

《数值分析》上机综合大作业成绩

上机实验报告书

**2021 / 2022 学年 第 2 学期**

**课程名称： 数值分析**

**专业班级： 计算机2002班**

**学 号： 200405212**

**姓 名： 包孝存**

**指导教师： 李威**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 评分项目 | | 分值比例 | 实际得分 |
| 1 | 出勤情况 | | 10% |  |
| 2 | 技术选择合理性 | | 10% |  |
| 3 | 编程质量和答辩 | 设计方案合理、正确 | 10% |  |
| 对设计过程进行阐述 | 10% |  |
| 程序结果演示 | 10% |  |
| 回答问题 | 10% |  |
| 4 | 设计报告 | 文字描述和说明 | 10% |  |
| 图表清楚正确 | 10% |  |
| 成果体现 | 10% |  |
| 撰写规范性 | 10% |  |
| 总得分/成绩等级 | | | |  |
| 备注： | | | | |

**题目1 插值法的应用**

**1．1 题目的主要研究内容及预期达到的目标**

1. **研究内容**

设 f(x)=1/(1+x2) x∈[-5,5]

在[-5,5]内取n+1个等距节点xk=-5+10/n\*k (k=0,1,2,L ,n)

构造当 n=2,4,6,8,10时的插值多项式Ln(x),并在同一张图上画出f(x)和所有Ln(x)的图形。

1. **预期目标**

利用牛顿插值法做出各阶插值多项式图像，与原函数画在同一坐标系钟进行对比，找出阶数增加时，插值函数变化的趋势，以及各阶插值函数的图像特点。

**1．2 题目研究的工作基础或实验条件**

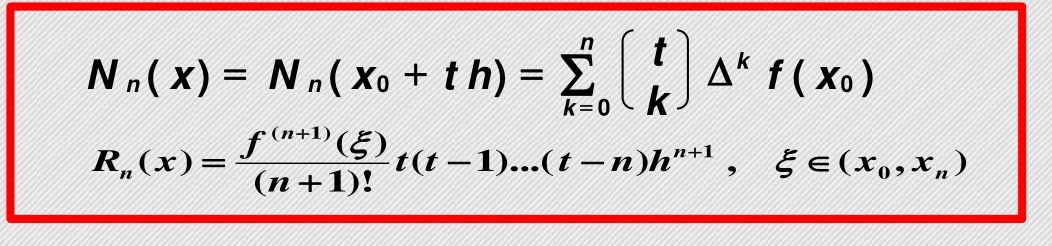
（1）硬件环境 windows 11

（2）软件环境 Visual C++,Eaxy X

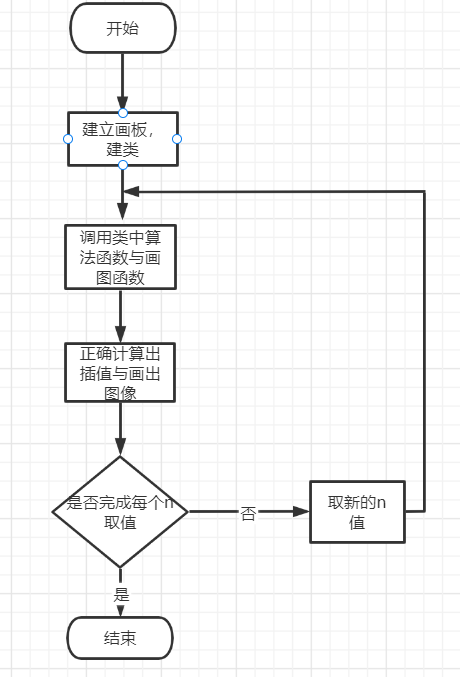
**1．3 设计思想**

选择牛顿等距插值法。使用C++，用一个一维数组存储不同的X值，用一个二维数组存储f(X)以及各阶差分的值，代入公式进行计算，并使用Easy X画出函数图像。分别画出n=2，4，6，8，10时得到函数图像。

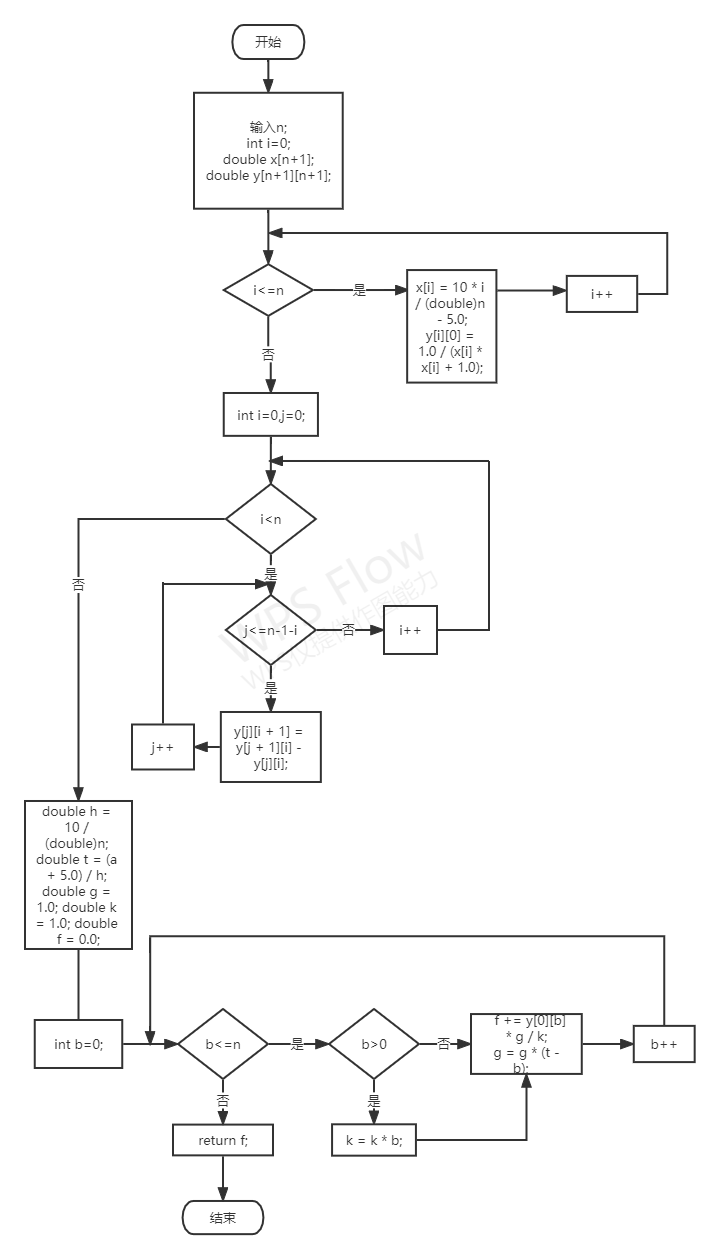
牛顿等距插值法公式：

****

**1．4 流程图**



整体流程图

****

算法流程图

**1．5 主要程序代码**

头文件 picture.h

#pragma once

#include<graphics.h>/\*图形库\*/

#include <conio.h>

#include <iostream>

#include<time.h>

#include<iomanip>

class picture

{

public:

double newton(int n,double a);

void draw(int n, COLORREF pointcolour);

~picture() {};

private:

COLORREF pointcolour;

};

源文件 asignment.cpp

#include"picture.h"

using namespace std;

double picture::newton(int n,double a)

{

double\* x = new double[n + 1];//一维数组x[n+1]

double\*\* y;

y = new double\* [n+1];

for (int i = 0; i < n + 1; i++)

{

y[i] = new double[n + 1];

}//二维数组y[n+1][n+1]

for (int i = 0; i <= n; i++)

{

x[i] = 10 \* i / (double)n - 5.0;//一维数组存储 x值

y[i][0] = 1.0 / (x[i] \* x[i] + 1.0);//二维数组第一列存 标准函数值

}

for (int i = 0; i < n; i++)//计算差分

for (int j = 0; j <= n - 1 - i; j++)

{

y[j][i + 1] = y[j + 1][i] - y[j][i];

}

double h = 10 / (double)n; double t = (a + 5.0) / h; double g = 1.0; double k = 1.0; double f = 0.0;

for (int b = 0; b <=n; b++)//代入等距插值公式

{ if (b >0 )

{

k = k \* b;

}

f += y[0][b] \* g / k;

g = g \* (t - b);

}

return f;

for (int i = 0; i < n + 1; i++)//释放内存

{

delete[] y[i];

delete[]y;

}delete[]x;

}

void picture::draw(int n, COLORREF pointcolour)

{

double i;

for (i = -5.0; i <= 5.0; i = i + 0.001)

{

double x = i \* 100 + 640;

double y = 720 - (newton(n,i) \* 200 + 360) + 50;

putpixel(x, y, pointcolour);

}

}

int main()

{

class picture one;

initgraph(1280, 720, SHOWCONSOLE);//创建窗口及大小

setbkcolor(BLACK);

cleardevice();/\*清屏\*/

setlinecolor(WHITE);

line(640, 0, 640, 720);

line(0, 405, 1280, 405);

setlinecolor(WHITE);

line(540, 408, 540, 400);line(440, 408, 440, 400);line(340,408,340,400); line(240, 408, 240,400);

line(140, 408, 140, 400); line(1140, 408, 1140, 400); line(740, 408, 740, 400); line(840, 408, 840, 400);

line(940, 408, 940, 400); line(1040, 408, 1040, 400); line(640, 210, 648, 210); line(640, 15, 648, 15); line(640, 600, 648, 600);

//建立画板作图

for(double i= -5.0; i <= 5.0; i = i + 0.001)

{

double x = i \* 100 + 640;

double y = 720 - (1 / (i \* i + 1) \* 200 + 360) + 50;

putpixel(x, y, GREEN);

}//原函数

one.draw(2, YELLOW);//n=2

one.draw(4,RED);//n=4

one.draw(6,BLUE);//n=6

one.draw(8,BROWN);//n=8

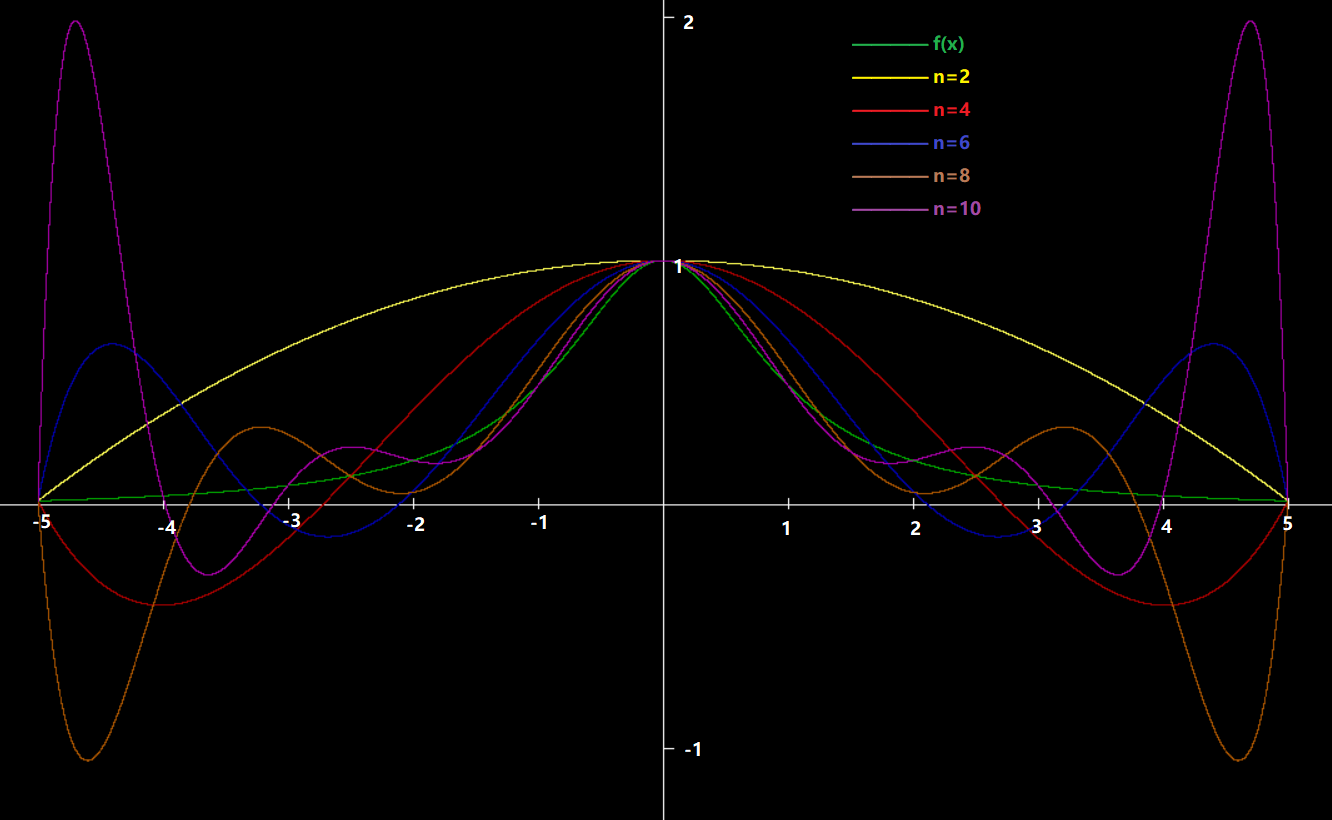
one.draw(10,MAGENTA);//n=10

getchar();

closegraph();

}

**1．6 运行结果及分析**

****

牛顿插值法的优点是计算较简单，尤其是增加节点时，计算只增加一项，这是拉格朗日插值无法比的。  
缺点是仍没有改变拉格朗日的插值曲线在节点处有尖点，不光滑，插值多项式在节点处不可导等缺点。从图中可以看出，n取2，4，6，8，10时，插值的阶数越高在插值区间中部和原函数越接近，但越接近区间边缘插值结果越偏离原函数，在两端产生激烈的震荡，发生函数不收敛的龙格现象，对结果造成较大影响。所以并非n越大结果越准确，应选取适当的插值阶数n。

**1．7 心得体会**

通过本次课程实验，我了解到了牛顿法的本质：用来求方程解，以及其原理与用法，并且对其优缺点和对阶数的选取也有了一定的思考，对所学的知识进行实际应用，把牛顿法与编译语言相结合，独立并且成功的用其解决问题，有了长足的进步。同时感谢老师的支持与指导。