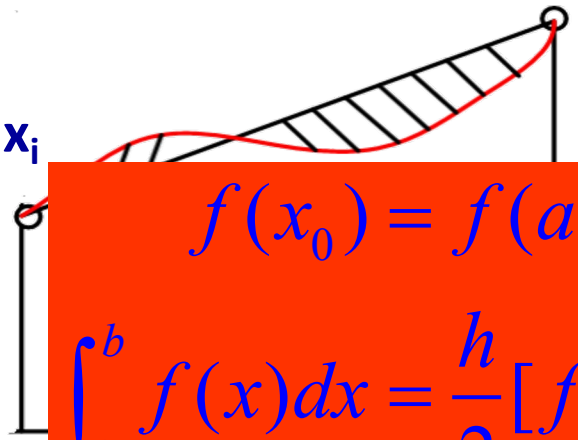
则(1)式变为 $$\int_a^b f(x)dx = \sum_{i=0}^{n-1} f(x_i)h$$

显然可以看出，矩形公式的误差太大。下面介绍几个实用积分公式。

XI'AN JIAOTONG UNIVERSITY

线性近似——梯形公式

在每个区间 $[x_i, x_{i+1}]$ 进行线性插值



$$\varphi(x) = f(x_i)\frac{x - x_{i+1}}{} + f(x_{i+1})\frac{x - x_i}{}$$

Lagrange 插值

$$f(x_0) = f(a), f(x_n) = f(b), x_i = a + ih$$

$$\int_a^b f(x)dx = \frac{h}{2}[f(a) + 2\sum_{i=1}^{n-1} f(a+ih) + f(b)] + O(h^2)$$

$\frac{h}{2}$

$R($ 定理

$$= \frac{h}{2}[2\sum_{i=0}^{n} f(a+ih) - f(a) - f(b)] + O(h^2)$$

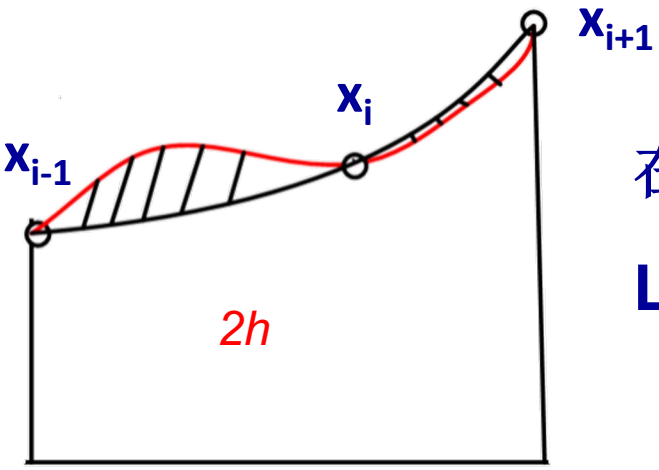
上式又称复化梯形公式

$h^3)$

$$\Rightarrow \int_a^b f(x)dx = \sum_{i=0}^{n-1}\int_{x_i}^{x_{i+1}} f(x)dx = \frac{h}{2}\sum_{i=0}^{n-1}[f(x_i) + f(x_{i+1})] + O(h^2)$$

XI'AN JIAOTONG UNIVERSITY

二阶多项式近似——辛普生公式



在区间 $[x_{i-1}, x_{i+1}]$ 上对 f(x) 进行 Lagrange二阶插值。

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

$$+\frac{(x-x_{i-1})(x-x_i)}{(x_{i+1}-x_{i-1})(x_{i+1}-x_i)}f(x_{i+1})+O(h^3)$$

$$=l_{i-1}(x)f(x_{i-1})+l_i(x)f(x_i)+l_{i+1}(x)f(x_{i+1})+O(h^3)$$

XI'AN JIAOTONG UNIVERSITY

二阶多项式近似——辛普生公式

容易验证 $\int_{x_{i-1}}^{x_{i+1}} l_{i-1}(x)dx = \frac{h}{3}; \int_{x_{i-1}}^{x_{i+1}} l_i(x)dx = \frac{4h}{3}; \int_{x_{i-1}}^{x_{i+1}} l_{i+1}(x)dx = \frac{h}{3}$

$$\Rightarrow \int_{x_{i-1}}^{x_{i+1}} f(x)dx = \frac{h}{3}[f(x_{i-1}) + 4f(x_i) + f(x_{i+1})] + O(h^4)$$

$$\Rightarrow \int_a^b f(x)dx = \sum^{n/2-1} \int^{x_{2j+2}} f(x)dx$$

将[a,b]区间2n等分， $h=(b-a)/2n$, $x_k=a+kh$ $(k=0,1,...2n)$

对于每个$[x_{i-1},x_{i+1}]$区间利用Simpson公式，然后求和可得：

$$\int_a^b f(x)dx = \frac{h}{3}[f(a) + 4\sum_{k=0}^{n-1} f(a+(2k+1)h) + 2\sum_{k=1}^{n-1} f(a+2kh) + f(b)]$$

上式又称复化Simpson公式

不同积分方法的结果比较

计算 $\int_0^1 e^x dx = 1.718282$ 的误差

N	h	Trapezoidal	Simpson's
4	0.2500000	−0.008940	−0.000037
8	0.1250000	−0.002237	0.000002
16	0.0625000	−0.000559	0.000000
32	0.0312500	−0.000140	0.000000
64	0.0156250	−0.000035	0.000000
128	0.0078125	−0.000008	0.000000

注意，此时误差随h减小而减小，舍入误差并不重要，这是因为积分公式中，所有 f 的值的符号都一样。

见Matlab程序

chap1_ex_integration.m

高阶的算法

选取更多的点，进行更高阶的插值可以得到更高阶的算法，如

simpson 3/8 算法（三阶插值）

$$\int_{x_0}^{x_3} f(x)\,dx = \frac{3h}{8}[f_0 + 3f_1 + 3f_2 + f_3] + \mathcal{O}(h^5)$$

将[a,b]区间3n等分， h=(b-a)/3n, x_k =a+kh (k=0,1,...3n)

对于每个[x_{i-1},x_{i+2}]区间利用simpson 3/8算法，然后求和可得：

$$\int_a^b f(x)dx = \frac{3h}{8}[f(a)+3\sum_{k=0}^{n-1}[f(a+(3k+1)h)+f(a+(3k+2)h)]+2\sum_{k=1}^{n-1}f(a+3kh)+f(b)]$$

注意：在积分区间[a,b]的等分数n一定是插值阶数的整数倍。如对于Bode算法，n一定是4的整数倍。

例题2：利用梯形算法，simpson方法，simpson 3/8法计算

$$\pi = \int_0^1 \frac{4}{1+x^2}\,dx$$

梯形算法

$$\int_a^b f(x)dx = \frac{h}{2}[f(a) + 2\sum_{i=1}^{n-1} f(a+ih) + f(b)] + O(h^2)$$

simpson法

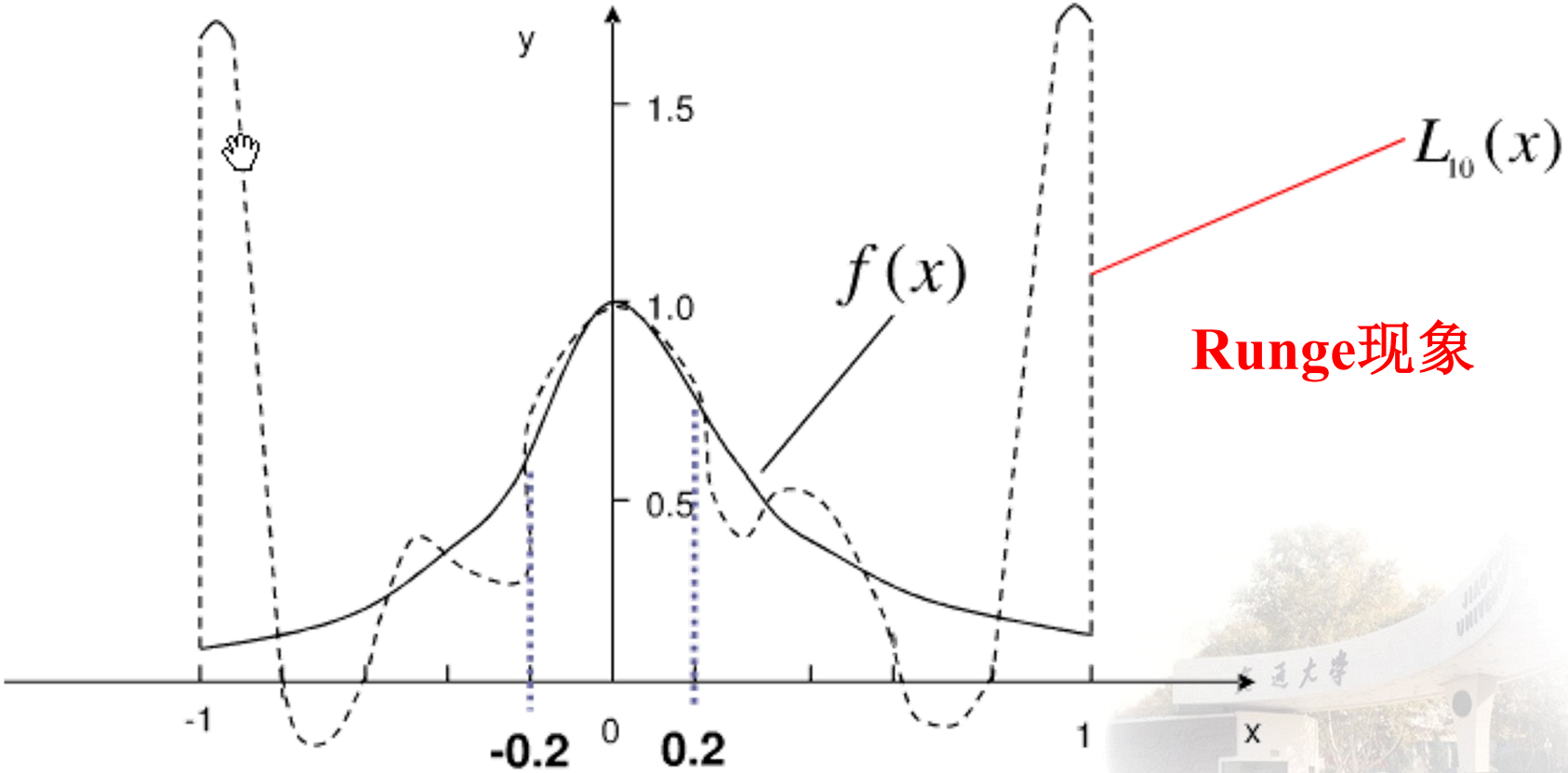
$$\int_a^b f(x)dx = \frac{h}{3}[f(a) + 4\sum_{k=0}^{n-1} f(a+(2k+1)h) + 2\sum_{k=1}^{n-1} f(a+2kh) + f(b)]$$

simpson 3/8法

$$\int_a^b f(x)dx = \frac{3h}{8}[f(a) + 3\sum_{k=0}^{n-1}[f(a+(3k+1)h) + f(a+(3k+2)h)] + 2\sum_{k=1}^{n-1} f(a+3kh) + f(b)]$$

见Matlab程序
chap1_example_2_integration.m

过高阶的插值可能导致严重的振荡行为，从而给出被积函数不准确的插值。所以为了得到更高精度，往往用低阶插值，同时减小 h。



反常积分的处理

反常积分分为**两类**：

- 积分区间有限，在积分区间内被积函数有**奇点**
- 积分区间为无限

策略 通过积分变量的变换，将反常积分变换为普通积分

（1）积分区间内含有奇点的积分 $\int_0^1 dx(1-x^2)^{-1/2}g(x)$

在 x=1 处有一个奇点，假设函数 g 在这点的值有限，则积分为有限值。

做变换 $t=(1-x)^{1/2}$，积分变为 $2\int_0^1 dt(2-t^2)^{-1/2}g(1-t^2)$

（2）无限区间的积分 $\int_1^{\infty} dx x^{-2} g(x)$

$g(x)$ 在 x 很大时趋于常数，积分为有限值。

做变换 $t=x^{-1}$，积分变为 $\int_0^1 dt g(t^{-1})$

Matlab自带的积分指令

- quad 采用自适应Simpson算法。根据积分精度的需要，自动调节积分取点的数目。
- 调用格式为 quad(函数句柄, 积分下限，积分上限)

如 **quad(@sin, 0.5, 0.6, tol)** $\Rightarrow \int_{0.5}^{0.6} \sin(x)dx$

tol为误差标准，缺省时为1e-6. 注意：误差不是与精确值比较，而是与步长进一步减小时的结果进行比较

例题. 利用Matlab自带的积分指令quad计算下述定积分

$$\int_1^{40} \frac{x^4 e^x}{(e^x - 1)^2} dx$$

见Matlab程序
chap1_integration_matlab.m

例题. 利用Matlab自带的积分指令quad计算下述定积分

$$\int_0^{\pi} \frac{\sin x}{x} dx$$

见Matlab程序
chap1_integration_matlab_2.m

- 指令dblquad是计算已知函数的二重积分，积分时要按照积分变量顺序指明两个变量的积分区间。

例题. 利用Matlab自带的积分指令dblquad计算下述定积分

$$\int_{\pi}^{2\pi} dx \int_{0}^{\pi} dy (y \sin x + x \cos y)$$

见Matlab程序
chap1_integration_matlab_3.m

- 指令triplequad是计算已知函数的三重积分，用法与dblquad类似，积分时要按照积分变量顺序指明三个变量的积分区间。

3. 方程求根

3.1 方程数值求根的必要性

阿贝尔-鲁菲尼定理：**高于四次的一般代数方程没有一般形式的代数解**

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 \quad (a_n \neq 0)$$

对于更为复杂的方程，如非线性方程

$$cos(x)cosh(x) + 1 = 0$$

很难利用解析的方法求得方程的根。

后面很多章节，如常微分方程的边值问题的打靶法求解等问题中也将用到方程的数值求根问题。

阿贝尔
(1802—1829)

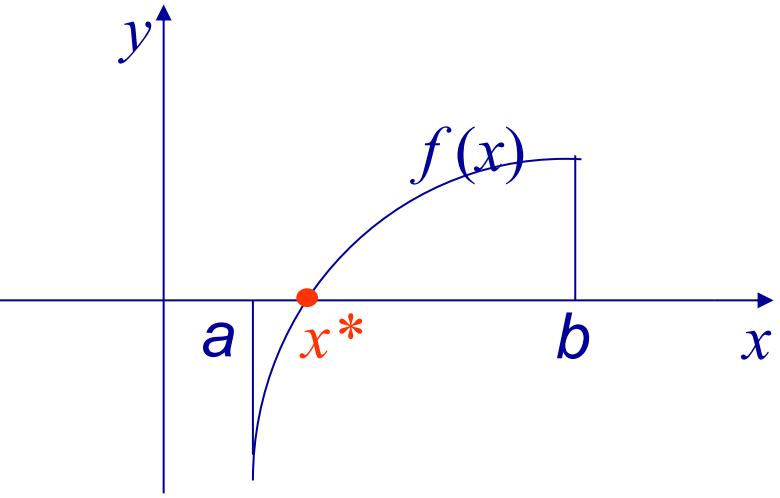
作为基础我们只讨论一元非线性方程的数值求根问题。

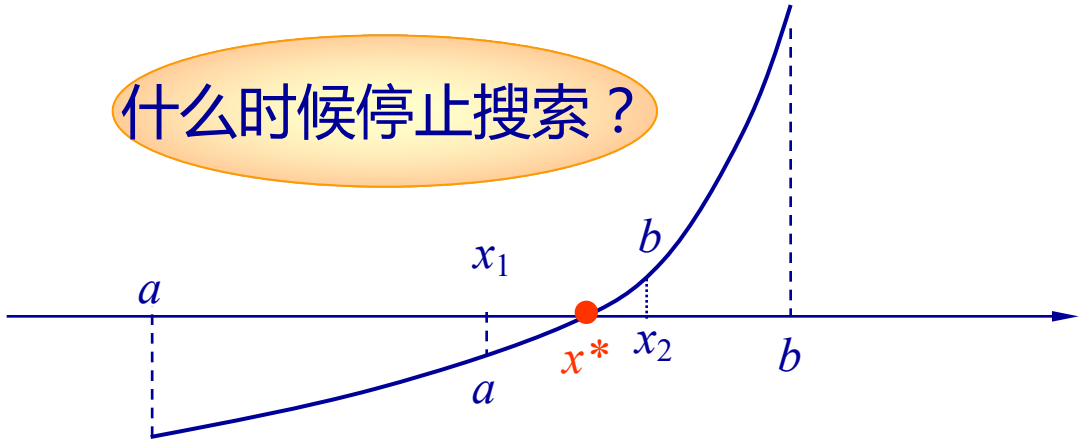
XI'AN JIAOTONG UNIVERSITY

3.2 二分法（对分搜索法）

求 $f(x)=0$ 的根

根的存在性原理：若 $f \in C[a,b]$，（在[a,b]上单调连续），且 $f(a)\cdot f(b)<0$，则 $f(x)=0$ 在 (a,b) 上必有根。





如何搜索？

$$|x_{k+1}-x_k|<\varepsilon_1$$

二分法的算法实现

给定有根区间 [a, b] ($f(a)\,f(b) < 0$) 和 精度要求 ε

1. 令 $x = (a+b)/2$;

2. 如果 $|b - a| < \varepsilon$，结束运算，输出 x;

3. 如果 $f(a)\,f(x) < 0$，则令 $b = x$，否则令 $a = x$, 返回第1步

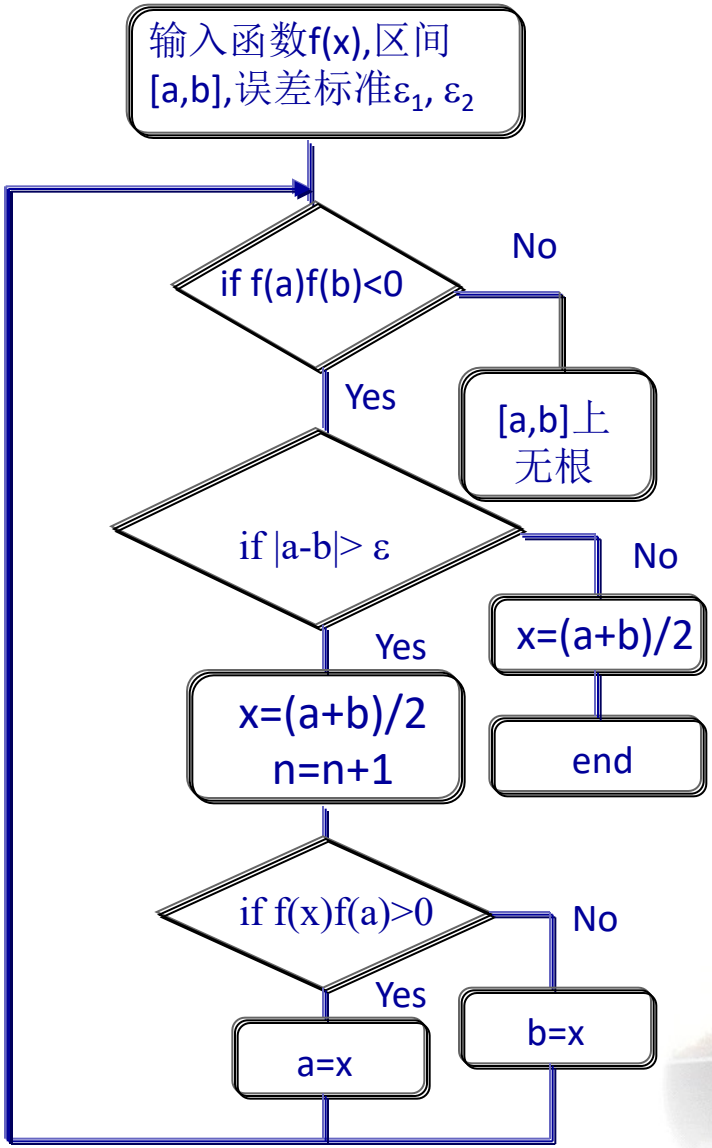
二分法的优缺点

✓ 简单易用

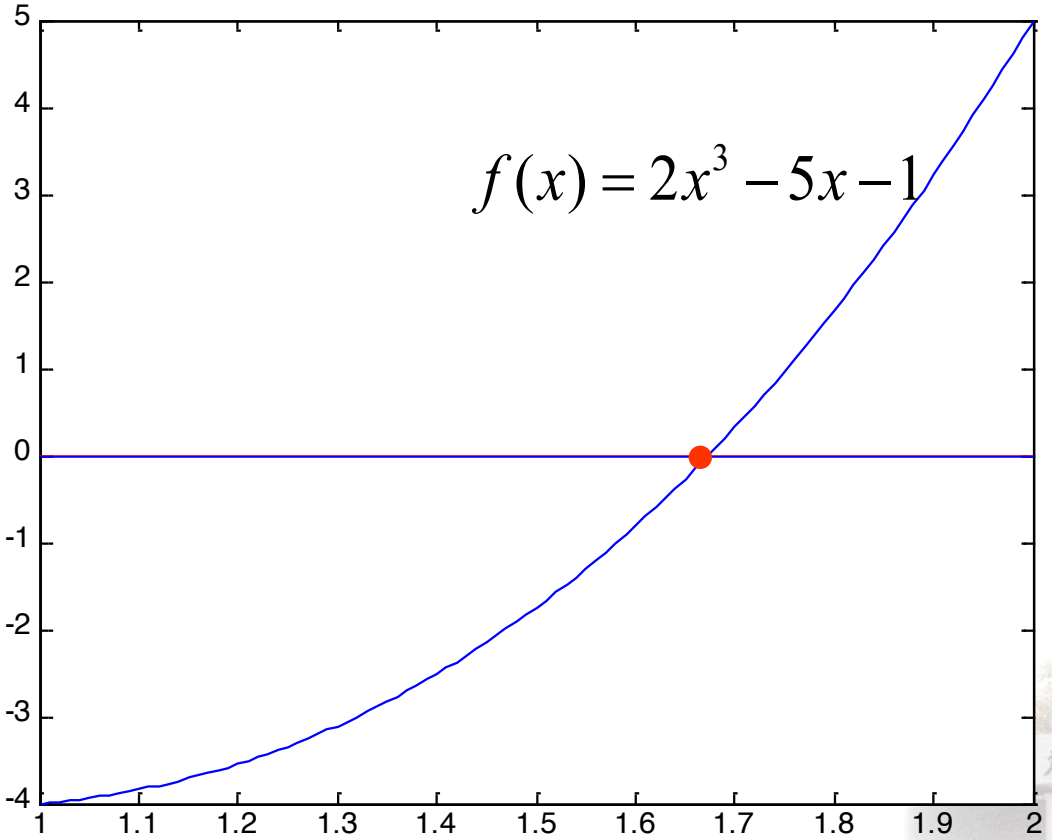
✓ 稳妥，对 $f(x)$ 要求不高，只要连续即可收敛

X 收敛速度慢，且只能求单根

二分法的流程图


输入函数f(x),区间[a,b],误差标准ε1, ε2
if f(a)f(b)<0
No
[a,b]上无根
Yes
if |a-b|> ε
No
x=(a+b)/2
end
Yes
x=(a+b)/2
n=n+1
if f(x)f(a)>0
No
b=x
Yes
a=x

例3：用二分法求方程 $2x^3-5x-1=0$ 在区间 [1,2]内的实根，要求误差限为 $\varepsilon \le 10^{-2}$ 。



解：令 $f(x)=2x^3-5x-1$

$f(1)<0, f(2)>0$ 记 $I_0=[1,2]$, $x_0=(1+2)/2=1.5$

因为 $f(x_0)f(1)>0$ 得 $I_1=[1.5, 2]$, $x_1=(1.5+2)/2=1.75$

$f(x_1)f(1.5)<0$ 得 $I_2=[1.5, 1.75]$, $x_2=(1.5+1.75)/2=1.625$

.....

$I_6=[1.681875, 1.6875]$,

$I_7=[1.671875, 1.679688]$

$b_7 - a_7=0.7813\times10^{-2} < 10^{-2}$

$\therefore x^* \approx x_7 = 1.6758$

见Matlab程序
chap1_example_3_bisection.m

XI'AN JIAOTONG UNIVERSITY

3.2 Newton-Raphson方法

设一元方程 $f(x)=0$ 的非线性函数 $f(x)$ 连续可微。我们在其解的近似值 x_k 附近将 $f(x)$ 作Taylor展开

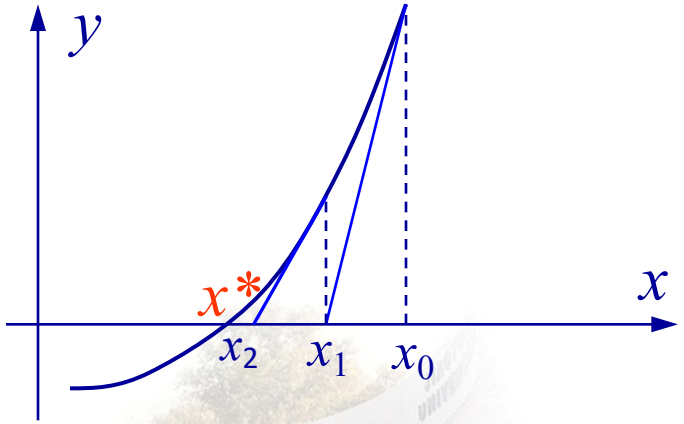
$$f(x^*) = f(x_k) + (x^* - x_k) f'(x_k) + (x^* - x_k)^2 \frac{f''(x_k)}{2!} + \cdots = 0$$

取其线性部分，有 $x^* \approx x_k - \dfrac{f(x_k)}{f'(x_k)}$

几何意义

因此可以构造迭代公式：

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \qquad k = 0,1,2,\cdots$$

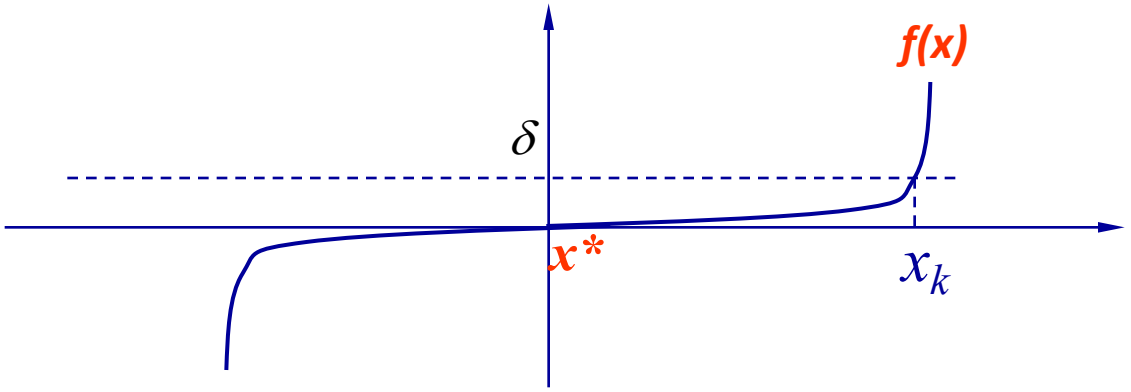


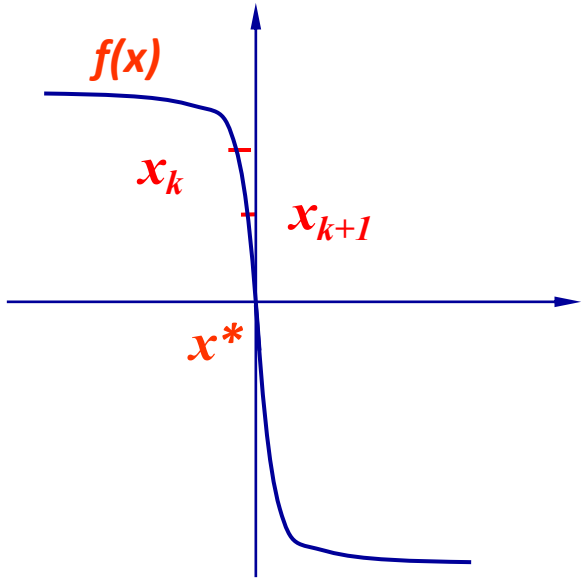
什么时候停止迭代？

$|f(x_k)| < \delta$

and $|x_{k+1} - x_k| < \varepsilon_1$

取k=0, $x_1 \approx x_0 - \dfrac{f(x_0)}{f'(x_0)}$





只用 $|f(x)|<\delta$ 可能造成误差偏大

牛顿迭代法的算法构造

只用 $|x_{k+1}-x_k|<\varepsilon_1$ 可能造成误差偏大

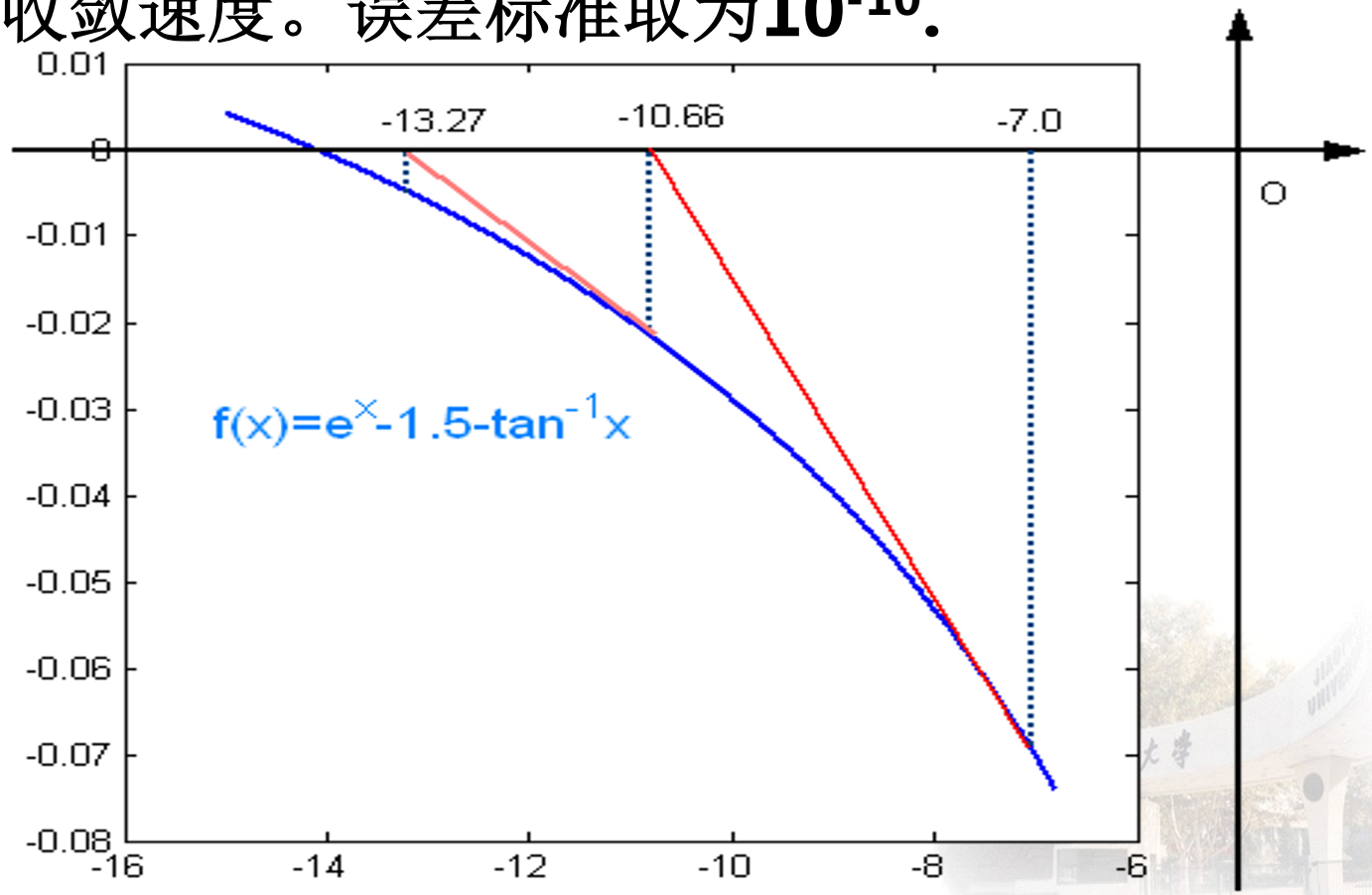
1: 初始化 x_0，误差标准δ, ε_1 ，置 k=0

2:计算 $x_{k+1}=x_k-f(x_k)/f'(x_k)$

3: 如果$|f(x_k)|\le\delta$ 且 $|x_{k+1}-x_k|<\varepsilon_1$ 则停止.

4: $k=k+1$, 转至2

例4： 利用牛顿迭代法求解 $f(x)=e^{x}-1.5-\tan^{-1}x$ 的零点。初始点 $x_0=-7.0$；并用二分法求[-16,-7]上的解，比较两方法的收敛速度。误差标准取为10^{-10}.



解： $f(x_0)= -0.702\times10^{-1}$ ，$f'(x)= e^x-(1+x^2)^{-1}$

计算迭代格式: $x_{k+1} = x_k - \dfrac{f(x_k)}{f'(x_k)}$

计算结果如下表：(取$|f(x)| <= 10^{-10}$)

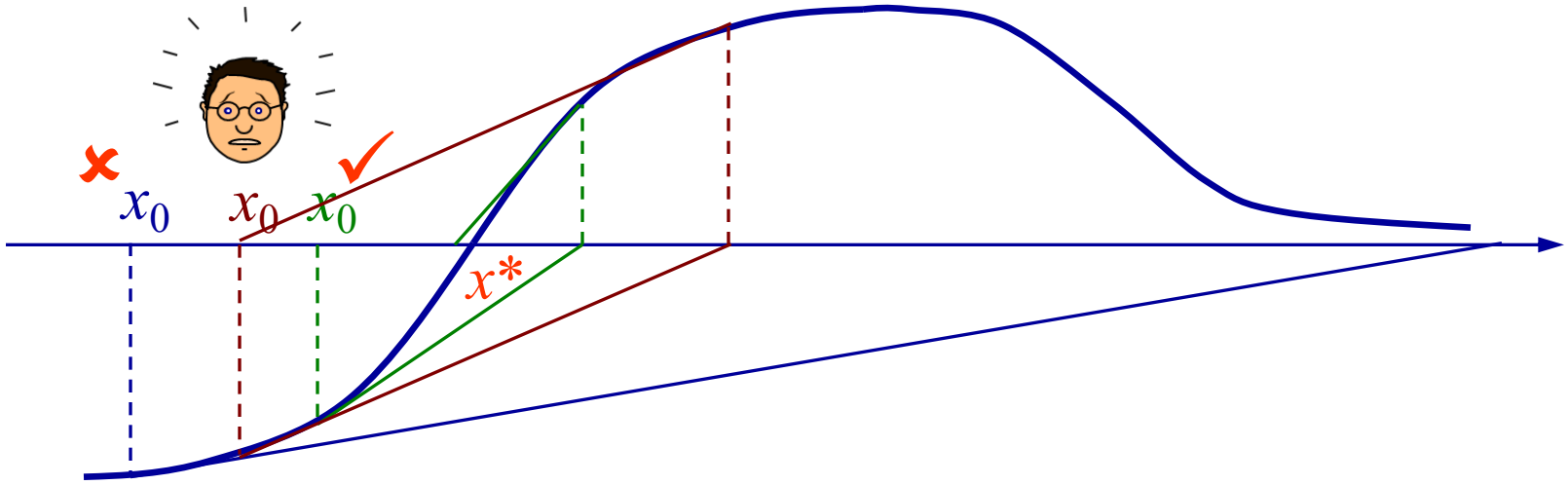
k	x	$f(x)$
0	-7.0000	-0.0701888
1	-10.6771	-0.0225666
2	-13.2792	-0.00436602
3	-14.0537	-0.00023902
4	-14.1011	-7.99585e-007
5	-14.1013	-9.00833e-012

见Matlab程序

chap1_example_4_bisection_newton.m

算法说明

注：Newton-Raphson Method 收敛性依赖于x_0 的选取。

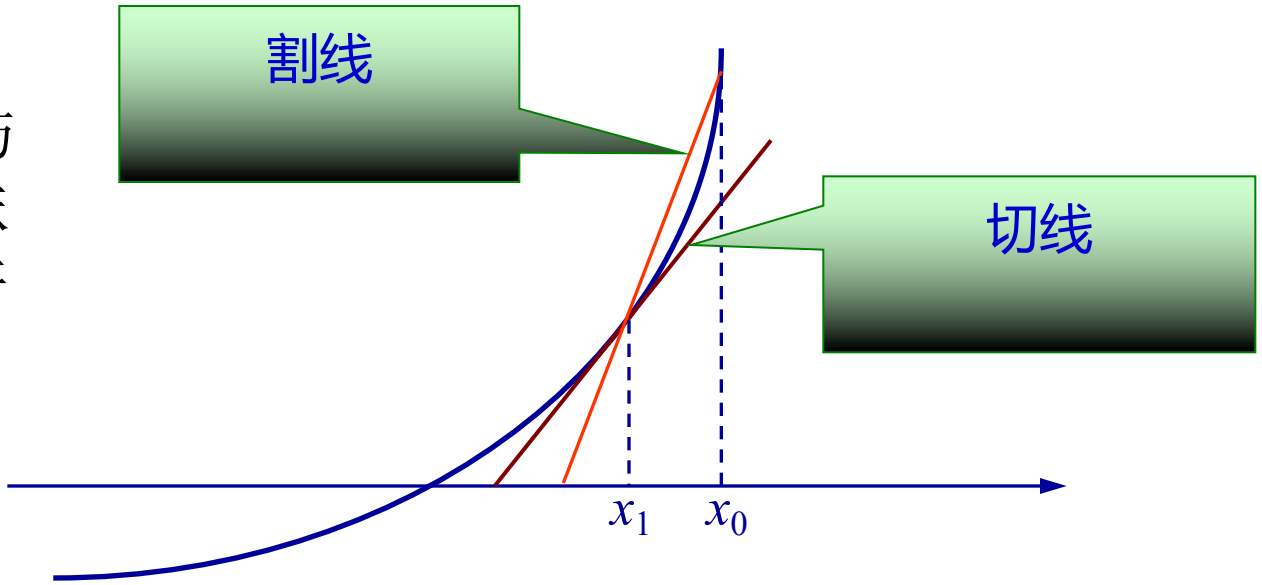


✓收敛快，稳定性好，精度高等优点，是求解非线性方程的有效方法之一

X每次迭代均需计算函数值与导数值，计算量大。当**导数值**提供有困难时， Newton法无法进行。**收敛性依赖初值选择。**

3.4 弦割法

弦割法在Newton-Raphon法的效率与必须计算导数的麻烦间提供了一种折中。

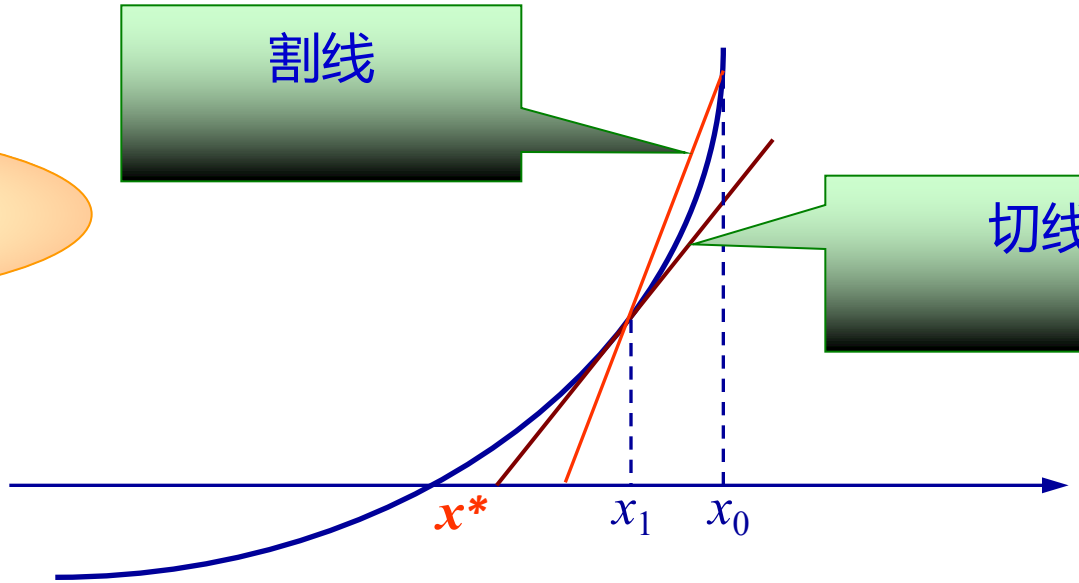


切线斜率 $\approx$ 割线斜率 $\Rightarrow$ $f'(x_k) \approx \dfrac{f(x_k)-f(x_{k-1})}{x_k - x_{k-1}}$ 向后差分

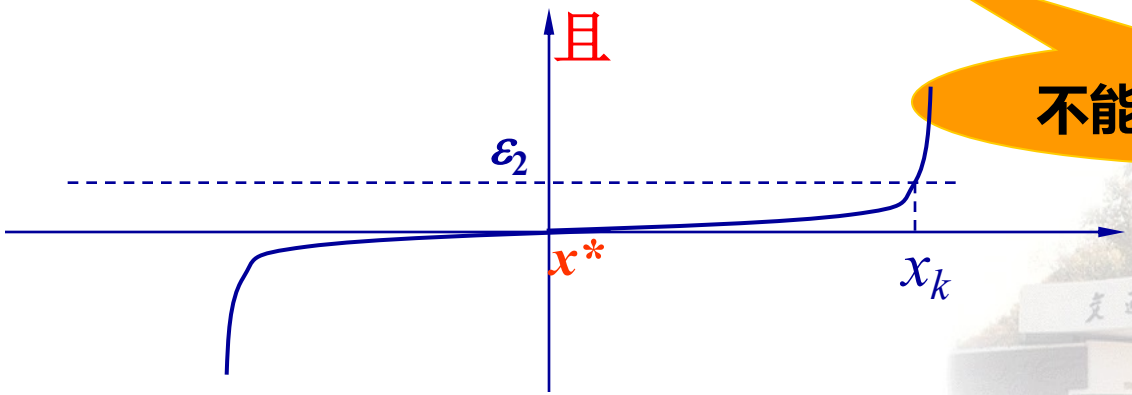
代入Newton迭代公式 $\Rightarrow$ $x_{k+1} = x_k - \dfrac{f(x_k)(x_k - x_{k-1})}{f(x_k)-f(x_{k-1})}$

任意2个初值 x_0 和 x_1 可以启动这个递推关系。

When to stop?



$$|x_{k+1} - x_k| < \varepsilon_1 \quad \text{或} \quad |f(x_k)| < \varepsilon_2$$



不能保证 x_k 的精度

弦割法的算法构造

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

1: 初始化 x_0、x_1, 误差标准δ与ε,置 k=0

2: 如果$|f(x_k)| \le \delta$ 且$|x_k - x_{k+1}| \le \varepsilon$，则停止.

3: 计算 $x_{k+1} = x_k - \dfrac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}$

4: $k=k+1$, 转至2.

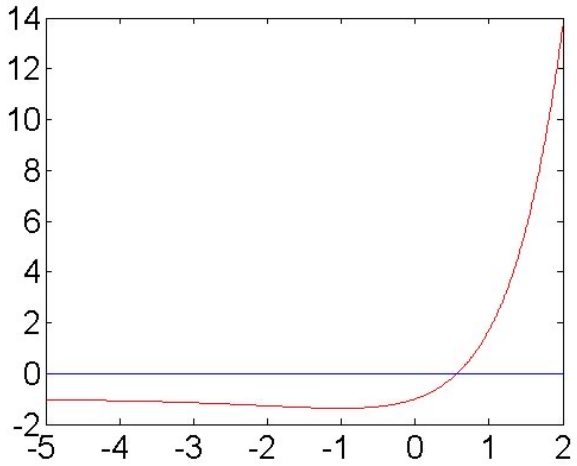
画出弦割法的流程图

例5. 用弦割法求方程 $xe^x - 1 = 0$ 在 $x = 0.5$ 附近的根（$\varepsilon = 10^{-4}$）

调换x_0,x_1有何影响？

解：取 $x_0 = 0.5, x_1 = 0.6$

由迭代公式求得下表



k	x_k	$x_k - x_{k-1}$
0	0.5	
1	0.6	
2	0.56532	-0.03468
3	0.56709	0.00178
4	0.56714	0.00001

故 $x^* \approx 0.56714$，满足精度要求.

见Matlab程序

chap1_example_5_bisection_newton_secant.m

三种求根算法的比较

- 二分法最稳妥，但是效率最低。
- 牛顿法效率最高，但是要求函数解析，容易计算导数。
- 弦割法是前面两种方法的折衷，既有较高的效率，又不必像牛顿法那样必须计算函数 f 的导数。 如果初始猜测比较接近待求解，则其收敛速度几乎与牛顿算法一样快。

如果待求解附近，函数的行为不好（如在x_0附近有拐点），则自动的牛顿法和弦割法都可能无法收敛或收敛到错误的结果。保险的做法是先用二分法初步的定出解的位置，再用两个自动方法中的一个定出解的精确位置。

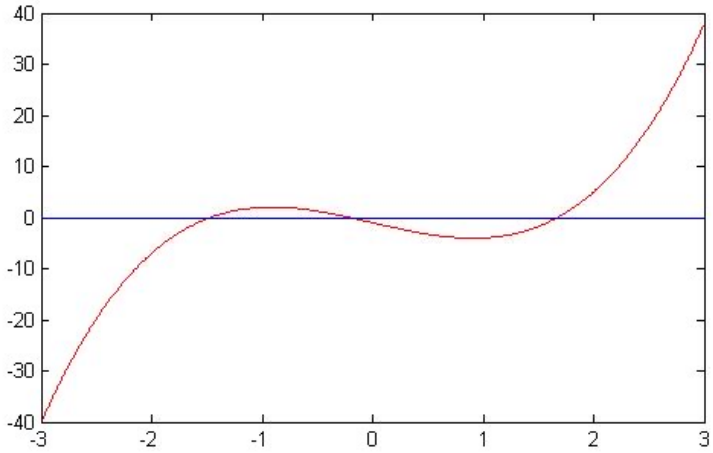
3.5 方程的多根问题

如果我们要求解的方程具有多个实根，如

$$2x^3 - 5x - 1 = 0$$

有3个实根。如何用数值方法求出所有3个根？

如何确定多个不同的有根区间？



逐步搜索法

设$f(x)=0$在$[a,b]$上有实根，选定步长h，从a开始依次扫描计算$x_k=a+kh$ $(k=0,1,2,\ldots)$处的函数值$f(x_k)$；若$f(x_k)$，则x_k即为方程的一个实根．

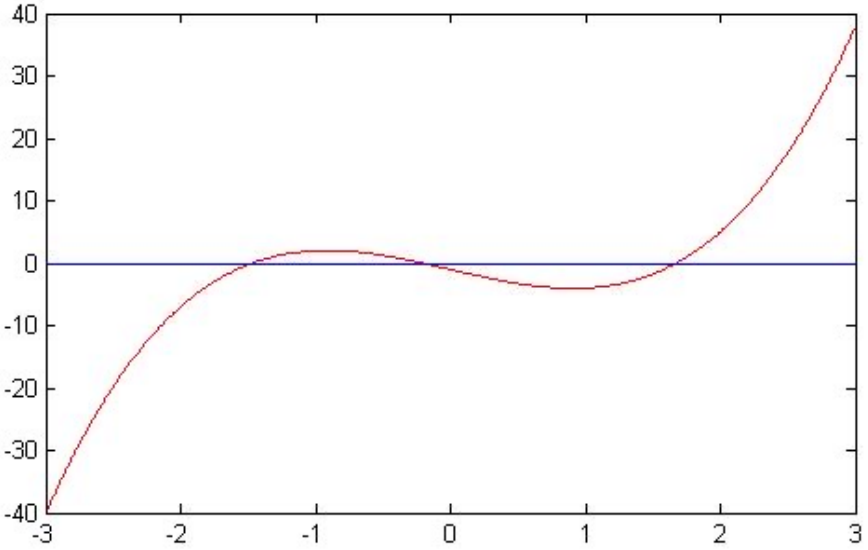
若某相邻两个点 $f(x_k)f(x_{k+1})<0$（异号），则说明在小区间 $[x_k, x_{k+1}]$ 内至少有一个实根；可取 x_k 或 x_{k+1} 或 $\frac{x_k + x_{k+1}}{2}$ 为方程的根 x^* 的近似值 .

只要步长足够小,就能得到任意精度的近似根,但步长越小,计算量越大.通常只用来确定根的大致范围.

因此，在求解多根问题时，我们可以利用逐步搜索法找出所有根的存在区间，然后利用二分法、Newton-Raphson法或弦割法求解精确值。

例6. 求解下述方程的所有实根

$$2x^3 - 5x - 1 = 0$$



见Matlab程序
chap1_example_6_several.m

Matlab自带的求根指令

（1）fzero：求单变量函数的零点

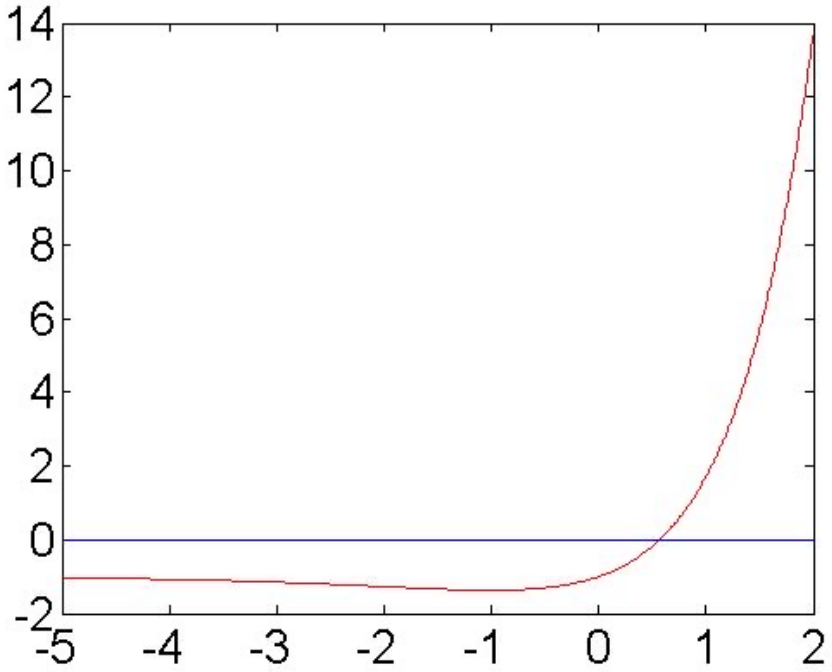
使用zeroin算法（结合了二分法、弦割法以及其它方法的一种综合方法）。

X=fzero(函数句柄，猜测的初始值或搜寻的区间)

```
f= @(x)sin(x);
fzero(f,[3,4])
```

例7. 利用Matlab自带指令求解下述方程在x=0.5附近的实根

$$xe^{x}-1=0$$



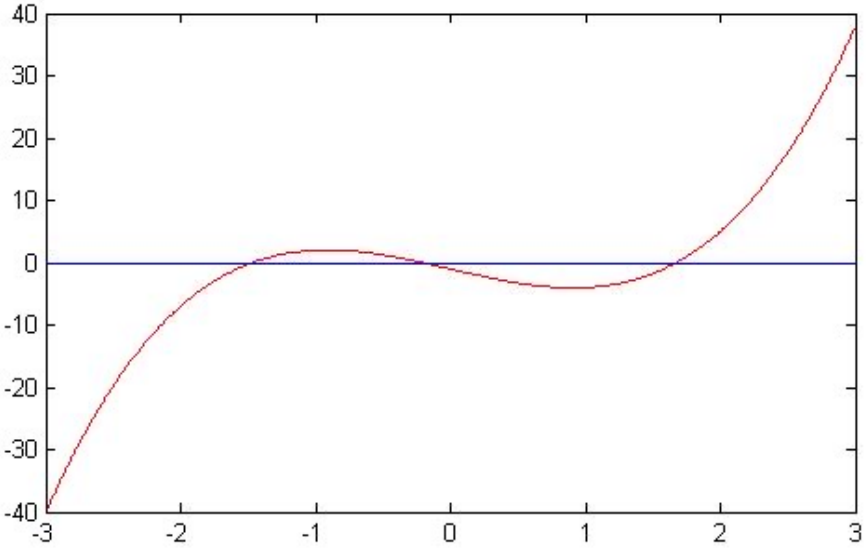
见Matlab程序
chap1_example_7_Matlab.m

（2）roots：求多项式的全部零点

roots(c)指令可以求出**多项式**的全部零点，其中c为各幂次项的系数。

例8. 用Matlab自带指令求解下述方程的所有实根

$$2x^3 - 5x - 1 = 0$$



见Matlab程序

chap1_example_8_several_Matlab.m

（3）fsolve：非线性方程组的数值求解

fsolve指令可以解出多变量的非线性方程组。

用法如下：x=fsolve(fun,x0)或[x,fval]=fsolve(fun,x0)

其中，x为方程的零点，x0为猜测的初始值，fun为要求解的非线性方程组，fval为零点位置对应的函数值

例9. 用Matlab自带指令求解下述非线性方程组

$$\begin{cases} 2\cos x + \sqrt{y} - \ln z = 7 \\ 2^x + 2y - 8z = -1 \\ x + y - \cosh z = 0 \end{cases}$$

求解步骤如下：

(1) 首先要编写一个函数文件，如fun.m，将方程组中的x,y,z看作一个矢量X的三个分量，函数值Y矢量中各个分量代表的含义为

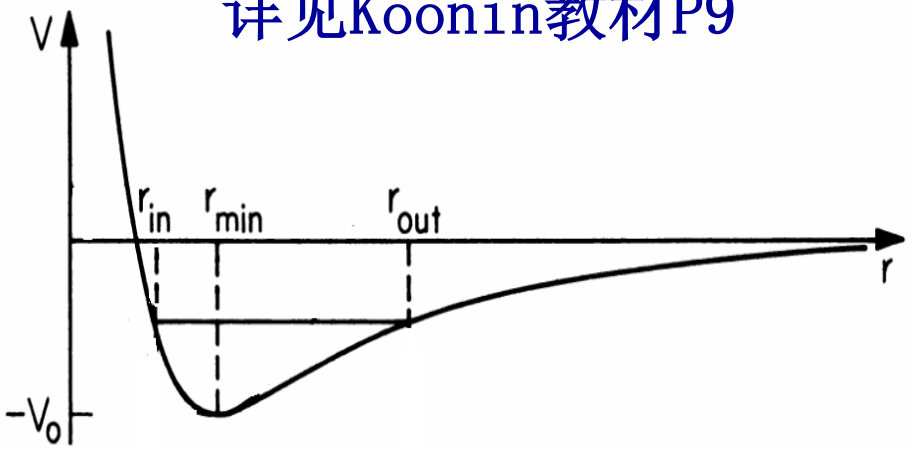
$$\begin{cases} Y(1) = 2\cos x + \sqrt{y} - \ln z - 7 \\ Y(2) = 2^x + 2y - 8z + 1 \\ Y(3) = x + y - \cosh z \end{cases}$$

(2) 写一个程序，给定零点猜测值，使用fsolve命令调用上述函数

结果见Matlab程序
chap1_example_9_Matlab.m

分子振动的半经典量子化

详见Koonin教材P9



原子的相互作用势为 *Lennard-Jones势*

$$V(r) = 4V_0[(\frac{a}{r})^{12} - (\frac{a}{r})^{6}]$$

势能最低处为$r_{min}=2^{1/6}a$，深度为$-V_0$

考虑一个双原子分子，其能量为E_n的相对运动的振动态可以用一维薛定谔方程的束缚态解$\Psi_n(r)$来描述

$$[-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}+V(r)]\psi_n = E_n\psi_n.$$

约化质量 $m=\frac{m_1m_2}{m_1+m_2}$

目标：对给定的势，求得能量E_n

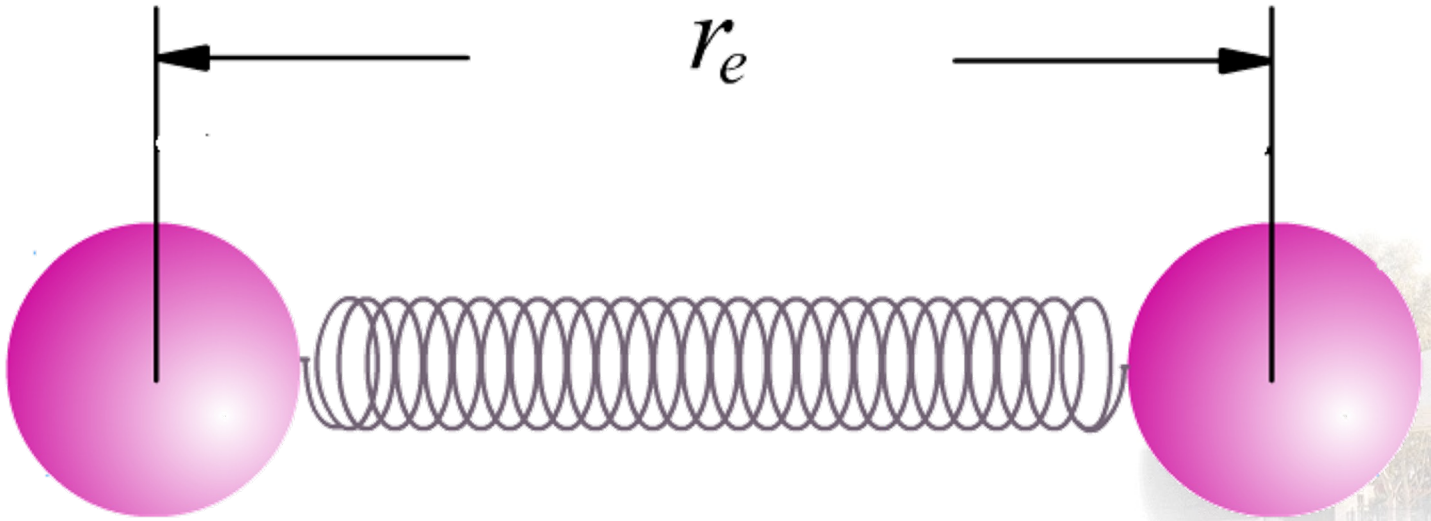
标准办法：数值求解常微分方程本征值问题，将在第三章讨论

我们这里采用的方法：半经典量子化

半经典量子化

通过考虑原子核在势场中做经典运动，然后应用量子化规则，可以定出其振动能 E_n。

这就是所谓的Bohr-Sommerfeld-Wilson量子化法则。



Bohr的量子化法则

Bohr 根据对应原理的思想得出了一个角动量量子化的条件，即电子运动的角动量 J 只能是 ħ 的整数倍

$$J = n\hbar, \quad n = 1, 2, 3, \ldots$$

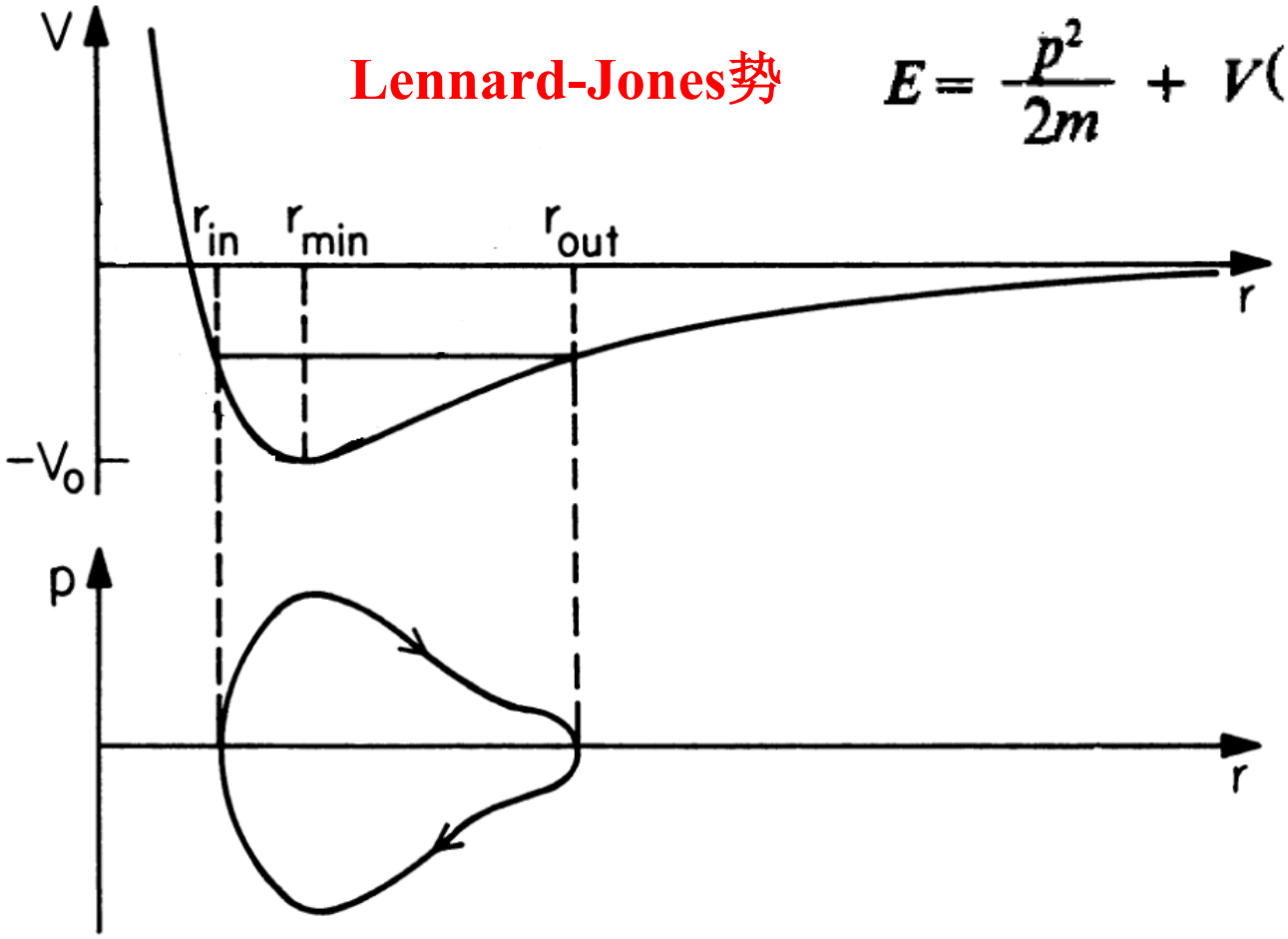
推广的量子化法则

Sommerfeld 等为处理多自由度体系的周期运动的能量量子化，给出了推广的量子化条件

$$\oint p_k dq_k = n_k h, \quad n_k = 1, 2, 3, \ldots$$

其中 q_k, p_k 代表一对共轭的正则坐标与动量，$\oint$ 代表对周期运动积分一个周期。

在相空间轨迹为



Lennard-Jones势

$$E = \frac{p^2}{2m} + V(r)$$

势场 V 中，原子核间距的经典的运动，可以在能量 $-V_0<E<0$上发生。原子核间距在 r_{in} 和 r_{out} 之间周期性的振动，对应相空间中的一条封闭轨道，轨道方程为

$$p(r)=\pm[2m(E-V(r))]^{1/2}$$

改进的Sommerfeld量子化条件为

$$\oint p(r)dr=(n+\frac{1}{2})2\pi\hbar$$

量子化条件为(n+1/2)h，而不是nh,是因为考虑到量子条件下振动的最小能量为 $\hbar\omega/2$

其中n为非负整数,即

$$S(E_n)=2[\frac{2m}{\hbar^2}]^{1/2}\int_{r_{in}}^{r_{out}}[E_n-V(r)]^{1/2}dr=(n+\frac{1}{2})2\pi$$

$$V(r) = 4V_0[(\frac{a}{r})^{12} - (\frac{a}{r})^{6}]$$

无量纲化标度

为了对 Lennard-Jones势确定量子化条件，定义几个无量纲量

$$\epsilon = \frac{E}{V_0},\ x = \frac{r}{a},\ \gamma = [\frac{2ma^2V_0}{\hbar^2}]^{1/2}$$

量子化条件变为

$$s(\epsilon_n) \equiv \frac{1}{2}S(\epsilon_n V_0) = \gamma \int_{x_{in}}^{x_{out}} [\epsilon_n - v(x)]^{1/2} dx = (n + \frac{1}{2})\pi$$

其中 $v(x) = 4[\frac{1}{x^{12}} - \frac{1}{x^6}]$ 是标度化后的位势，可参见

LJ_potential.m程序， n是非负整数。

若知道分子的转动惯量（可以从分子转动的能量得知）和离解能（把分子分解为组成它的两个原子所需的能量），就能从实验观测获得参量 a 和 V_0，从而定出γ 。

分子	H_2	HD	O_2
γ	21.7	24.8	150

程序结构

$$s(\epsilon_n) \equiv \frac{1}{2} S(\epsilon_n V_0) = \gamma \int_{x_{in}}^{x_{out}} [\epsilon_n - v(x)]^{1/2} dx = (n + \frac{1}{2})\pi$$

面临的问题是求解下面的方程

$$\gamma \int_{x_{in}}^{x_{out}} [\epsilon_n - v(x)]^{1/2} dx - (n + \frac{1}{2})\pi = 0 \qquad (1)$$

其中

$$v(x_{in}) - \epsilon_n = 0 \qquad (2)$$

$$v(x_{out}) - \epsilon_n = 0 \qquad (3)$$

面临的问题是利用上面三个方程求解四个未知数

数值求解此问题就是利用计算机强大的计算能力进行搜索对比，从而找到方程（1）的根。

构造函数

$$vh(\epsilon_n) = \gamma \int_{x_{in}}^{x_{out}} [\epsilon_n - v(x)]^{1/2} dx - (n + \frac{1}{2})\pi$$

其中 x_{in} = qiugen(ε_n–v(x)), x_{out} = qiugen(ε_n–v(x))

有

ε_n = qiugen(vh(x))

见Matlab程序
Project_1_semiclass.m

问题1

假设原子间相互作用势为谐振子势（抛物线势），利用半经典量子化条件，解析求系统能级。

$$S(E_n) = 2[\frac{2m}{\hbar^2}]^{1/2} \int_{r_{in}}^{r_{out}} [E_n - V(r)]^{1/2} dr = (n + \frac{1}{2})2\pi$$

$$V(r) = \frac{1}{2} m\omega^2 (r - r_{min})^2 - V_0$$

$$\Rightarrow \quad 2[\frac{2m}{\hbar^2}]^{1/2} \int_{r_{in}}^{r_{out}} [E_n - \frac{1}{2} m\omega^2 (r - r_{min})^2 + V_0]^{1/2} dr = (n + \frac{1}{2})2\pi$$

其中

$$r_{in} = r_{min} - \sqrt{2(E_n + V_0)/(m\omega^2)}$$

$$r_{out} = r_{min} + \sqrt{2(E_n + V_0)/(m\omega^2)}$$

$$\Rightarrow \quad E_n = V_0 + (n + \frac{1}{2})\hbar\omega$$

问题2

Page 13 习题1.12

对氢分子更适用的势函数为

$$V(r) = V_0[(1 - e^{-\beta(r-r_{min})})^2 - 1]$$

调节参量β，使得极小点上的二阶导数同观测到的H_2第一振动态的激发能相拟合。

第一次作业

1. 用数值法求解定积分 $\int_0^1 t^{-2/3}(1-t)^{-1/3}\,dt$,其精确解为 $2\pi/\sqrt{3}$

要求：分别用梯形公式、 Simpson公式求定积分的值，并考察不同h值对结果的精度影响。

（提示：把积分区域分为两部分，在每个积分中作不同的变量替换以处理奇点。）

2.分别用二分法、牛顿迭代法，弦割法以及Matlab自带指令求下面方程

$$f(x)=(x-1)[\sin(x-1)+3x]-x^3+1=0$$

在0.95附近的根.

3. 针对“振动的半经典量子化”课题，完成以下作业：

（1）阅读并理解程序

（2）（P12,习题1.11）运行程序，得到能谱，并画出最低能级ε_1与 γ 的关系（γ值从20运行到200），并解释得到的结果。

（3）（P12,习题1.7）修改程序，改用谐振子势计算能谱，并与解析结果比较。

（4）（P13,习题1.12）修改程序，改用Morse势计算能谱，比较并解释与LJ能量值的差异。

要求程序有对应注释，并配有对结果较详细的Word文档解释