数值计算模块三

数值积分与微分

目录

- 复习内容——函数指针做函数参数
- 数值积分
- 数值微分

函数的指针和指向函数的指针变量

一个函数在编译时被分配给一个入口地址，这个入口地址就称为函数的指针。可以定义一个指针变量指向函数，该指针称为指向函数的指针变量。

```
int (*p)(int,int);
```

定义p是指向函数的指针变量，它可以指向类型为整型且有个整型参数的函数。p的类型用int(*)(int,int)表示

```
int (*p)(int, int);
```

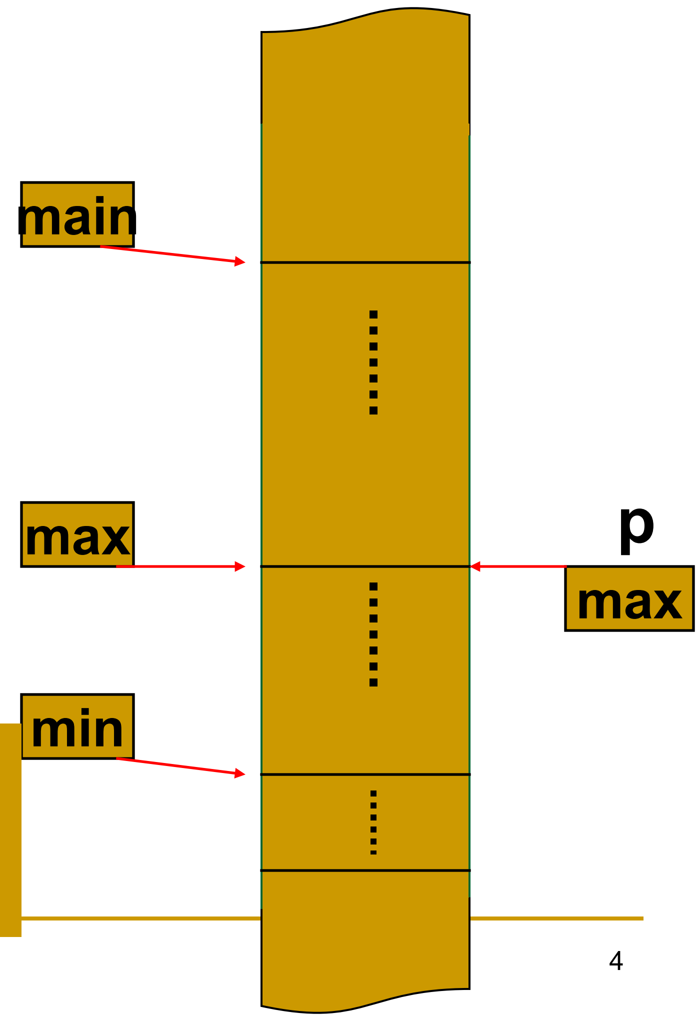
数据类型标识符　(*指针变量名) (函数参数表列)

函数返回值的类型

例如：

```
char  *p1,a='c';
int   (*p)  (int,int );
p1=&a;
p= max
```

(*p)



定义p1为指向字符串的指针变量

p指向一个带整型返回值的函数

用函数指针实现求整数a和b中的大者

```c
#include "stdio.h"

int main()
{
    int max(int,int);
    int (*p)(int,int);

    int a,b,c;
    p=max;
    printf("请输入a和b:");
    scanf("%d,%d",&a,&b);

    c=(*p)(a,b);

    printf("%d,%d,max=%d\n",a,b,c);
    return 0;
}
```

```c
int max(int x,int y)
{
    int z;
    if(x>y)  z=x;
    else     z=y;
    return(z);
}
```

注意：对函数指针变量p进行p++, p+n, p--等运算无意义。因为p只是函数入口地址，并不代表某一条语句地址，不能用p++指向下一条语句。

输入两个整数，然后让用户选择1或2，选1时调用max函数，输出二者中的大数，选2时调用min函数，输出二者中的小数

```
#include "stdio.h"

int main()
{
    int max(int,int);   int min(int x,int y);
    int (*p)(int,int);
    int a,b,c,n;
    scanf("%d,%d",&a,&b);
    scanf("%d",&n);
    if (n==1) p=max;
    else if (n==2) p=min;
    c=(*p)(a,b);
    printf("a=%d,b=%d\n",a,b);
    if (n==1) printf("max=%d\n",c);
    else  printf("min=%d\n",c);
    return 0;
}
```

```
int max(int x,int y)
{
    int z;
    if(x>y)  z=x;
    else     z=y;
    return(z);
}

int min(int x,int y)
{
    int z;
    if(x<y)  z=x;
    else     z=y;
    return(z);
}
```

□ 把指向函数的指针变量作函数参数

这是函数指针变量常用的用途之一。函数的参数可以是普通变量、指针变量、数组名等，也可以是函数指针。用函数指针作为函数的参数，实现函数地址的传递，即将函数名传给形参。

例如：

```
sub(int (*x1)(), int(*x2)();)
{int a,b,i=1,j=2;
  a=(*x1)(i);
  b=(*x2)(i,j);
}
```

这两条语句的含义？

调用函数f1和f2

main函数中，有下列语句

```
sub(f1,f2);
```

f1和f2为函数名。从程序中，是否可以判断函数f1，f2是否有参数？

把函数指针变量作为函数参数的好处：

当每次调用的函数不是固定的时候，用指针变量就比较方便。

函数指针举例：设一个函数process,调用它的时候，依据参数不同，每次实现不同的功能。输入a,b，分别求a,b中的大数，a,b中的小数和a,b之和。

```
main()
{int max(),min(),add();
int a,b;
scanf(“%d,%d”,&a,&b);
printf(“max=”);
process(a,b,max);
printf(“min=”);
process(a,b,min);
 printf(“sum=”);
process(a,b,add);
}
```

```
max(int x, int y)
{int z;
if (x>y) z=x;
else z=y; return(z);
}
```

```
min(int x, int y)
{int z;
if (x<y) z=x;
else z=y; return(z);
}
```

```
add(int x, int y)
{int z;
z=x+y;
return(z);
}
```

```
process(int x,y,(*fun)();)
{int result;
result=(*fun)(x,y);
printf(“%d\n”,result);
}
```

数值积分

- 关于积分，有***Newton-Leibniz***公式

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

- 但是，在很多情况下，还是要数值积分：

  1、函数由离散数据组成

  2、***F(x)***求不出

  3、***F(x)***非常复杂

数值积分

- 定义数值积分如下：是离散点上的函数值的线性组合

$$I_n(f)=\lim_{n->\infty}\sum_{i=0}^{n}a_i f(x_i)$$

不依赖显式的函数表达式

称为**积分系数**，与$f(x)$无关，与积分区间和积分点有关

例：

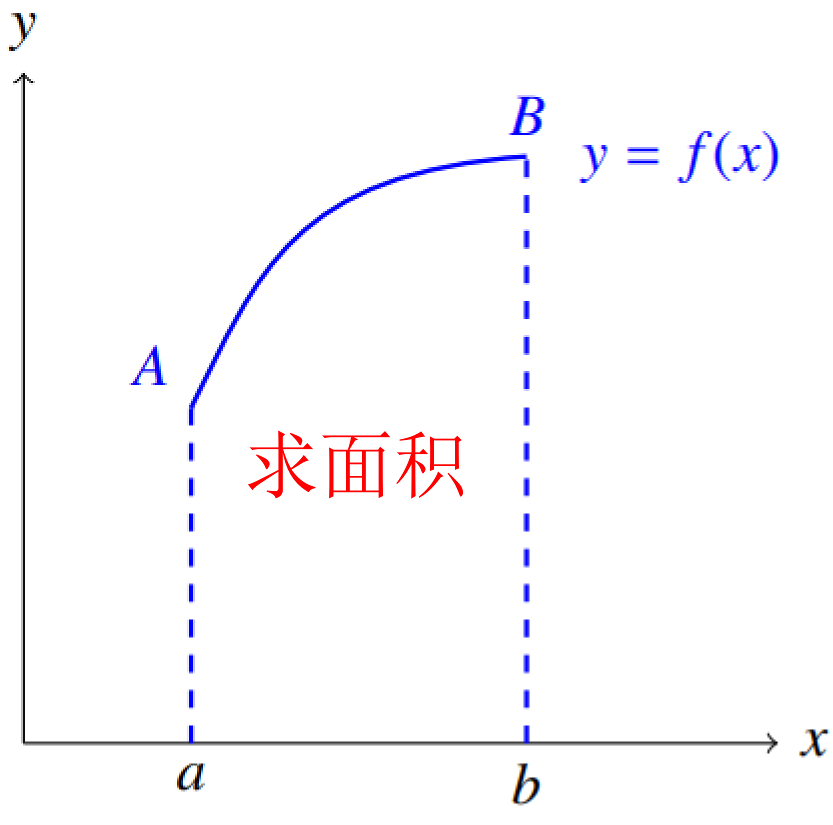
$$I_0(f)=(b-a)f(a)$$ 左矩形公式

$$I_0(f)=\left(b-a\right)f(b)$$ 右矩形公式

$$I_0(f)=(b-a)f(\frac{a+b}{2})$$ 中矩形公式

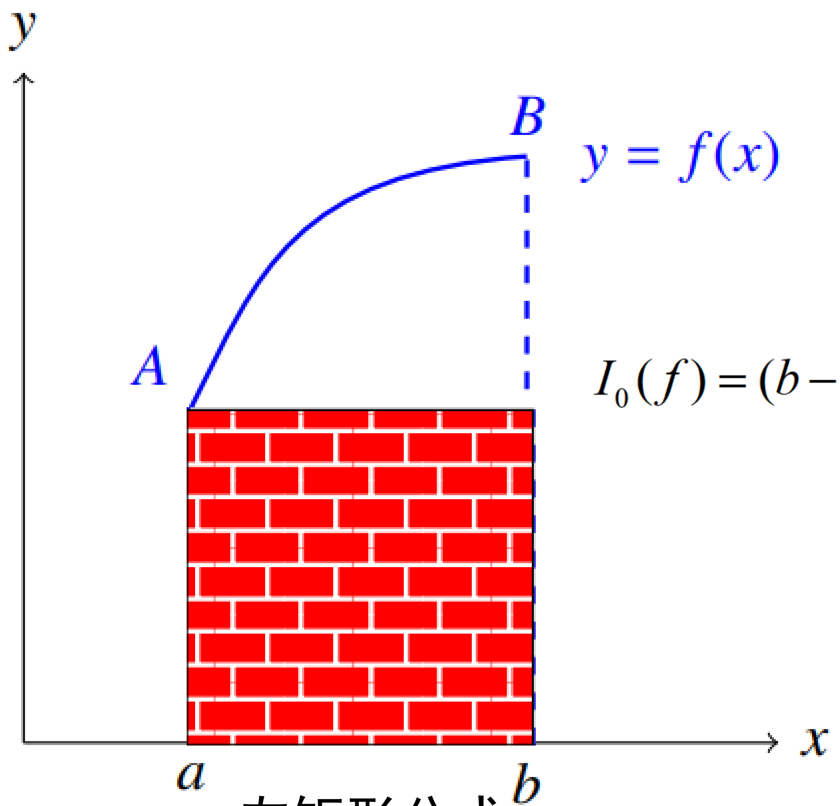
数值积分

- 数值积分的基本思想



数值积分

➤ 数值积分的基本思想



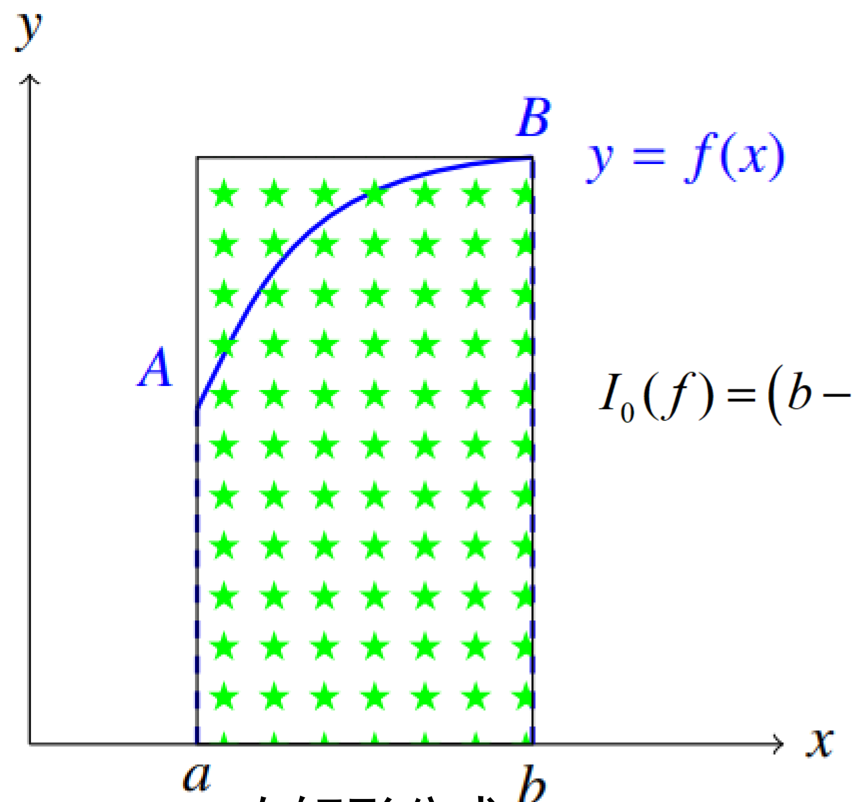
$$I_0(f)=(b-a)f(a)$$

左矩形公式

左矩形公式

数值积分

- 数值积分的基本思想



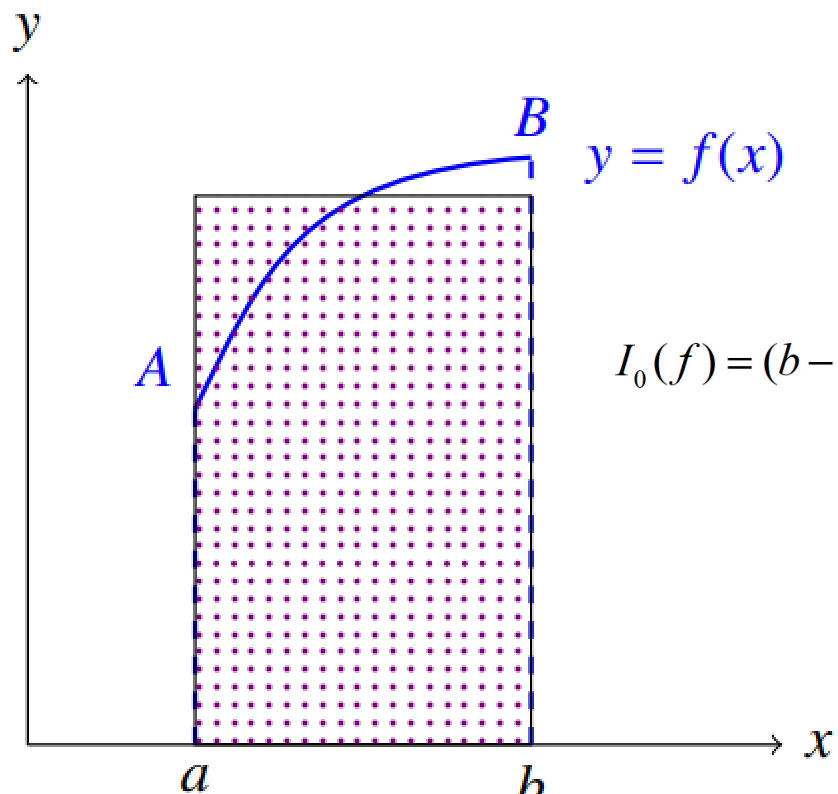
右矩形公式

$$I_0(f) = (b-a)f(b)$$

右矩形公式

数值积分

➢ 数值积分的基本思想



$$I_0(f) = (b-a)f(\frac{a+b}{2})$$ 中矩形公式

中矩形公式

数值积分

➢ 矩形法

矩形法是一种计算定积分近似值的方法，其思想是求若干个矩形的面积之和，这些矩形的高由函数值来决定。将积分区间[a, b] 划分为n个长度相等的子区间，每个子区间的长度为(a-b)/n 。这些矩形左上角、右上角或顶边中点在被积函数上。这样，这些矩形的面积之和就约等于定积分的近似值。

➢ 梯形法

为了计算出更加准确的定积分，采用梯形代替矩形计算定积分近似值，其思想是求若干个梯形的面积之和，这些梯形的长短边高由函数值来决定。这些梯形左上角和右上角在被积函数上。这样，这些梯形的面积之和就约等于定积分的近似值。

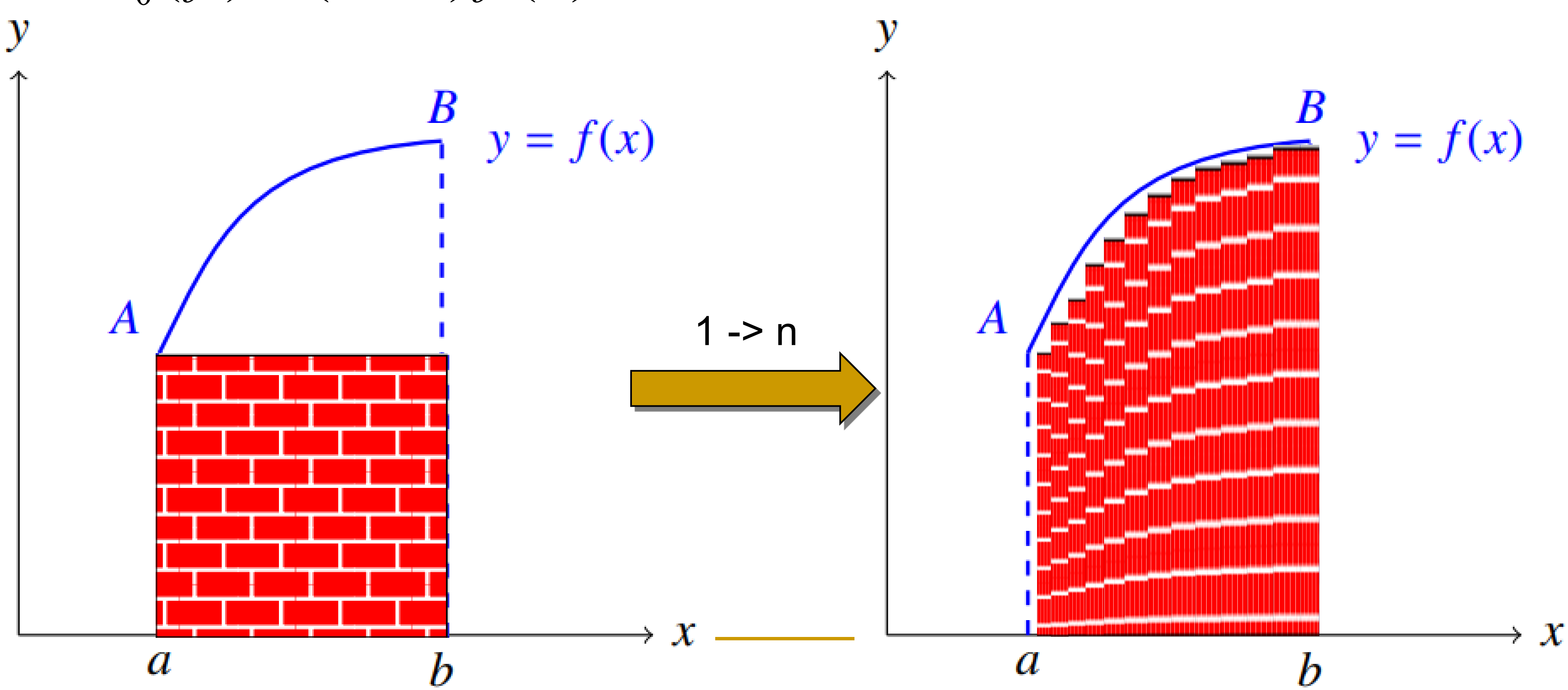
➢ 辛普森法

矩形法和梯形法都是用直线线段拟合函数曲线的方法，另一种形式是采用曲线段拟合函数，实现近似逼近的数值积分方法。辛普森法（Simpson’s rule）是以二次曲线逼近的方式取代矩形或梯形积分公式，以求得定积分的数值近似解。

数值积分

➢ **矩形法**

$$I_0(f) = (b-a)f(a)$$



数值积分

➢ **矩形法**

double rint1(a, b, n, f)

形参与函数类型	参数意义
double a	积分下限
double b	积分上限
int n	划分个数
double (*f)()	指向被积函数的指针

数值积分

- 函数实现（定长矩形法）

$$I_n(f)=\lim_{n->\infty}\sum_{i=0}^{n}a_i f(x_i)=\sum_{i=0}^{n}hf(a+ih)$$

```
double rint1(double a, double b,  int n,  double (*f)(double))
{

    double x, h, s;

    h = (b - a) / n;

    x = a;
    s = 0;

    for (int i = 1; i <= n; i++) {
        x = x + h;
        s = s + (*f)(x) * h;
    }

    return s;

}
```

数值积分

➢ 变步长矩形法

double rint2(a, b, eps, f)

形参与函数类型	参数意义
double a	积分下限
double b	积分上限
double eps	积分精度要求
double (*f)()	指向被积函数的指针

(1)将积分区间一等分，求出积分值；

(2)将每个小区间二等分，求出积分值；

(3)判断二等分前后积分值的差的绝对值，若满足精度要求则停止，否则转步骤(2)

数值积分

➢ 函数实现（变长矩形法）

```
double rint2(double a, double b,  double eps,  double (*f)(double))
{
    int n = 1;
    double x = a, h, s, s1, p;
    h = (b - a); s = (*f)(x)*h; p = eps + 1.0;
    while (p>=eps)
    {
        s1 = 0.0; x = a;
        for (int i = 1; i <= n; i++)
        {
            x = x + h;
            s1 = s1 + (*f)(x) * h;
        }
        p = fabs(s1 - s);
        s = s1;
        n += n;
        h = h / 2.0;
    }
    return s;
}
```

$$I_n(f) = \lim_{n->\infty}\sum_{i=0}^{n} a_i f(x_i)=\sum_{i=0}^{n} hf(a+ih)$$

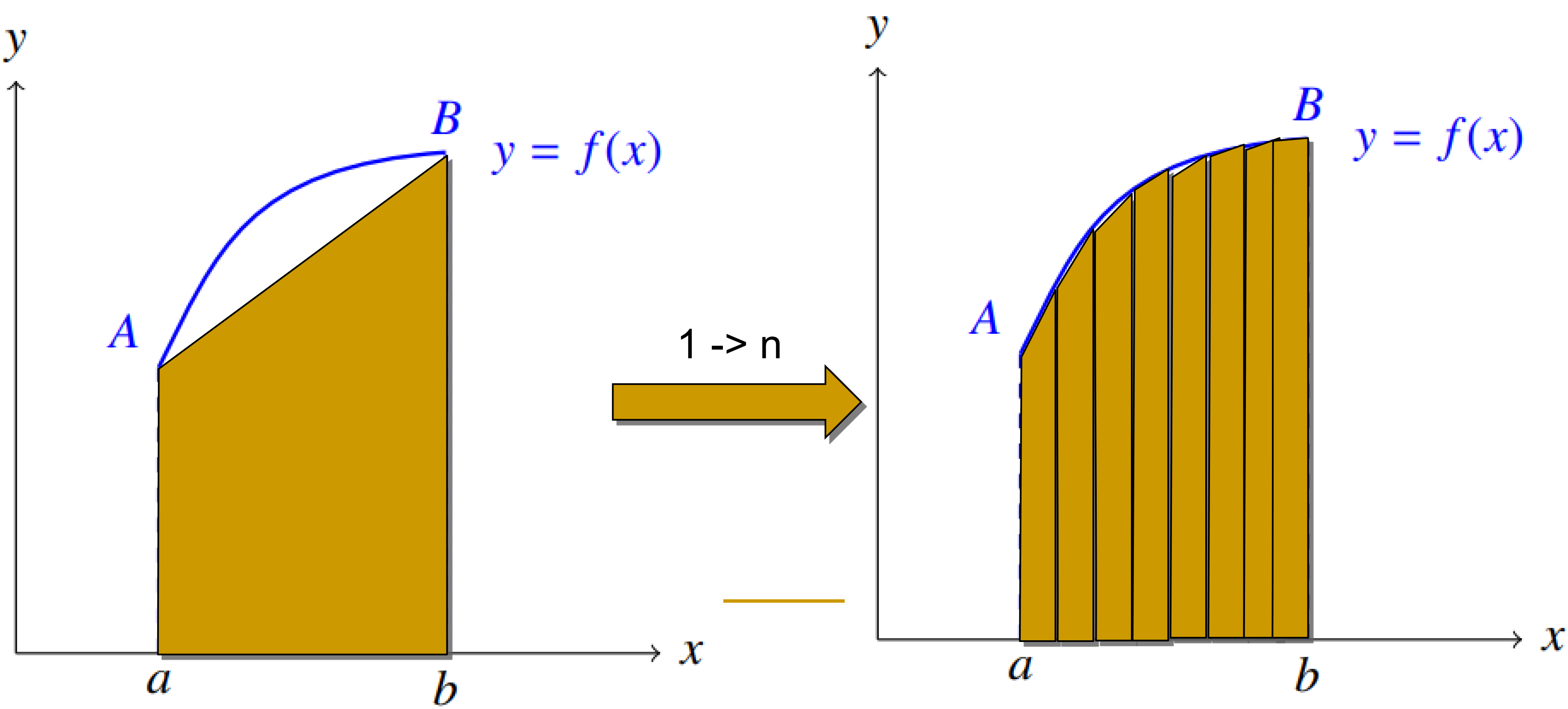
$$h = \frac{h}{2}$$

$$\left|I_n(f) - I_{2n}(f)\right| < \varepsilon$$

数值积分

- **梯形法**

$$I_1(f)=\frac{(b-a)}{2}\left[f(a)+f(\mathrm{b})\right]$$



数值积分

➢ **梯形法**

double tint1(a, b, n, f)

形参与函数类型	参数意义
double a	积分下限
double b	积分上限
int n	划分个数
double (*f)()	指向被积函数的指针

数值积分

➢ 函数实现（定长梯形法）

$$I_n(f)=\lim_{n\to\infty}\sum_{i=0}^{n}a_i f(x_i)=\sum_{i=0}^{n}\frac{h}{2}(f(a+ih)+f(a+(i+1)h))$$

```
double rint1(double a, double b,  int n,  double (*f)(double))
{
      double x, h, s, fa, fb;

      h = (b - a) / n;
      x = a;
      s = 0;

      for (int i = 0; i < n; i++) {
            fa = (*f)(x + i * h); fb = (*f)(x + (i + 1)*h);
            s = s +  h/2*(fa+fb);
      }

      return s;

}
```

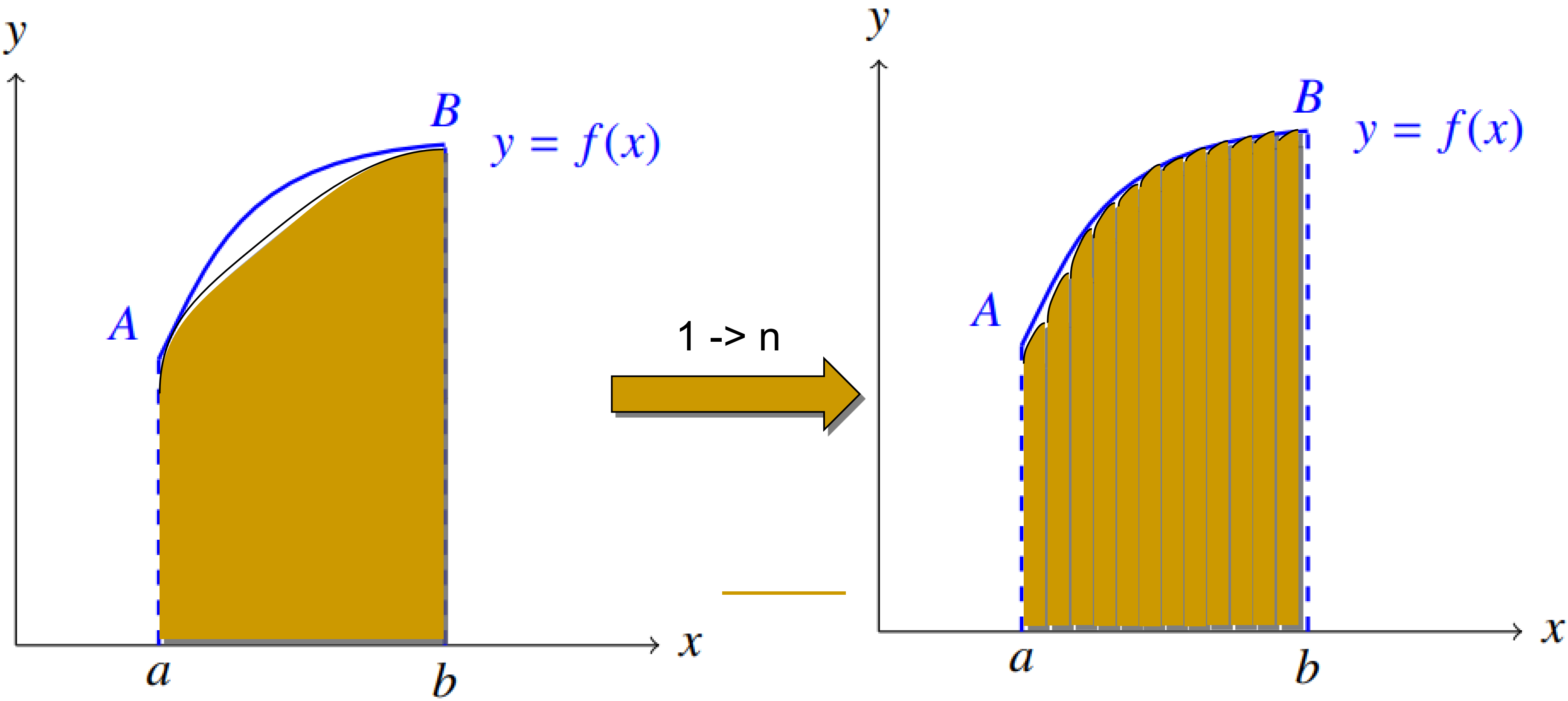
自行实现变步长梯形求积法

数值积分

➢ **辛普森法**

辛普森法则（Simpson's rule）是一种数值积分方法，是牛顿-寇次公式的特殊形式，以二次曲线逼近的方式取代矩形或梯形积分公式，以求得定积分的数值近似解。其近似值如下

$$I_2(f) \approx \frac{(b-a)}{6}\left[f(a)+4\,\mathrm{f}(\frac{a+b}{2})+f(\mathrm{b}) \right]$$



用基函数法构造：

$$l_i(x)=\frac{(x-x_0)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)},\quad i=0,1\cdots n$$

使满足： $$l_i(x_j)=\begin{cases}1, & i=j\\ 0, & i\neq j\end{cases}\quad \therefore L_n(x_j)=y_j$$

则 $$\boxed{L_n(x)=\sum_{i=0}^{n} y_i l_i(x)}$$ 即为

拉格朗日(Lagrange) 插值多项式

用拉格朗日插值法构造一个二次多项式 $P(x)$，使其通过这三个点 $(x_0, f(x_0))$、$(x_1, f(x_1))$ 和 $(x_2, f(x_2))$。

拉格朗日插值基函数为：

$$L_0(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}$$

$$L_1(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)}$$

$$L_2(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}$$

代入具体的点：

$$L_0(x) = \frac{(x - \frac{a+b}{2})(x - b)}{(a - \frac{a+b}{2})(a - b)} = \frac{(x - \frac{a+b}{2})(x - b)}{(\frac{a-b}{2})(a - b)} = \frac{(x - \frac{a+b}{2})(x - b)}{\frac{(a-b)^2}{2}}$$

$$L_1(x) = \frac{(x - a)(x - b)}{(\frac{a+b}{2} - a)(\frac{a+b}{2} - b)} = \frac{(x - a)(x - b)}{(\frac{-a+b}{2})(\frac{a-b}{2})} = \frac{(x - a)(x - b)}{\frac{(b-a)^2}{4}} = \frac{4(x - a)(x - b)}{(b - a)^2}$$

$$L_2(x) = \frac{(x - a)(x - \frac{a+b}{2})}{(b - a)(b - \frac{a+b}{2})} = \frac{(x - a)(x - \frac{a+b}{2})}{(b - a)(\frac{b-a}{2})} = \frac{2(x - a)(x - \frac{a+b}{2})}{(b - a)^2}$$

构造的二次多项式为：

$$P(x) = f(a)L_0(x) + f\left(\frac{a+b}{2}\right)L_1(x) + f(b)L_2(x)$$

计算 $P(x)$ 在区间 $[a,b]$ 上的积分：

$$\int_a^b P(x)\,dx = \int_a^b \left[f(a)L_0(x) + f\left(\frac{a+b}{2}\right)L_1(x) + f(b)L_2(x)\right]dx$$

分别计算每个基函数的积分：

1. **计算 $L_0(x)$ 的积分**:

$$\int_a^b L_0(x)\,dx = \int_a^b \frac{(x-\frac{a+b}{2})(x-b)}{\frac{(a-b)^2}{2}}\,dx$$

令 $h = \frac{b-a}{2}$，则积分区间变为 $[-h, h]$:

$$\int_{-h}^{h} \frac{(x-0)(x-(-h))}{\frac{(-2h)^2}{2}}\,dx = \int_{-h}^{h} \frac{x(x+h)}{2h^2}\,dx$$

展开并积分:

$$\int_{-h}^{h} \frac{x^2+hx}{2h^2}\,dx = \frac{1}{2h^2}\left[\int_{-h}^{h} x^2\,dx + h\int_{-h}^{h} x\,dx\right]$$

由于奇函数在对称区间上的积分为零:

$$\int_{-h}^{h} x\,dx = 0$$

所以:

$$\int_{-h}^{h} \frac{x^2}{2h^2}\,dx = \frac{1}{2h^2}\cdot\frac{2h^3}{3} = \frac{h}{3}$$

因此:

$$\int_a^b L_0(x)\,dx = \frac{h}{3} = \frac{b-a}{6}$$

2. **计算 $L_1(x)$ 的积分**:

$$\int_a^b L_1(x)\,dx = \int_a^b \frac{4(x-a)(x-b)}{(b-a)^2}\,dx$$

同样令 $h = \frac{b-a}{2}$，则积分区间变为 $[-h, h]$:

$$\int_{-h}^{h} \frac{4(x+h)(x-h)}{4h^2}\,dx = \int_{-h}^{h} \frac{(x^2-h^2)}{h^2}\,dx$$

展开并积分:

$$\int_{-h}^{h} \frac{x^2-h^2}{h^2}\,dx = \frac{1}{h^2}\left[\int_{-h}^{h} x^2\,dx - h^2\int_{-h}^{h} 1\,dx\right]$$

计算各部分:

$$\int_{-h}^{h} x^2\,dx = \frac{2h^3}{3}, \quad \int_{-h}^{h} 1\,dx = 2h$$

所以:

$$\frac{1}{h^2}\left(\frac{2h^3}{3} - 2h^3\right) = \frac{1}{h^2}\left(-\frac{4h^3}{3}\right) = -\frac{4h}{3}$$

因此:

$$\int_a^b L_1(x)\,dx = -\frac{4h}{3} = -\frac{2(b-a)}{3}$$

3. **计算 $L_2(x)$ 的积分**:

由于对称性，$\int_a^b L_2(x)\,dx = \int_a^b L_0(x)\,dx = \frac{b-a}{6}$

$$\boxed{\int_a^b P(x)\,dx = f(a)\cdot\frac{b-a}{6} + f\left(\frac{a+b}{2}\right)\cdot\left(-\frac{2(b-a)}{3}\right) + f(b)\cdot\frac{b-a}{6}}$$

数值积分

- 辛普森法

double sint1(a, b, n, f)

形参与函数类型	参数意义
double a	积分下限
double b	积分上限
int n	划分个数
double (*f)()	指向被积函数的指针

数值积分

➢ 函数实现（辛普森求积法）

```
double sint1(double a, double b,  int n,  double (*f)(double))
{

    double x, h, s, fa, fb, fab;

    h = (b - a) / n;
    x = a;
    s = 0;

    for (int i = 0; i < n; i++)
    {
        fa = (*f)(x + i * h); fb = (*f)(x + (i + 1)*h);
        fab = (*f)(x + i * h + h / 2);
        s = s + h / 6 * (fa +4*fab+ fb);
    }
    return s;

}
```

自行实现变步辛普森求积法

$$I_2(f) \approx \frac{(b-a)}{6}\left[f(a)+4\,\mathrm{f}\left(\frac{a+b}{2}\right)+f(\mathrm{b}) \right]$$

数值积分

- **例子**

计算下列定积分

$$T_1 = \int_0^1 e^{-x^2} dx$$

$$T_2 = \int_0^1 \frac{\ln(1+x)}{1+x^2} dx$$

$$T_3 = \int_{-1}^1 \frac{1}{1+25x^2} dx$$

数值积分

- 函数实现

```
double f1(double x)
{
    return exp(-x * x);
}
double f2(double x)
{
    return log(1.0 + x) / (1.0 + x * x);
}
double f3(double x)
{
    return 1 / (1 + 25 * x*x);
}
```

数值积分

➢ 函数实现

```
void main()
{
    printf("%e\t%e\t%e\n", rint1(0, 1, 50, f1), rint1(0, 1, 50,
    f2), rint1(-1, 1, 50, f3));

    printf("%e\t%e\t%e\n", tint1(0, 1, 50, f1), tint1(0, 1, 50,
    f2), tint1(-1, 1, 50, f3));

    printf("%e\t%e\t%e\n", sint1(0, 1, 50, f1), sint1(0, 1, 50,
    f2), sint1(-1, 1, 50, f3));
}
```

7.404784e-01	2.756274e-01	5.493406e-01
7.467996e-01	2.721617e-01	5.493406e-01
7.468241e-01	2.721983e-01	5.493603e-01

数值微分

- 但是，在很多情况下，还是要数值微分：

  1、函数由函数表给出

  2、$F(x)$函数的导数不易求取

- 用差商求取微分
- 用插值函数求取微分

数值微分

1. 用差商求微分

微积分学中，函数的导数是通过差商的极限来定义，表示函数在某点的瞬时变化率，即平均变化率的极限。**其几何意义为曲线的斜率。**

$$f'(x)=\begin{cases}\lim\limits_{h\to 0}\dfrac{f(x+h)-f(x)}{h}\\[2ex]\lim\limits_{h\to 0}\dfrac{f(x)-f(x-h)}{h}\\[2ex]\lim\limits_{h\to 0}\dfrac{f(x+h)-f(x-h)}{2h}\end{cases}$$

数值微分

对于一个在点 $x = x_0$ 处有 n 阶导数的函数 $f(x)$，其泰勒展开式可以表示为：

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k + R_n(x)$$

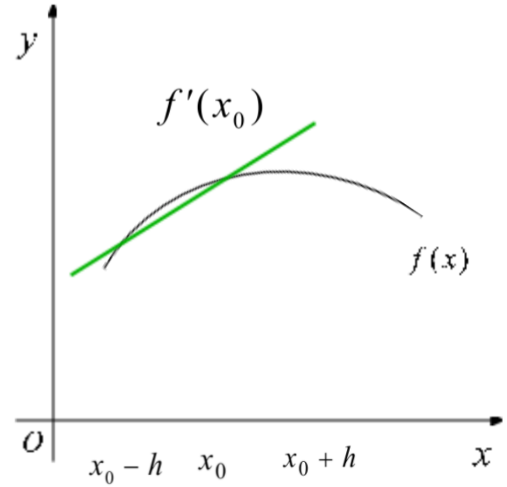
向前差商　$f(x_0 + h)$在$x = x_0$处泰勒展开

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

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

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

截断误差$O(h)$



向后差商　$f'(x_0) \approx \dfrac{f(x_0) - f(x_0 - h)}{h}$　$R(x) = \dfrac{h}{2!} f''(\xi)$　截断误差$O(h)$

中心差商　$f'(x_0) \approx \dfrac{f(x_0 + h) - f(x_0 - h)}{2h}$　$R(x) = -\dfrac{h^2}{6} f'''(\xi)$　截断误差$O(h^2)$

二阶差商　$f''(x_0) \approx \dfrac{f(x_0 + h) - 2f(x_0) + f(x_0 - h)}{h^2} + O(h^2)$　截断误差$O(h^2)$

数值微分

➢ 辛普森法

double CDF(double x, double eps, double (*f)())

形参与函数类型	参数意义
double x	求取导数的点
double eps	积分精度要求
double (*f)()	指向被积函数的指针

数值微分

- 函数实现

```
double CDF(double x,  double eps, double (*f)(double))
{

    double h = 0.4;

    double y1=(fun(x+h)-fun(x-h))/(2*h);

    double y2,temp;

    do {
    h /= 2;
    y2=(fun(x+h)-fun(x-h))/(2 *h);
    temp = y1;  y1= y2;
    } while(fabs(temp -y2)>eps);

    return y2;

}
```

数值微分

$$L_i(x)=\prod_{\substack{j=0\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}\qquad L_n(x)=\sum_{i=0}^{n}f(x_i)\cdot L_i(x)$$

2. 插值型求导公式

对于未知函数 $f(x)$ 的函数表，利用Lagrange方法建立插值函数 $L(x)$ 的，用插值函数的导数近似函数 $f(x)$ 的导数。

Lagrange插值

$$f(x)=L_n(x)+\frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}(x)$$

$[f^{(n+1)}(\xi)]'$ 难以求出

对 $f(x)$两边求导 $$f'(x)=L_n'(x)+\boxed{\frac{[f^{(n+1)}(\xi)]'}{(n+1)!}}\omega_{n+1}(x)+\frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}'(x)$$

在节点处$(x=x_k)$, $f'(x_k)$可以求出

$$f'(x_k)=L_n'(x_k)+\frac{[f^{(n+1)}(\xi)]'}{(n+1)!}\underbrace{\omega_{n+1}(x_k)}_{=0}+\frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}'(x_k)$$

$$=\boxed{L_n'(x_k)}+\boxed{\frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}'(x_k)}$$

求导公式的误差

- 插值型求导公式

$$\boxed{\omega_{n+1}(x)=(x-x_0)(x-x_1)\cdots(x-x_n)}$$

实验内容

(1)编写用矩形法、梯形法和辛普森法求解定积分

$$\int_0^3 \frac{x}{1+x^2}dx \quad 和 \quad \int_0^2 1+x^2 dx$$

并对三种算法的运行结果进行定性和定量的分析。

(2)请查阅文献整理求数值积分的其它方法，并至少编程实现其中的一种算法。结合（1）的结果对比分析几种算法的性能。

(3)请查阅文献整理求数值微分的其它方法，并至少编程实现其中的一种算法。

作业：提交实验报告