姓名：吴宇迪

学号：10182403

3-1

一阶插值公式

$$y=\frac{x-x_1}{x_0-x_1}y_0+\frac{x-x_0}{x_1-x_0}y_1$$

二阶插值公式

$$y=\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}y_0+\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}y_1+\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}y_2$$

所用代码截图如下

Lagrange_eval.m × wuyudi_3_1.m

D: > desktop > jsff > Lagrange_eval.m

```
function[y0, N] = Lagrange_eval(X, Y, x0)
    %X,Y是已知插值点坐标
    %x0 是插值点
    %y0 是 Lagrange 插值多项式在 x0 处的值
    %N 是 Lagrange 插值函数的权系数
    m = length(X);
    N = zeros(m, 1);
    y0 = 0;

    for i = 1:m
        N(i) = 1;

        for j = 1:m

            if j ~= i;
                N(i) = N(i) * (x0 - X(j)) / (X(i) - X(j));
            end

        end

        y0 = y0 + Y(i) * N(i);
    end
```

x=0.358 时如图所示

wuyudi_3_1.m ×

D: > desktop > jsff > wuyudi_3_1.m

```matlab
X = [0.3, 0.4];
Y = [0.29850, 0.39646];

list = [0.3, 0.4, 0.5, 0.6, 0.7];
x0 = 0.358;
disp('181 wuyudi')
disp('y0=');
disp(Lagrange_eval(X, Y, x0));

%---

X = [0.3, 0.4, 0.5];
Y = [0.29850, 0.39646, 0.49311];

x0 = 0.358;
disp('y0=');
disp(Lagrange_eval(X, Y, x0));

%---

X = [0.3, 0.4,0.5,0.6];
Y = [0.29850, 0.39646, 0.49311,0.58813];

x0 = 0.358;
disp('y0=');
disp(Lagrange_eval(X, Y, x0));
```

MATLAB Command Window

```
  Toolbox Path Cache read in 0.02 seconds.
MATLAB Path initialized in 0.32 seconds.

To get started, type one of these: helpwin, helpdesk, or demo.
For product information, visit www.mathworks.com.

181 wuyudi
y0=
   0.355316800000000

y0=
   0.355476358000000

y0=
   0.355457909360000

>>
```

x=0.462 时

wuyudi_3_1.m ×

D: > desktop > jsff > wuyudi_3_1.m

```matlab
X = [0.3, 0.4];
Y = [0.29850, 0.39646];
x0 = 0.462;
disp('181 wuyudi')
disp('y0=');
disp(Lagrange_eval(X, Y, x0));

%---

X = [0.3, 0.4, 0.5];
Y = [0.29850, 0.39646, 0.49311];
disp('y0=');
disp(Lagrange_eval(X, Y, x0));

%---

X = [0.3, 0.4, 0.5, 0.6];
Y = [0.29850, 0.39646, 0.49311, 0.58813];
disp('y0=');
disp(Lagrange_eval(X, Y, x0));
```

MATLAB Command Window

```
  Toolbox Path Cache read in 0.02 seconds.
MATLAB Path initialized in 0.33 seconds.

To get started, type one of these: helpwin, helpdesk, or demo.
For product information, visit www.mathworks.com.

181 wuyudi
y0=
   0.457195200000000

y0=
   0.456537318000000

y0=
   0.456557673840000

>>
```

x=0.514 时

```
wuyudi_3_1.m
D: > desktop > jsff > wuyudi_3_1.m
X = [0.5, 0.6];
Y = [0.49311, 0.58813];
x0 = 0.514;
disp('181 wuyudi')
disp('y0=');
disp(Lagrange_eval(X, Y, x0));

%---

X = [0.4, 0.5, 0.6];
Y = [0.39646, 0.49311, 0.58813];
disp('y0=');
disp(Lagrange_eval(X, Y, x0));

%---

X = [0.4, 0.5, 0.6, 0.7];
Y = [0.39646, 0.49311, 0.58813, 0.68122];
disp('y0=');
disp(Lagrange_eval(X, Y, x0));
```

```
MATLAB Command Window
 Toolbox Path Cache read in 0.03 seconds.
MATLAB Path initialized in 0.42 seconds.

To get started, type one of these: helpwin, helpdesk, or demo.
For product information, visit www.mathworks.com.

181 wuyudi
y0=
    0.506412800000000

y0=
    0.506510926000000

y0=
    0.506517788800000

>>
```

实验结论：

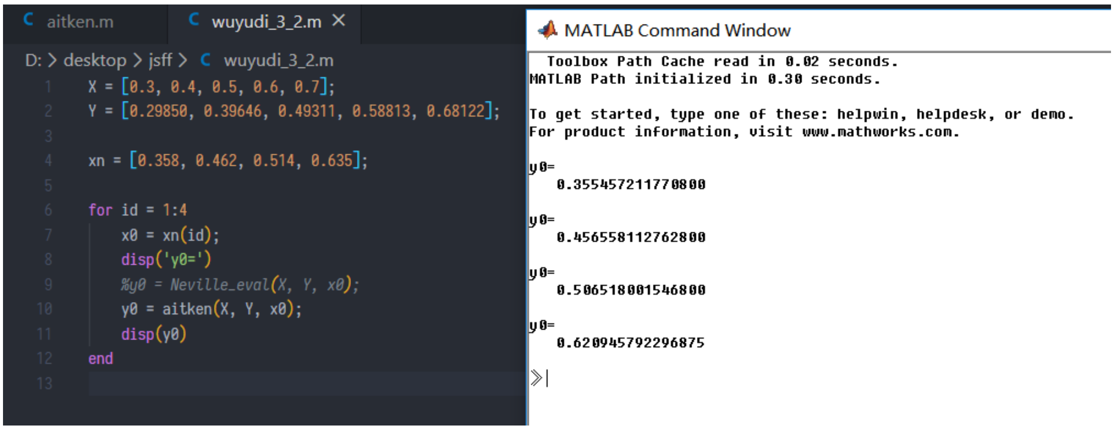
次数越高，插值精度越高

3-2

Aitken 代码如下

```
wuyudi_3_2.m    aitken.m
D: > desktop > jsff > aitken.m > ...
function f1 = aitken(x, f, x1)

    n = length(x);
    D = x1 - x;
    A(:, 1) = f;

    for i = 2:n

        for j = 2:i
            A(i, j) = (D(j - 1) * A(i, j - 1) - D(i) * A(j - 1, j - 1)) / (x(i) - x(j - 1));
        end

    end

    f1 = A(n, n);
```

运行结果如图



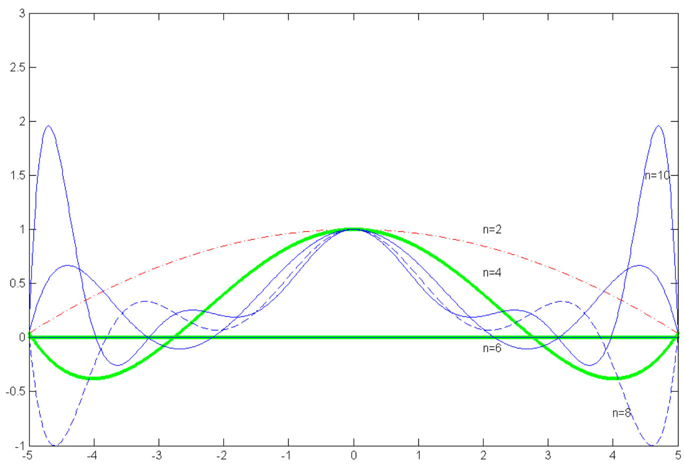
利用 Neville 求解如下

结果非常接近

实验结论：Neville 和 Aitken 插值比拉格朗日插值更精确

3-3

n=2,4,6,8,10 的 $\frac{1}{1+x^2}$ 的 runge 现象如图。



```matlab
% n = 2
x = linspace(-5, 5, 3);
y = 1 ./ (1 + x.^2);
% 插值数据
x0 = linspace(-5, 5, 1000);
y0 = zeros(1000);
% 插值向量

for index = 1:1000
    y0(index) = Lagrange_eval(x, y, x0(index));
end

plot(x0, y0,  '-.r')
ylim([-1,3])
text(2,1,'n=2')
hold on

%--------------------------------------------
```

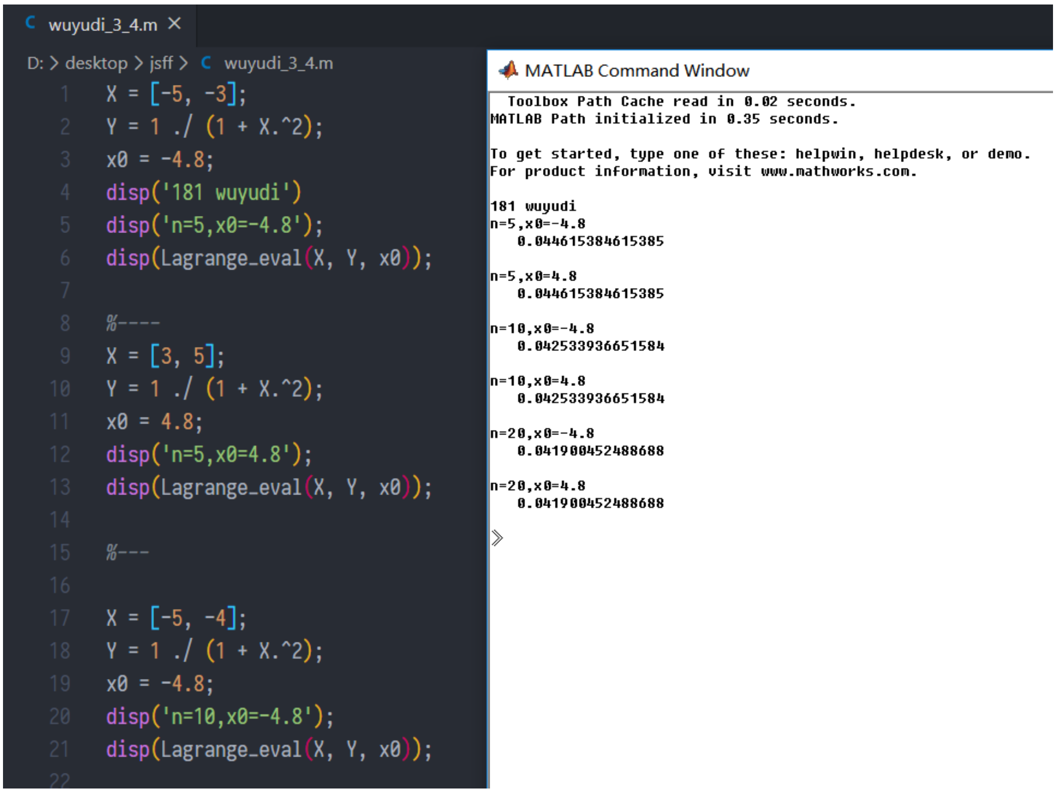
```matlab
% n =4
x = linspace(-5, 5, 5);
y = 1 ./ (1 + x.^2);
% 插值数据
x0 = linspace(-5, 5, 1000);
y0 = zeros(1000);
% 插值向量

for index = 1:1000
    y0(index) = Lagrange_eval(x, y, x0(index));
end

plot(x0, y0,  '.g')
ylim([-1, 3])
text(2, 0.6, 'n=4')
hold on

%--------------------------------------------
```

```matlab
%--------------------------------------------

% n = 6
x = linspace(-5, 5, 7);
y = 1 ./ (1 + x.^2);
% 插值数据
x0 = linspace(-5, 5, 1000);
y0 = zeros(1000);
% 插值向量

for index = 1:1000
    y0(index) = Lagrange_eval(x, y, x0(index));
end

plot(x0, y0,  'b')
ylim([-1, 3])
text(2,-0.1, 'n=6')
hold on

%--------------------------------------------
```

```matlab
% n = 8
x = linspace(-5, 5, 9);
y = 1 ./ (1 + x.^2);
% 插值数据
x0 = linspace(-5, 5, 1000);
y0 = zeros(1000);
% 插值向量

for index = 1:1000
    y0(index) = Lagrange_eval(x, y, x0(index));
end

plot(x0, y0,  '-c')
ylim([-1, 3])
text(4, -0.7,  'n=8')
hold on

%--------------------------------------------
```

```matlab
% n = 10
x = linspace(-5, 5, 11);
y = 1 ./ (1 + x.^2);
% 插值数据
x0 = linspace(-5, 5, 1000);
y0 = zeros(1000);
% 插值向量

for index = 1:1000
    y0(index) = Lagrange_eval(x, y, x0(index));
end

plot(x0, y0,  'k')
ylim([-1, 3])
text(4.5, 1.5,  'n=10')
hold off
```

分段线性插值如下



实验结论：

Lagrange 插值，当次数过高会有 Runge 现象。

分段线性插值，等分数越多，精度越高

设计思想

（1）Lagrange 插值： Lagrange 具有累加的嵌套结构，容易编制其计算程序。事实上，在逻辑上表现为二重循环，内循环（j 循环）累乘求得系数，然后再通过外循环（i 循环）累加得出插值结果 y。

（2）分段线性插值：分段插值是将被插值函数逐步多项式化。分段插值的处理过程分两步，将区间分成几个子段，并在每个子段上构造插值多项式装配在一起，作为整个区间的插值函数。在分化的每个节点给出数据，连接相邻节点得一折线，该折线函数可以视作插值问题的解。

（3）Neville 插值： Neville 插值的基本思想和 Aitken 插值一样，不同的是 Neville 插值每次选取的两个插值节点都是上一步相邻节点插值后得到的，而不是新的插值节点，这样得到的插值函数和原函数更加接近。

Atiken 逐步插值： Aitken 插值是对三步插值转化为两步插值的重复，先将前两个插值点插值生成新的数据，然后与第三个插值点进行新的两点插值，不断重复这个插值过程，每一步增加一个新的节点，直到遍历所有节点为止，最终获得与原函数更加接近的插值函数。

（4）Hermite 插值： Hermite 插值是 Lagrange 插值的综合与推广，， 为了保证插值函数能更好地密合原来的函数，要求 “过点”，即两者在节点上有相同的函数值，而且要求 “相切”，即在节点上还具有相同的导数值。

四：实验体会

Lagrange 插值在高次插值时同原函数插值偏差大，拉格朗日插值模型简单，结构紧凑，是经典的插值法。但是由于拉格朗日的插值多项式和每个节点都有关，当改变节点个数时，需要重新计算。且当增大插值阶数时容易出现龙格现象。分段线性插值是将整个区间分成许多小段, 运用低次插值, 从而提高精度。分段线性插值算法简单, 计算量小, 但精度不高。Neville 插值的基本思想和 Aitken 插值一样，不同的是 Neville 插值每次选取的两个插值节点都是上一步相邻节点插值后得到的, 而不是新的插值节点, 这样得到的插值函数和原函数更加接近。Aitken 插值是对三步插值转化为两步插值的重复，先将前两个插值点插值生成新的数据，然后与第三个插值点进行新的两点插值，不断重复这个插值过程，每一步增加一个新的节点，直到遍历所有节点为止，最终获得与原函数更加接近的插值函数。Hermite 插值是 Lagrange 插值的综合与推广，，为了保证插值函数能更好地密合原来的函数，要求“过点”，即两者在节点上有相同的函数值，而且要求“相切”，即在节点上还具有相同的导数值。这就保证了有较高的精度。