拉格朗日插值及牛顿插值算法的实现

第七小组：任浩辰　刘阳

科目：数值分析

日期：2020 年 4 月 5 日

1　实验目的

通过编程实践，熟练掌握拉格朗日插值算法和牛顿插值算法，对比二者的不同。并观察不同节点数量，不同拟合函数对于结果的影响

2　实验步骤

1. 验证拉格朗日插值算法对于不同函数的插值效果
2. 验证随着插值结点的增多插值曲线的变化情况
3. 验证插商的基本性质
4. 比较拉格朗日插值与牛顿插值的插值结果

3　实验内容

3.1　运用积函数法求拉格朗日问题

先构造一组基函数

$$\begin{aligned} l_i(x) &= \frac{(x-x_0)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)} \\ &= \prod_{j=0 j\neq i}^{n} \frac{x-x_j}{x_i-x_j}, i=0,1,\ldots,n \end{aligned} \tag{1}$$

$l_i(x)$ 是 n 次多项式，满足

$$l_i(x) = \begin{cases} 0 & j \neq i \\ 1 & j = i \end{cases}$$

令

$$p_n(x) = \sum_{i=0}^{n} y_i l_i(x) = \sum_{i=0}^{n} y_i (\prod_{j=0 j\neq i}^{n} \frac{x-x_j}{x_i-x_j})$$

上式称为 n 次拉格朗日插值多项式，其插值余项为

$$R_n(x) = f(x) - p_n(x) = \frac{f^{(n+1)}(\epsilon)}{(n+1)!} \sum_{i=0}^{n} (x-x_i), \epsilon \in (a,b)$$

可以用作误差估计上限。

3.2 牛顿插值

n 次牛顿插值多项式 $p_n(x)$ 表达式如下

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

其中，称

$$\frac{f[x_0, \ldots x_{k-1}] - f[x_1, \ldots, x_k]}{x_0 - x_k}$$

为函数 f(x) 关于点 $x_0, x_1, \ldots, x_k$ 的 k 阶差商，记为

$$f[x_0, \ldots, x_k] = \frac{f[x_0, \ldots x_{k-1}] - f[x_1, \ldots, x_k]}{x_0 - x_k}$$

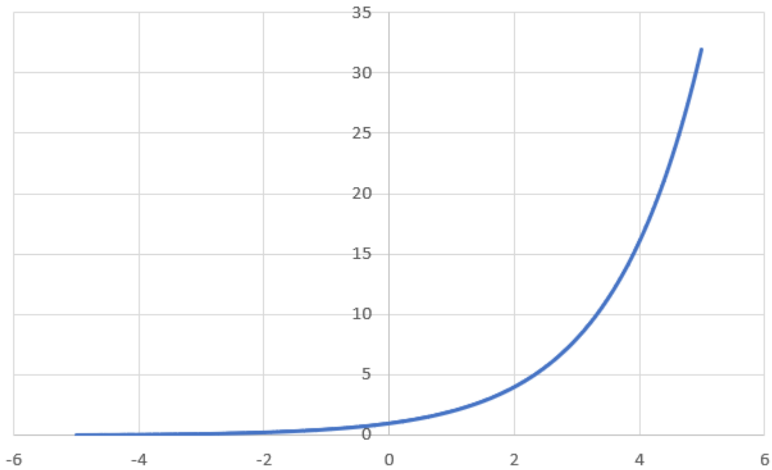
牛顿插值的余项公式与拉格朗日插值相同。

4 实验过程

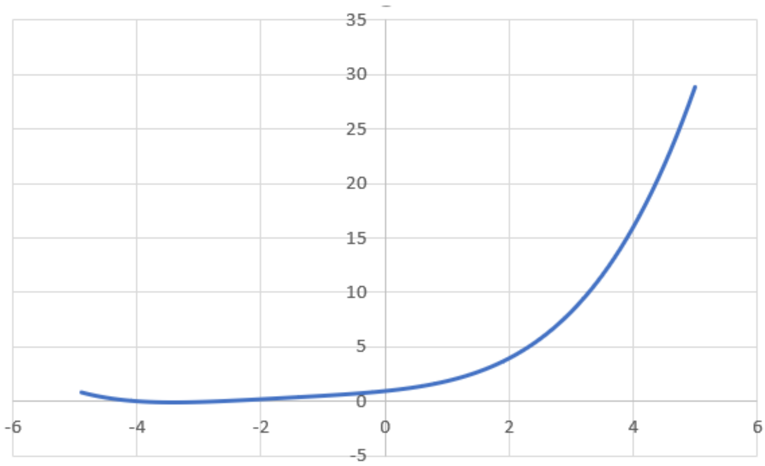
4.1 验证拉格朗日插值算法对不同函数的实现效果

1. 指数函数

首先，运用 excel 软件画出指数函数 $f(x) = 2^x$ 的图像。这里截取了 [-5,5] 区间的图像，每个点之间的间隔为 0.01。结果如图所示：



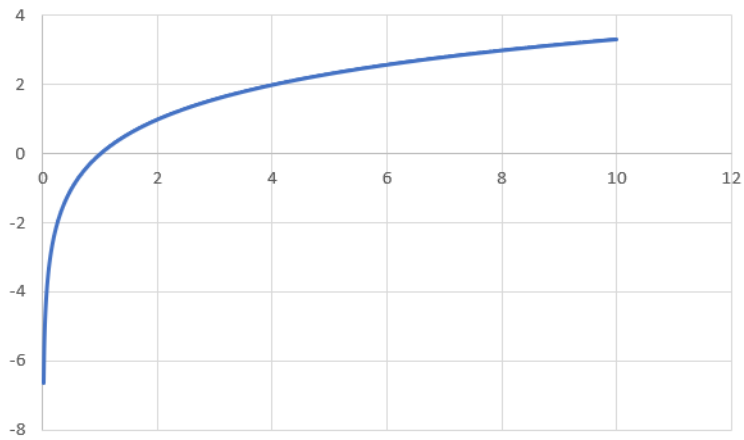
再使用拉格朗日插值算法，利用点 x=-4,x=-2,x=0,x=2,x=4 处的值，计算其拉格朗日 4 次表达式。在程序中，共计算了 [-5,5] 之间的 100 个点，每个点之间的间隔为 0.1，并将结果输出到 csv 文件中，最后用 excel 做出其图像，结果如图所示：



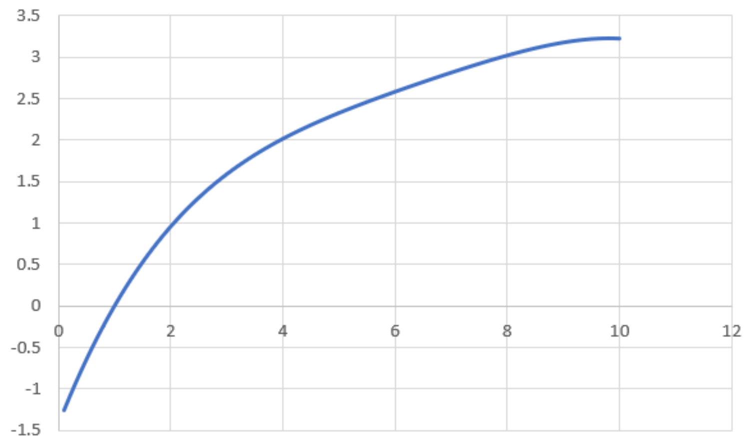
从图中可以看出，在 [-5,-2] 区间内出现了误差偏大的情况，其余位置拟合曲线都较为接近。

2. 对数函数

首先使用 excel 做出函数 $f(x) = \log_2 x$ 的图像。这里截取了 (0,10] 区间的图像，每个点之间的间隔为 0.01. 结果如图所示：



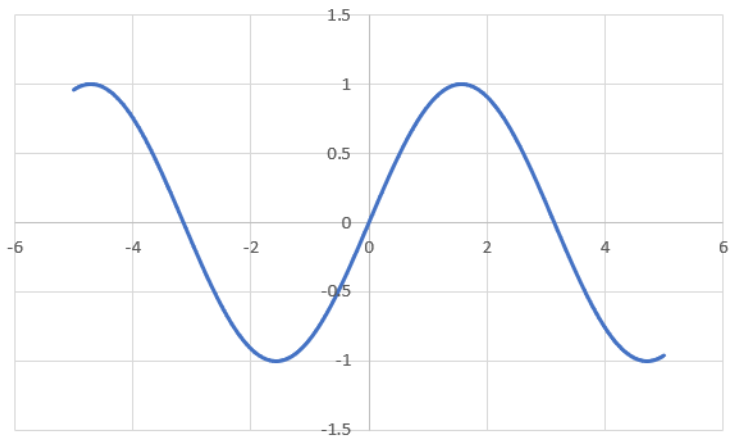
再使用拉格朗日插值算法，利用点 x=1,x=3,x=5,x=7,x=9，共 5 个点处的值，计算其拉格朗日 4 次表达式。在程序中，共计算了 (0,10] 之间的 100 个点，每个点之间的间隔为 0.1，并将结果输出到 csv 文件中，最后用 excel 做出其图像，结果如图所示：



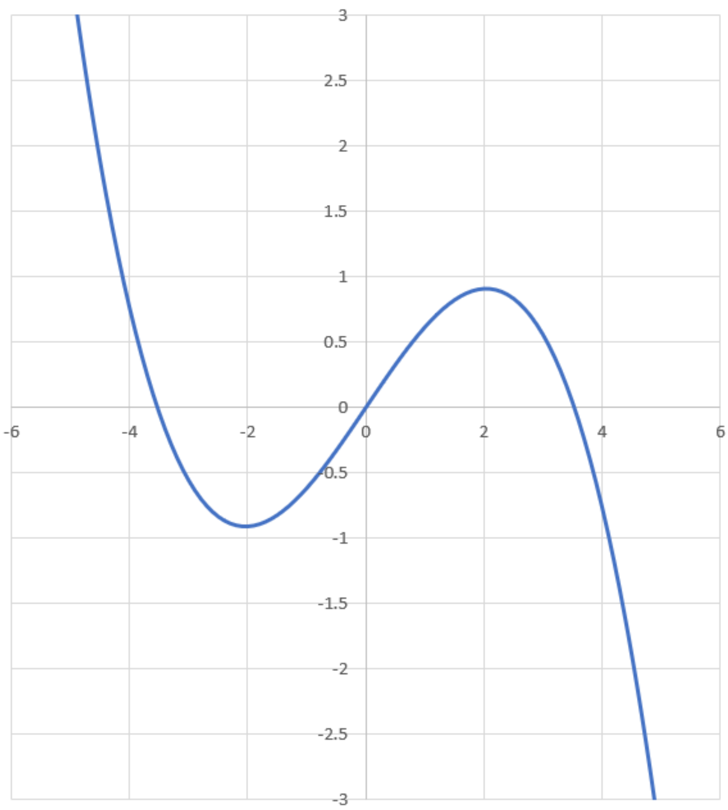
可以看出，拉格朗日插值在结点个数相同时，拟合对数函数的相似度比指数函数低。

3. 三角函数

首先使用 excel 做出函数 $f(x) = sinx$ 的图像。这里截取了 [-3,3] 区间的图像，每个点之间的间隔为 0.01. 结果如图所示：



再使用拉格朗日插值算法，利用点 x=-4,x=-2,x=0,x=2,x=4，共 5 个点处的值，计算其拉格朗日 4 次表达式。在程序中，共计算了 [-5,5] 之间的 100 个点，每个点之间的间隔为 0.1，并将结果输出到 csv 文件中，最后用 excel 做出其图像，结果如图所示：



综上所述，在已知结点个数相同，拉格朗日插值多项式次数相同的条件下，拟合情况最好的是指数函数，对数函数和三角函数较差。

4.2 随插值结点增多，插值曲线的变化情况

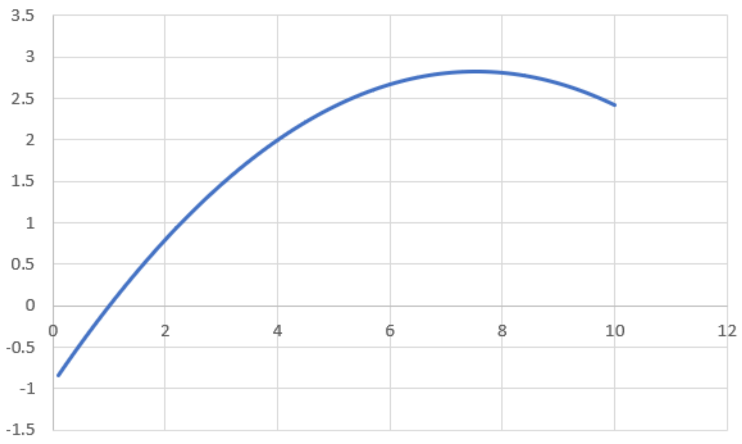
原函数 $f(x) = \log_2 x$，选取的结点个数与其对应的拟合结果如下

1. 选取的结点为三个时

结点数据如下表：

x	1	4	7
y	0	2	2.807354922

其拟合图像如图所示

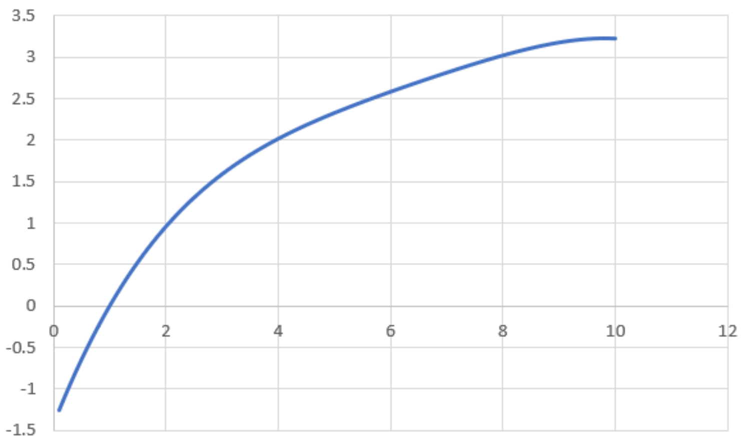


2. 选取的结点为五个时

结点数据如下表：

x	1	3	5	7	9
y	0	1.584963	2.321928	2.807355	3.169925

其拟合图像如图所示



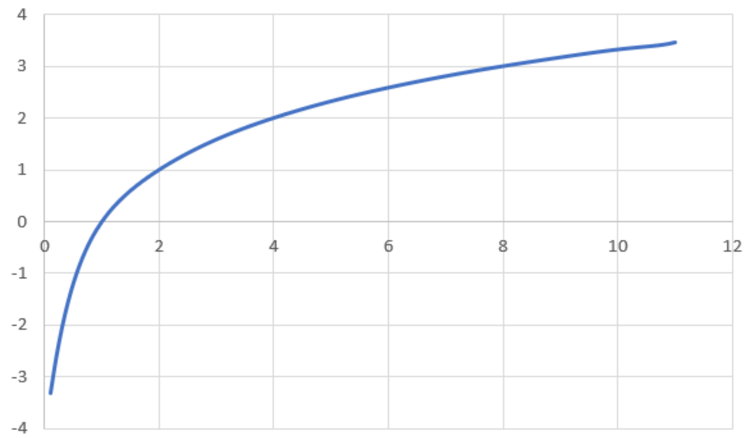
3. 选取的结点为 12 个时

结点数据如下表：

x	0.1	1	2	3	4	5
y	-3.32193	0	1	1.584963	2	2.321928

6	7	8	9	10	11
2.584963	2.807355	3	3.169925	3.321928	3.459432

其拟合图像如图所示



对比三种插值情况可以看出，对于同一函数，插值节点越多，拉格朗日插值多项式次数越高，拟合结果更精确。

4.3 验证差商的基本性质

1. 性质一

k 阶差商可以表示为 k+1 个函数值 $f(x_0), f(x_1)...f(x_k)$ 的线性组合，即

$$f[x_0, ..., x_k] = \sum_{j=0}^{k} \frac{f(x_j)}{(x_j - x_0)...(x_j - x_{j-1})(x_j - x_{j+1})...(x_j - x_k)}$$

证明过程如下：

使用归纳法。当 k=1 时，有

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

设当 k=n 时也成立，即

$$f[x_0, ..., x_n] = \sum_{j=0}^{n} \frac{f(x_j)}{(x_j - x_0)...(x_j - x_{j-1})(x_j - x_{j+1})...(x_j - x_n)}$$

$$f[x_0, ..., x_{n+1}] = \sum_{j=0}^{n+1} \frac{f(x_j)}{(x_j - x_0)...(x_j - x_{j-1})(x_j - x_{j+1})...(x_j - x_{n+1})}$$

则由差商的定义可得，当 k=n+1 时，有下面的等式成立：

$$\begin{aligned} f[x_0, ..., x_{n+1}] &= \frac{f[x_0, ..., x_n] - f[x_1, ..., x_{n+1}]}{x_0 - x_{n+1}} \\ &= \sum_{j=0}^{n+1} \frac{f(x_j)}{(x_j - x_0)...(x_j - x_{j-1})(x_j - x_{j+1})...(x_j - x_{n+1})} \end{aligned} \tag{3}$$

即可证明性质 1。

2. 性质二

k 阶差商关于结点 $x_0, x_1, ..., x_k$ 是对称的，或者说商差值与结点顺序无关。

证明过程如下：

由性质一可以得到等式

$$f[x_0, ..., x_k] = \sum_{j=0}^{k} \frac{f(x_j)}{(x_j - x_0)...(x_j - x_{j-1})(x_j - x_{j+1})...(x_j - x_k)}$$

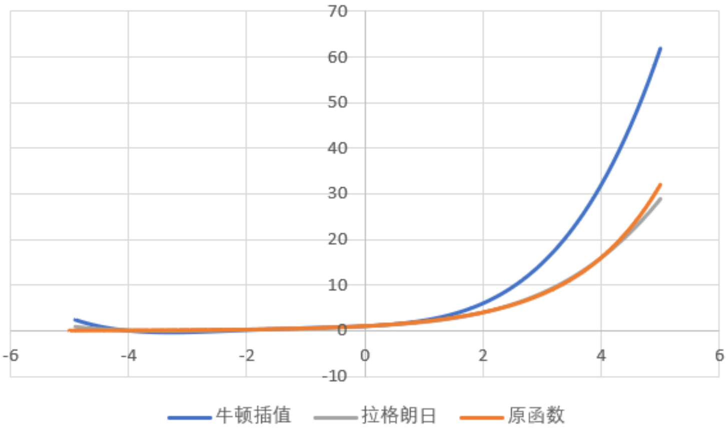
从等式中可以看出，k 阶差商的值与结点顺序无关，只与结点的值有关，性质二得证。

4.4 比较拉格朗日插值与牛顿插值结果

将 $f(x) = 2^x$ 作为原函数，已知结点数据如下表

x	-4	-2	0	2	4
y	0.0625	0.25	1	4	16

分别作拉格朗日插值和牛顿插值，并计算出 [-5,5] 区间内的 100 个点，点之间的间隔为 0.1，将拟合结果画出图像，结果如图所示：



从图中可以看出，对于原函数是 $f(x) = 2^x$ 的指数函数来说，在已知结点相同的情况下，拉格朗日插值拟合的结果要比牛顿插值更加接近原函数。

但在另一方面，拉格朗日插值法的线性插值的计算过程没有继承性，即增加一个节点时整个计算工作必须重新开始。而牛顿插值中，由于差商的计算能够递推，就避免了这一问题，这样大量的节省了乘、除法运算次数，减少了计算的时间。

因此，针对不同的实际情况，可以选择性的使用不同的插值算法。牛顿插值法适用于原函数不改变，而被插的点不断增多的情况，随着被插点增多，我们对函数的拟合精度也变高，这种方法可以很好的复用以前的计算结果，简化计算过程。而拉格朗日插值法适用于已经确定结点，不再改变的情况。

5 核心程序及算法

5.1 拉格朗日插值算法计算

代码如下：

```
/**
* 用拉格朗日插值算法计算某点的函数估计值
* @param x0 已知结点的x坐标
* @param y0 已知结点的y坐标
* @param n 已知结点数量
* @param x 要求的点的x坐标
*@return
*/
double lagrange(const double x0[], const double y0[], int n, double x){
  double ans = 0;//保存结果
  double temp = 1;//保存基函数

  for (int i = 0; i < n; i ++){
    for (int j = 0; j < n; ++j) {
      if(j != i){
        temp = temp * (x - x0[j]) / (x0[i] - x0[j]);//基函数计算
      }
    }
    ans += y0[i] * temp;//基函数与f(x)值的乘积求和
    temp = 1;//归1继续计算下一个基函数
  }
  return ans;
}
```

5.2 牛顿插值算法计算

代码如下：

```
/**
* 利用差商性质1计算差商
* @param x0 已知结点x坐标
* @param y0 已知结点y坐标
* @param n 要求差商的次数
* @return n次差商
*/
double differenceQuo(const double x0[], const double y0[], int n){
  double ans = 0;
  double temp = 1;

  for (int i = 0; i < n; ++i) {
    for (int j = 0; j < n; ++j) {
      if (j != i){
```

```
        temp *= 1 / (x0[i] - x0[j]);
      }
    }
    temp *= y0[i];
    ans += temp;
    temp = 1;
  }

  return ans;
}

/**
* 用牛顿插值法计算某一点的函数估计值
* @param x0 已知结点x坐标
* @param y0 已知结点y坐标
* @param n 已知结点数量
* @param x 要求的点的x坐标
* @return
*/
double newton(const double x0[], const double y0[], int n, double x){
  double ans = 0;//保存计算结果
  double temp = 1;//保存牛顿插值中的每一项

  for (int i = 0; i < n; ++i) {
    for(int j = 0;j < i; ++j){
      temp *= (x - x0[j]);
    }
    temp *= differenceQuo(x0,y0,i);//与差商相乘
    ans += temp;//求和
    temp = 1;//归1进行下一轮计算
  }
  return ans;
}
```

6 实验结果

1. 拉格朗日插值算法针对不同函数的拟合情况：实验中分别使用拉格朗日插值算法，拟合了指数函数、对数函数和三角函数，并且保持结点数量相同，均为 5 个。其中，指数函数的拟合效果最好，对数函数和三角函数稍差。

2. 不同节点数量对拟合结果的影响：原函数采用了对数函数 $f(x) = log_2 x$，当结点个数分别为 3，5，12 个时，比较拟合图像，可以非常直观的体现出，当结点数量增多时，拟合效果更好。

3. 证明差商的性质：实验中成功证明了差商的两条性质，运用了数学归纳法等方法，简化计算过程并成功推出正确结果。

4. 对比拉格朗日插值算法和牛顿插值算法拟合结果：原函数采用了指数函数 $f(x) = 2^x$，实

验结果表明在选取结点数为 5 个时，拉格朗日拟合效果较好，但同时牛顿插值也在结点数量不确定等情况下，具有更好的计算优势。

7 实验体会

在这次的实验过程中，利用 c++ 编写了拉格朗日插值算法和牛顿插值算法的有关代码，增加了编程能力，也进一步理解了这两种算法。

体会到了两种算法在不同条件下的不同结果。实验结果表明，在插值拟合时，应该尽量选取尽可能多的点，使拟合的结果更精确；在选择算法时一定要考虑现实情况，如果结点个数不确定时，可以选择牛顿插值算法，这样可以有效利用计算结果，简化计算过程，节省时间。