数值分析上机习题报告 (2)

张宏毅 1500017736

March 5, 2017

Problem

对 Runge 函数 $R(x)=\frac{1}{1+x^2}$ $(x\in[-5,5])$，利用下列条件作插值逼近，并与 $R(x)$ 的图像作比较：

（1）用等距节点 $x_i=-5+i$ $(i=0,1,2,\cdots,10)$，绘出它的 10 次 Newton 插值多项式的图像；

（2）用节点 $x_i=5\cos\left(\frac{2i+1}{42}\pi\right)$ $(i=0,1,2,\cdots,20)$，绘出它的 20 次 Lagrange 插值多项式的图像。

Solution

条件一为等距节点插值，选取了 $x=-5,-4,\cdots,4,5$ 共 11 个节点。Newton 插值的基本算法为，利用差商的定义式

$$f[x_i,x_{i+1},\cdots,x_{i+j}]=\frac{f[x_i,x_{i+1},\cdots,x_{i+j-1}]-f[x_{i+1},x_{i+2},\cdots,x_{i+j}]}{x_i-x_{i+j}}$$

递推出所有的系数 $f[x_0,x_1,\cdots,x_m]$ $(0\le m\le n)$，进而可直接写出 Newton 插值多项式

$$N_n(x)=f[x_0]+\sum_{m=1}^{n}f[x_0,x_1,\cdots,x_m](x-x_0)(x-x_1)\cdots(x-x_{m-1}).$$

在实际的多项式计算中，可利用 Horner 法则来减少加法和乘法运算的总次数，即利用已知的初始条件 $p_n(x)=f[x_0,x_1,\cdots,x_n]$ 及递推公式 $p_k(x)=(x-x_{k+1})p_{k+1}(x)+f[x_0,x_1,\cdots,x_k]$ $(0\le k\le n-1)$，来简化求解多项式 $N_n(x)=p_0(x)$ 的过程。具体代码（Python）如下：

```
# Newton's Interpolation method
# Input: Data points, corresponding values and number of points
# Output: Interpolation polynomial

def NewtonInterp(x, y, n):
   f = [[0 for i in range(n)] for j in range(n)]
   for i in range(0, n):
      f[i][i] = y[i]

   # Compute divided differences
   for j in range(1, n):
      for i in range(0, n-j):
         f[i][i+j] = (f[i][i+j-1]-f[i+1][i+j]) / (x[i]-x[i+j])

   # Compute interpolation polynomial
```

```
    def InterpPoly(t):
        ans = f[0][n-1]
        for i in range(n-2, -1, -1):
            ans = ans * (t-x[i]) + f[0][i]
        return ans
    return InterpPoly
```

得到的图像如图 1 所示。从图中可以看出，该插值多项式表现出了非常明显的 Runge 现象——虽然它与给定函数在 11 个点上的取值相同，在 $|x| < 1$ 时也表现出了不错的逼近性质，但当 $|x| > 1$ 且 x 不为整数时，插值多项式与被插函数之间的显著误差是令人难以忍受的，且插值多项式在区间端点上的取值似有发散迹象，这完全偏离了我们用插值多项式逼近被插函数的初衷。

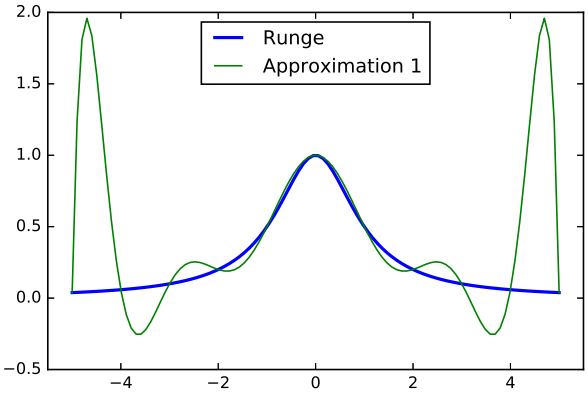

图 1: Runge 函数及其 10 次等距插值多项式的图像

条件二为切比雪夫插值，选取了 $x_i = 5\cos\left(\frac{2i+1}{42}\pi\right)$ $(i = 0, 1, 2, \cdots, 20)$ 共 21 个节点。Lagrange 插值的基本思想是，在多项式空间 $\mathbb{P}_n$ 中选取一组基 $\{l_i(x)\}_{i=0}^n$，使得插值节点在其上的取值 $l_i(x_j) = \delta_{ij}$，进而可写出 Lagrange 插值多项式 $L_n(x) = \sum_{i=0}^n y_i \cdot l_i(x)$。考虑到 $l_i(x)$ 是多项式，有零点 $x_j (j \neq i)$ 且 $l_i(x_i) = 1$，因而必有

$$l_i(x) = \frac{\prod_{j\neq i}(x - x_j)}{\prod_{j\neq i}(x_i - x_j)}.$$

于是完成 Lagrange 插值多项式的构造。具体代码如下：

```
# Lagrange's Interpolation method
# Input: Data points, corresponding values and number of points
# Output: Interpolation polynomial

def LagrangeInterp(x, y, n):
    def InterpPoly(t):
        ans = 0
        for i in range(n):
            tmp = 1
            for j in range(n):
                if j != i:
                    tmp = tmp * (t-x[j]) / (x[i]-x[j])
            ans = ans + y[i] * tmp
        return ans
    return InterpPoly
```

得到的图像如图 2 所示。可以看到，尽管我们常说高次插值多项式是不稳定的，但利用切比雪夫节点进行插值得到的多项式，对被插函数的逼近效果还是相当不错的，极大地消除了 Runge 现象。两种条件插值的差异表明，插值多项式的逼近效果强烈地依赖于插值节点的选取。等距节点的插值多项式序列在很多情况下不是一致收敛的，甚至收敛性也不能保证，而切比雪夫节点做到了在所有可能的插值节点中，逼近误差的最大值达到最小。

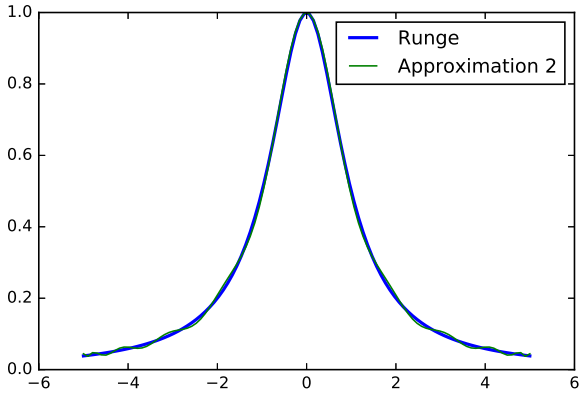

图 2: Runge 函数及其 20 次切比雪夫插值的图像