计算方法实验报告

姓名：段佳慧

学号：200110807

院系：计算机科学与技术学院

专业：计算机科学与技术

班级：8 班

实验报告一

第一部分：问题分析 *（描述并总结出实验题目）*

摘要：利用拉格朗日插值多项式 Pn(x)求 f(x)的近似值。

实验目的：利用拉格朗日插值多项式 Pn(x)求 f(x)的近似值。

实验意义：根据给定平面上的 n+1 个不同的数据点（xk, f(xk)），求 f(x)在插值点 x 的近似值 Pn(x)。
通过离散的点的坐标，利用插值函数逼近和模拟这个未知函数，从而得出其函数值。
更好地利用 MATLAB 更清晰地理解拉格朗日插值的含义

第二部分：数学原理

已知一个函数的若干点坐标（x_k, y_k），k=0, 1, . . . , n, 若想通过这些已知信息来猜测这个函数，则可以通过构造一个过这 n+1 个点、次数不超过 n 的多项式 Pn(xk)来实现, 其中 Pn(xk)满足

$$P_n(x)=y_k,\ k=0, 1, \ldots, n.$$

可以证明满足以上条件的多项式是存在且唯一的。
运用这种思想的其中一个方法是拉格朗日插值法，应用拉格朗日插值公式所得到的拉格朗日插值多项式：

$$L(x) := \sum_{j=0}^{k} y_j \ell_j(x)$$

其中每个 $l_j(x)$为拉格朗日基本多项式（或称插值基函数），其表达式为：

$$\ell_j(x) := \prod_{i=0,\, i\neq j}^{k} \frac{x - x_i}{x_j - x_i} = \frac{(x - x_0)}{(x_j - x_0)} \cdots \frac{(x - x_{j-1})}{(x_j - x_{j-1})} \frac{(x - x_{j+1})}{(x_j - x_{j+1})} \cdots \frac{(x - x_k)}{(x_j - x_k)}$$

拉格朗日基本多项式的特点是在 xj 上取值为 1，在其它的点 i≠j 上取值为 0。

第三部分：程序设计流程

输　　入：$n+1$个数据点$(x_k, f(x_k))$，$k=0,1,\cdots,n$；插值点x

输　　出：$f(x)$在插值点x的近似值$P_n(x)$

程序流程：

1 置$y=0.0$；$k=0$

2 当$k \le n$时，做2.1—2.4

2.1 置$l=1.0$；

2.2 对$j=0,1,\cdots,k-1,k+1,\cdots,n$，置$l=l\cdot(x-x_j)/(x_k-x_j)$

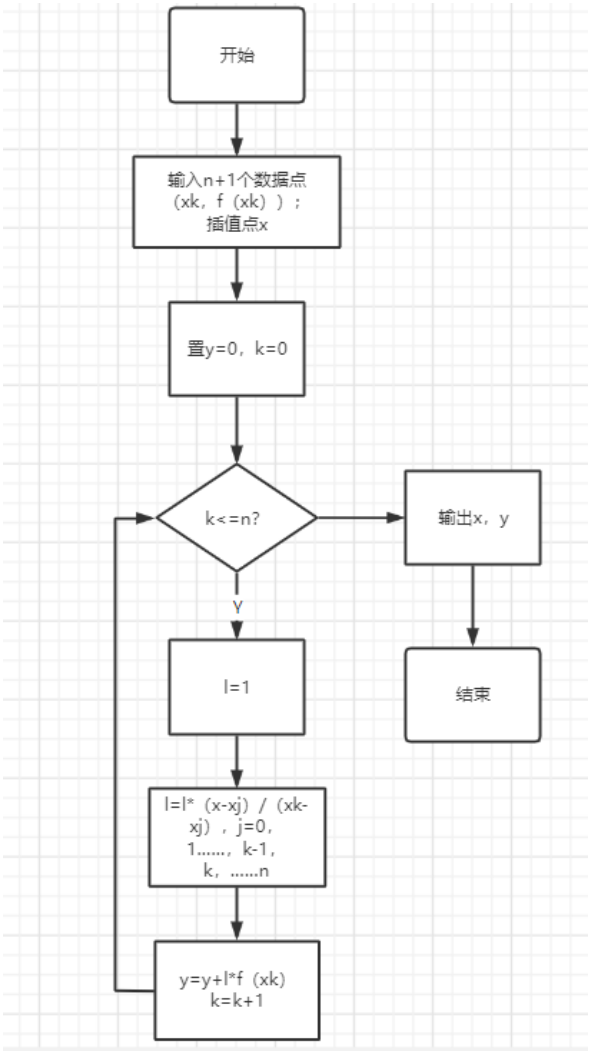
2.3 置$y=y+l\cdot f(x_k)$

2.4 置$k=k+1$

3 输出x, y

4 停机

流程图如下：



代码如下：

问题 1(1)

```
syms y k x n x1 xk h;
x1=input('请输入 x 的值：');
n=input('请输入 n 的值：');
y=0.0;
f=1/(1+x^2);
h=10.0/n;
for k=0:n
    l=1.0;
    xk=-5.0+k*h;
    for j=0:k-1
        if k-1==-1
            break;
        end
        xj=-5.0+j*h;
        l=l*(x1-xj)/(xk-xj);
    end
    for j=k+1:n
        if k+1==n+1
            break;
        end
        xj=-5.0+j*h;
        l=l*(x1-xj)/(xk-xj);
    end
    yk=double(subs(f,'x',xk));
    y=y+l*yk;
end
fprintf('x=%.10f',x1);
fprintf('y=%.10f',y);
```

问题 1(2)

```
syms y k x n x1 xk h;
x1=input('请输入 x 的值：');
n=input('请输入 n 的值：');
y=0.0;
f=exp(x);
h=2.0/n;
for k=0:n
    l=1.0;
    xk=-1.0+k*h;
    for j=0:k-1
        if k-1==-1
            break;
        end
        xj=-1.0+j*h;
        l=l*(x1-xj)/(xk-xj);
    end
    for j=k+1:n
        if k+1==n+1
            break;
        end
        xj=-1.0+j*h;
        l=l*(x1-xj)/(xk-xj);
    end
    yk=double(subs(f,'x',xk));
    y=y+l*yk;
end
fprintf('x=%.10f',x1);
fprintf('y=%.10f',y);
```

问题 2(1)

```
syms y k x n x1 xk h;
x1=input('请输入 x 的值：');
n=input('请输入 n 的值：');
y=0.0;
f=1/(1+x^2);
h=2.0/n;
for k=0:n
    l=1.0;
    xk=-1.0+k*h;
    for j=0:k-1
        if k-1==-1
            break;
        end
        xj=-1.0+j*h;
        l=l*(x1-xj)/(xk-xj);
    end
    for j=k+1:n
        if k+1==n+1
            break;
        end
        xj=-1.0+j*h;
        l=l*(x1-xj)/(xk-xj);
    end
    yk=double(subs(f,'x',xk));
    y=y+l*yk;
end
fprintf('x=%.10f',x1);
fprintf('y=%.10f',y);
```

问题 2(2)

```
syms y k x n x1 xk h;
x1=input('请输入 x 的值：');
n=input('请输入 n 的值：');
y=0.0;
f=1/(1+x^2);
h=2.0/n;
for k=0:n
    l=1.0;
    xk=-1.0+k*h;
    for j=0:k-1
        if k-1==-1
            break;
        end
        xj=-1.0+j*h;
        l=l*(x1-xj)/(xk-xj);
    end
    for j=k+1:n
        if k+1==n+1
            break;
        end
        xj=-1.0+j*h;
        l=l*(x1-xj)/(xk-xj);
    end
    yk=double(subs(f,'x',xk));
    y=y+l*yk;
end
fprintf('x=%.10f',x1);
fprintf('y=%.10f',y);
```

问题 4 代码如下：

```
syms y x x0 x1 x2 x3;
x3=input('请输入 x 的值: ');
x0=input('请输入 x0 的值: ');
x1=input('请输入 x1 的值: ');
x2=input('请输入 x2 的值: ');
y=0.0;
f=x^0.5;
l=1.0;
l=l*(x3-x1)/(x0-x1);
l=l*(x3-x2)/(x0-x2);
yk=double(subs(f,'x',x0));
y=y+l*yk;
l=1.0;
l=l*(x3-x0)/(x1-x0);
l=l*(x3-x2)/(x1-x2);
yk=double(subs(f,'x',x1));
y=y+l*yk;
l=1.0;
l=l*(x3-x0)/(x2-x0);
l=l*(x3-x1)/(x2-x1);
yk=double(subs(f,'x',x2));
y=y+l*yk;
fprintf('x=%.10f',x3);
fprintf('y=%.10f',y);
```

第四部分：实验结果、结论与讨论

问题 1（1）$f(x)=1/(x^2+1)$ ，$x\in[-5,5]$

n=5 时：

```
>> lagrange
请输入x的值：0.75
请输入n的值：5
x=0.7500000000y=0.52897385821agrange
请输入x的值：1.75
请输入n的值：5
x=1.7500000000y=0.37332481971agrange
请输入x的值：2.75
请输入n的值：5
x=2.7500000000y=0.1537334736>> lagrange
请输入x的值：3.75
请输入n的值：5
x=3.7500000000y=-0.0259540264>> lagrange
请输入x的值：4.75
请输入n的值：5
```

n=10 时：

```
>> lagrange
请输入x的值：0.75
请输入n的值：10
x=0.7500000000y=0.6789895773>> lagrange
请输入x的值：1.75
请输入n的值：10
x=1.7500000000y=0.1905804668>> lagrange
请输入x的值：2.75
请输入n的值：10
x=2.7500000000y=0.2155918789>> lagrange
请输入x的值：3.75
请输入n的值：10
x=3.7500000000y=-0.2314617499>> lagrange
请输入x的值：4.75
请输入n的值：10
x=4.7500000000y=1.9236311497>> lagrange
```

n=20 时：

```
请输入x的值：0.75
请输入n的值：20
x=0.7500000000y=0.6367553359>> lagrange
请输入x的值：1.75
请输入n的值：20
x=1.7500000000y=0.2384459337>> lagrange
请输入x的值：2.75
请输入n的值：20
x=2.7500000000y=0.0806599934>> lagrange
请输入x的值：3.75
请输入n的值：20
x=3.7500000000y=-0.4470519607>> lagrange
请输入x的值：4.75
请输入n的值：20
x=4.7500000000y=-39.9524490330>>
```

问题 1（2）$f(x)=e^x$ ， $x\in[-1,1]$

n=5 时：

```
请输入x的值：-0.95
请输入n的值：5
x=-0.9500000000y=0.3867981589>> lagrange_2
请输入x的值：-0.05
请输入n的值：5
x=-0.0500000000y=0.9512483334>> lagrange_2
请输入x的值：0.05
请输入n的值：5
x=0.0500000000y=1.0512902758>> lagrange_2
请输入x的值：0.95
请输入n的值：5
x=0.9500000000y=2.5857845510>>
```

n=10 时：

```
>> lagrange_2
请输入x的值：-0.95
请输入n的值：10
x=-0.9500000000y=0.3867410233>> lagrange_2
请输入x的值：-0.05
请输入n的值：10
x=-0.0500000000y=0.9512294245>> lagrange_2
请输入x的值：0.05
请输入n的值：10
x=0.0500000000y=1.0512710964>> lagrange_2
请输入x的值：0.95
请输入n的值：10
x=0.9500000000y=2.5857096595>>
```

n=20 时：

```
>> lagrange_2
请输入x的值：-0.95
请输入n的值：20
x=-0.9500000000y=0.3867410235>> lagrange_2
请输入x的值：-0.05
请输入n的值：20
x=-0.0500000000y=0.9512294245>> lagrange_2
请输入x的值：0.05
请输入n的值：20
x=0.0500000000y=1.0512710964>> lagrange_2
请输入x的值：0.95
请输入n的值：20
x=0.9500000000y=2.5857096593>>
```

问题 2（1）$f(x)=1/(x^2+1)$ ，$x\in[-1,1]$

n=5 时：

```
>> lagrange_3
请输入x的值：-0.95
请输入n的值：5
x=-0.9500000000y=0.5171472886>> lagrange_3
请输入x的值：-0.05
请输入n的值：5
x=-0.0500000000y=0.9927906710>> lagrange_3
请输入x的值：0.05
请输入n的值：5
x=0.0500000000y=0.9927906710>> lagrange_3
请输入x的值：0.95
请输入n的值：5
x=0.9500000000y=0.5171472886>>
```

n=10 时：

```
>> lagrange_3
请输入x的值：-0.95
请输入n的值：10
x=-0.9500000000y=0.5264079831>> lagrange_3
请输入x的值：-0.05
请输入n的值：10
x=-0.0500000000y=0.9975068566>> lagrange_3
请输入x的值：0.05
请输入n的值：10
x=0.0500000000y=0.9975068566>> lagrange_3
请输入x的值：0.95
请输入n的值：10
x=0.9500000000y=0.5264079831>>
```

n=20 时：

```
>> lagrange_3
请输入x的值：-0.95
请输入n的值：20
x=-0.9500000000y=0.5256203657>> lagrange_3
请输入x的值：-0.05
请输入n的值：20
x=-0.0500000000y=0.9975062344>> lagrange_3
请输入x的值：0.05
请输入n的值：20
x=0.0500000000y=0.9975062344>> lagrange_3
请输入x的值：0.95
请输入n的值：20
x=0.9500000000y=0.5256203657>>
```

问题 2（2）$f(x)=e^x$ ， $x\in[-5,5]$

n=5 时：

```
>> lagrange_4
请输入x的值：-4.75
请输入n的值：5
x=-4.7500000000y=1.1470347315>> lagrange_4
请输入x的值：-0.25
请输入n的值：5
x=-0.2500000000y=1.3021524636>> lagrange_4
请输入x的值：0.25
请输入n的值：5
x=0.2500000000y=1.8412104110>> lagrange_4
请输入x的值：4.75
请输入n的值：5
x=4.7500000000y=119.6210070561>>
```

n=10 时：

```
>> lagrange_4
请输入x的值：-4.75
请输入n的值：10
x=-4.7500000000y=-0.0019565505>> lagrange_4
请输入x的值：-0.25
请输入n的值：10
x=-0.2500000000y=0.7786863439>> lagrange_4
请输入x的值：0.25
请输入n的值：10
x=0.2500000000y=1.2841444870>> lagrange_4
请输入x的值：4.75
请输入n的值：10
x=4.7500000000y=115.6073600631>>
```

n=20 时：

```
>> lagrange_4
请输入x的值：-4.75
请输入n的值：20
x=-4.7500000000y=0.0086516938>> lagrange_4
请输入x的值：-0.25
请输入n的值：20
x=-0.2500000000y=0.7788007831>> lagrange_4
请输入x的值：0.25
请输入n的值：20
x=0.2500000000y=1.2840254167>> lagrange_4
请输入x的值：4.75
请输入n的值：20
x=4.7500000000y=115.5842845294>>
```

问题 4（1）f(x)=x^0.5， x0=1,x1=4,x2=9

```
>> lagrange_5
请输入x的值：5
请输入x0的值：1
请输入x1的值：4
请输入x2的值：9
x=5.0000000000y=2.2666666667>> lagrange_5
请输入x的值：50
请输入x0的值：1
请输入x1的值：4
请输入x2的值：9
x=50.0000000000y=-20.2333333333>> lagrange_5
请输入x的值：115
请输入x0的值：1
请输入x1的值：4
请输入x2的值：9
x=115.0000000000y=-171.9000000000>> lagrange_5
请输入x的值：185
请输入x0的值：1
请输入x1的值：4
请输入x2的值：9
x=185.0000000000y=-492.7333333333>>
```

问题 4（2）f(x)=x^0.5， x0=36,x1=49,x2=64

```
请输入x的值：5
请输入x0的值：36
请输入x1的值：49
请输入x2的值：64
x=5.0000000000y=3.1157509158>>
>> lagrange_5
请输入x的值：50
请输入x0的值：36
请输入x1的值：49
请输入x2的值：64
x=50.0000000000y=7.0717948718>>
>> lagrange_5
请输入x的值：115
请输入x0的值：36
请输入x1的值：49
请输入x2的值：64
x=115.0000000000y=10.1670329670>>
>> lagrange_5
请输入x的值：185
请输入x0的值：36
请输入x1的值：49
请输入x2的值：64
x=185.0000000000y=10.0388278388>>
```

问题 4（3）f(x)=x^0.5， x0=100, x1=121, x2=144

```
>> lagrange_5
请输入x的值: 5
请输入x0的值: 100
请输入x1的值: 121
请输入x2的值: 144
x=5.0000000000y=4.4391116130>> lagrange_5
请输入x的值: 50
请输入x0的值: 100
请输入x1的值: 121
请输入x2的值: 144
x=50.0000000000y=7.2849614154>> lagrange_5
请输入x的值: 115
请输入x0的值: 100
请输入x1的值: 121
请输入x2的值: 144
x=115.0000000000y=10.7227555054>> lagrange_5
请输入x的值: 185
请输入x0的值: 100
请输入x1的值: 121
请输入x2的值: 144
x=185.0000000000y=13.5356672313>>
```

问题 4（4）f(x)=x^0.5， x0=169, x1=196, x2=225

```
>> lagrange_5
请输入x的值: 5
请输入x0的值: 169
请输入x1的值: 196
请输入x2的值: 225
x=5.0000000000y=5.4971720489>> lagrange_5
请输入x的值: 50
请输入x0的值: 169
请输入x1的值: 196
请输入x2的值: 225
x=50.0000000000y=7.8001277139>> lagrange_5
请输入x的值: 115
请输入x0的值: 169
请输入x1的值: 196
请输入x2的值: 225
x=115.0000000000y=10.8004926108>> lagrange_5
请输入x的值: 185
请输入x0的值: 169
请输入x1的值: 196
请输入x2的值: 225
x=185.0000000000y=13.6006203248>>
```

问题：

1. 拉格朗日插值多项式的次数 n 并不是越大越好，对于有些函数，只当 x 在某一区间内才会收敛，在该区间以外，随着 n 增大，插值逼近的效果也变差，即 runge 现象。

2. 插值区间越小，所得结果也越精确，逼近效果比在大区间上用高次插值所得效果更好。

4. 内插比外推更可靠。

思考题：

1. 实验 1 中存在在取值区间的边缘，预估函数值和真实函数值相差过大的情况，即 runge 现象；解决办法是分段插值，将区间分为多段，并且在每个小区间上分别作低次插值（如线性插值）。

2. 插值区间越小，函数的变化趋势就相对确定，且对于连续函数，变化也就更加可以预测，所以也就越可靠。

4. 内插是要求的值在已知的序列中，趋势比较好预测；外推是要求的值在已知的序列外，趋势较难预测。

实验报告二

第一部分：问题分析 *（描述并总结出实验题目）*

摘要：高斯消元法第 k 步到第 k+1 步的消元过程，必须满足条件 $a_{kk}^{(k-1)} \neq 0$ 。而这个元 $a_{kk}^{(k-1)}$ 即被称为第 k 步的主元素。显然，高斯消去法是按照方程排列的自然顺序产生主元的，这样，一旦出现 $a_{kk}^{(k-1)} = 0$，计算就归于失败，而且即使 $a_{kk}^{(k-1)} \neq 0$ 但若其绝对值很小，也将会应用它作为除数，引起其他元素的数量级及舍入误差急剧增大，导致最终计算结果不可靠。为了避免在高斯消元法引用中可能出现的这类问题，就发展形成了列主元，全主元等多种消元方法，这些方法的基本点在于对高斯消去法的过程作某些技术性修改，全面或者局部地选择绝对值最大的元素作为主元素，从而构成了相应的主元素消元法，列主元素消元法以处理简单相对计算量小的特点，在各类主元消去法得到最为广泛的应用。

目的：利用 Gauss 列主元消去法求解线性方程组 Ax=b 。
意义：理解高斯列主消元法的基本原理，会使用高斯消元法解决问题。

第二部分：数学原理

高斯消去法中第 k-1 步消元得到的结果可由分块矩阵记为

$$\left[\boldsymbol{A}^{(k-1)}, \boldsymbol{b}^{(k-1)}\right]=\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \vdots & \boldsymbol{b}_1 \\ \boldsymbol{0} & \boldsymbol{A}_{kk} & \vdots & \boldsymbol{b}_2 \end{bmatrix}.$$

该式中，b1、b2 为对应于右端常数列的两个子块，而

$$\boldsymbol{A}_{11}=\begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & \cdots & a_{1,k-1}^{(0)} \\ & a_{22}^{(1)} & \cdots & a_{2,k-1}^{(1)} \\ & & & \vdots \\ & & & a_{k-1,k-1}^{(k-2)} \end{bmatrix},$$

$$\boldsymbol{A}_{kk}=\begin{bmatrix} a_{kk}^{(k-1)} & a_{k,k+1}^{(k-1)} & \cdots & a_{k,n}^{(k-1)} \\ a_{k+1,k}^{(k-1)} & a_{k+1,k+1}^{(k-1)} & \cdots & a_{k+1,n}^{(k-1)} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n,k}^{(k-1)} & a_{n,k+1}^{(k-1)} & \cdots & a_{n,n}^{(k-1)} \end{bmatrix}.$$

与高斯消去法不同的是，列主元消去法在第 k 步消元之前，先在它的第 k 列主对角元 $a_{k,k}^{(k-1)}$及其下方的所有元素中(亦即 A_{kk}的第一列元素中)选出绝对值最大的元素 $a_{ik,k}^{(k-1)}$作为这一列的主元，即$|a_{ik,k}^{(k-1)}|=\max(k<=i<=n)|a_{ik}^{(k-1)}|$（k=1,2,...,n-1）。对最上面那个式子作初等行变换$[i_k]\leftrightarrow[k]$（$i_k>=k$），这样就把 $a_{ik,k}^{(k-1)}$换到主对角元位置上。经过选列主元与行交换之后，再如高斯消去法一样作行的消元变换。上述过程从第一步消元开始执行，即 k=1，2，…，n-1，这就构成了列主元消去法。综上可见，列主元消去法仅需矩阵行的交换，因此不发生未知数排列次序的调换。而全主元消去法则要花费相当多的时间选取主元，
且还需记录系数矩阵列交换信息。

第三部分：程序设计流程

流程如下：

1 对 $k=1,2,\cdots,n-1$，做 1.1—1.3，消元过程

1.1 寻找最小的正整数 p， $k\le p\le n$ 和 $|a_{pk}|=\max\limits_{k\le j\le n}|a_{jk}|$。如果 $a_{pk}=0$，输出奇异标志，停机；

1.2 如果 $p\neq k$，那么交换 p,k 两行；

1.3 对 $i=k+1,\cdots,n$，记 $m_{ik}=a_{ik}/a_{kk}$，计算

$$\begin{cases} a_{ij}=a_{ij}-a_{kj}m_{ik} \\ i=k+1,\cdots,n \\ j=k+1,\cdots,n \\ b_i=b_i-b_k m_{ik} \\ i=k+1,\cdots,n \end{cases}$$

2. 如果 $a_{nn}=0$ 输出奇异标志，停机；

3. 置 $x_n=b_n/a_{nn}$，回代过程

4. 对 $k=n-1,\cdots,2,1$，置 $x_k=(b_k-\sum_{j=k+1}^{n}a_{kj}x_j)/a_{kk}$

代码如下：

```
A=input('输入系数矩阵 A: ');
b=input('输入 b 向量(行向量): ');
n=length(b);
x=zeros(1,n);
c=zeros(1,n);
d=0;
for k=1:n-1
    max=abs(A(k,k));
    p=k;
    for j=k+1:n
        if max<abs(A(j,k))
            max=abs(A(j,k));
            p=j;
        end
    end
    if max == 0
        disp('输出失败');
        return;
    end
    if(p~=k)
        for j=k:n
            c(j)=A(k,j);
            A(k,j)=A(p,j);
            A(p,j)=c(j);
        end
        d=b(k);
        b(k)=b(p);
        b(p)=d;
    end
    for i=k+1:n
        m=A(i,k)/A(k,k);
        for j=k+1:n
            A(i,j)=A(i,j)-A(k,j)*m;
        end
        b(i)=b(i)-b(k)*m;
        A(i,k)=0;
    end
end
if A(n,n) == 0
    disp('输出失败');
    return;
end
```

```
x(n)=b(n)/A(n,n);
for k=n-1:-1:1
    sum=0;
    for j=k+1:n
        sum=sum+A(k,j)*x(j);
    end
    x(k)=(b(k)-sum)/A(k,k);
end
disp(x);
```

第四部分：实验结果、结论与讨论

问题 1

```
>> gauss
输入系数矩阵A：[0.4096 0.1234 0.3678 0.2943;0.2246 0.3872 0.4015 0.1129;0.3645 0.1920 0.3781 0.0643;0.1784 0.4002 0.2786 0.3927]
输入b向量(行向量)：[1.1951 1.1262 0.9989 1.2499]
    1.0000    1.0000    1.0000    1.0000

>> gauss
输入系数矩阵A：[136.01 90.860 0 0;90.860 98.810 -67.590 0;0 -67.590 132.01 46.260;0 0 46.260 177.17]
输入b向量(行向量)：[226.87 122.08 110.68 223.43]
    1.0000    1.0000    1.0000    1.0000

>> gauss
输入系数矩阵A：[1 1/2 1/3 1/4;1/2 1/3 1/4 1/5;1/3 1/4 1/5 1/6;1/4 1/5 1/6 1/7]
输入b向量(行向量)：[25/12 77/60 57/60 319/420]
    1.0000    1.0000    1.0000    1.0000

>> gauss
输入系数矩阵A：[10 7 8 7;7 5 6 5;8 6 10 9;7 5 9 10]
输入b向量(行向量)：[32 23 33 31]
     1     1     1     1
```

问题 2

```
>> gauss
输入系数矩阵A：[197 305 -206 -804;46.8 71.3 -47.4 52.0;88.6 76.4 -10.8 802;1.45 5.90 6.13 36.5]
输入b向量(行向量)：[136 11.7 25.1 6.60]
    0.9537    0.3210    1.0787   -0.0901

>> gauss
输入系数矩阵A：[0.5398 0.7161 -0.5554 -0.2982;0.5257 0.6924 0.3565 -0.6255;0.6465 -0.8187 -0.1872 0.1291;0.5814 0.9400 -0.7779 -0.4042]
输入b向量(行向量)：[0.2058 -0.0503 0.1070 0.1859]
    0.5162    0.4152    0.1100    1.0365

>> gauss
输入系数矩阵A：[10 1 2;1 10 2;1 1 5]
输入b向量(行向量)：[13 13 7]
    1.0000    1.0000    1.0000

>> gauss
输入系数矩阵A：[4 -2 -4;-2 17 10;-4 10 9]
输入b向量(行向量)：[-2 25 15]
     1     1     1
```

无思考题

总结：这个实验的代码比较难想，主要是对于求解的过程，有单独两个全零矩阵进行迭代计算，一开始没有想到，后来是上网查了其他代码才想到的

实验报告三

第一部分：问题分析 *（描述并总结出实验题目）*

目的：利用牛顿迭代法求 f(x)=0 的根

意义：牛顿迭代法是通过函数上面的过某一点（自变量 x_0 为初值）的切线与 x 轴的交点 x_1 为新的自变量值，取这个自变量对应的函数值所对应的点再次重复上述取切线与 x 轴的交点 x_2 进行迭代，当误差范围小于一定的标准时可以近似切线与 x 轴的交点值 x_n(n 为迭代的总次数)就等于 f(x)=0 的根值 a。此方法当选择的初值 x_0 接近 f(x)=0 的根值 x^*时可以保证迭代收敛，是一种有效的非线性方程数值解法。此次实验有利于我们熟悉 MATLAB 语言编写程序，同样能加深对牛顿迭代法的理解，可以起到复习的作用。

第二部分：数学原理

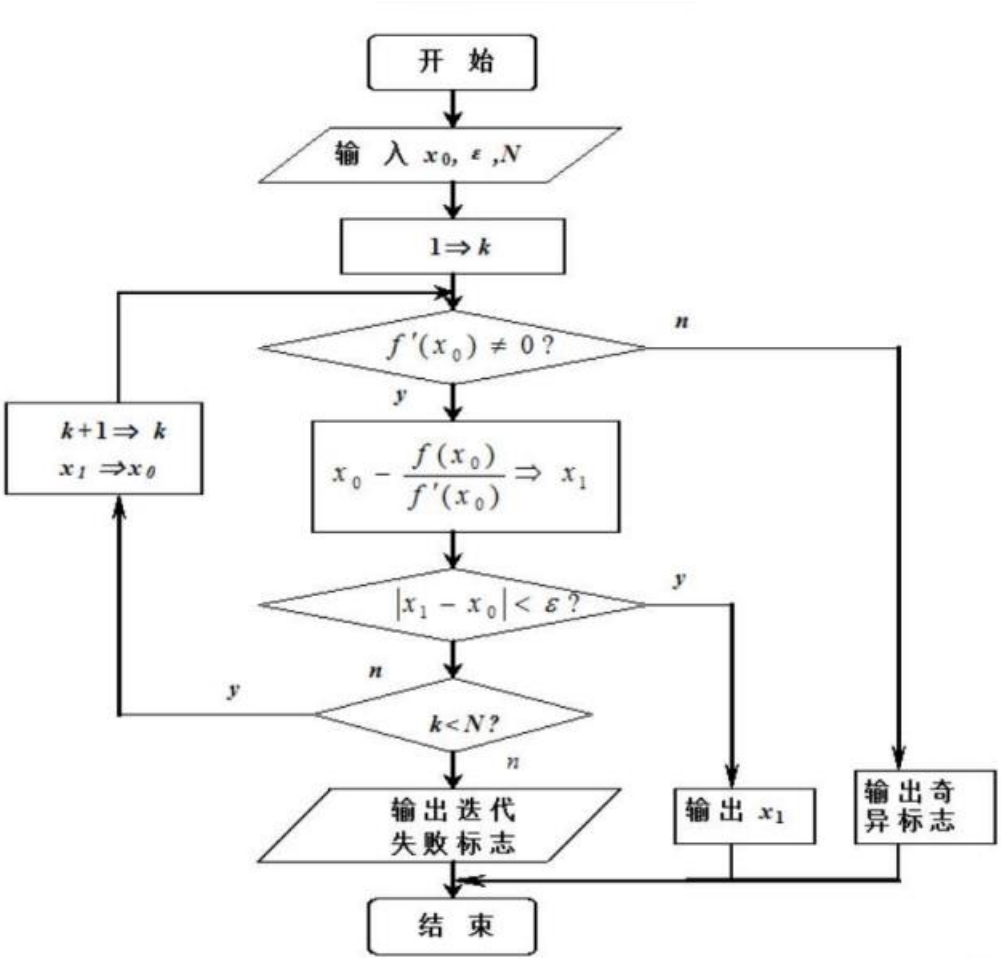
设 x^*是 f(x)=0 的根，选取 x_0 作为 x^*的初始近似值，过点$(x_0,f(x_0))$ 做曲线 y=f(x)的切线 L，L:$y=f(x_0)+f'(x_0)(x-x_0)$，则 L 与 x 轴交点的横坐标 $x_1=x_0-f(x_0)/f'(x_0)$，称 x_1 为 x^*的一次近似值。过点$(x_1,f(x_1))$做曲线 y=f(x)的切线，并求该切线与 x 轴交点的横坐标 $x_2=x_1-f(x_1)/f'(x_1)$，称 x_2 为 x^*的二次近似值。重复以上过程，得 x^*的近似值序列，其中，$x_{n+1}=x_n-f(x_n)/f'(x_n)$称为 x^*的 n+1 次近似值，上式称为牛顿迭代公式。

用牛顿迭代法解非线性方程，是把非线性方程 f(x)=0 线性化的一种近似方法。把 f(x)在点 x_0 的某邻域内展开成泰勒级数 $f(x)=f(x_0)+f'(x_0)(x-x_0)+f''(x_0)(x-x_0)^2/2!+...+f^{(n)}(x_0)(x-x_0)^n/n!+R(x_n)$,取其线性部分（即泰勒展开的前两项），并令其等于 0，即 $f(x_0)+f'(x_0)(x-x_0)=0$，以此作为非线性方程 f(x)=0 的近似方程，若 $f'(x_0)!=0$，则其解为 $x_1=x_0-f(x_0)/f'(x_0)$， 这样，得到牛顿迭代法的一个迭代关系式：$x_{n+1}=x_n-f(x_n)/f'(x_n)$。

已经证明，如果是连续的，并且待求的零点是孤立的，那么在零点周围存在一个区域，只要初始值位于这个邻近区域内，那么牛顿法必定收敛。 并且，如果不为 0, 那么牛顿法将具有平方收敛的性能. 粗略的说，这意味着每迭代一次，牛顿法结果的有效数字将增加一倍。

迭代法也称辗转法，是一种不断用变量的旧值递推新值的过程，跟迭代法相对应的是直接法（或者称为一次解法），即一次性解决问题。迭代算法是用计算机解决问题的一种基本方法。它利用计算机运算速度快、适合做重复性操作的特点，让计算机对一组指令（或一定步骤）重复执行，在每次执行这组指令（或这些步骤）时，都从变量的原值推出它的一个新值。

第三部分：程序设计流程


开 始
输 入 x_0, ε ,N
$1 \Rightarrow k$
$f'(x_0) \neq 0$?
n
y
$x_0 - \frac{f(x_0)}{f'(x_0)} \Rightarrow x_1$
$k+1 \Rightarrow k$
$x_1 \Rightarrow x_0$
$|x_1 - x_0| < \varepsilon$?
y
n
$k<N$?
y
n
输出迭代
失败标志
输出 x_1
输出奇
异标志
结 束

代码如下：

函数部分：

```
function a=newton(f,x0,e1,e2,N)
syms x F DF Tol x1;
n=1;
while n <= N
    F=subs(f,'x',x0);
    DF=subs(diff(f),'x',x0);
    if abs(F)<e1
        fprintf('x*=%.10f',x0);
        disp(' ');
        fprintf('迭代次数 n= %d',n);
        return;
   elseif abs(DF)<e2
            fprintf('输出失败');
            return;
    end
    x1=x0-F/DF;
    Tol=abs(x1-x0);
    if Tol<e1
        fprintf('x*=%.10f',x1);
        disp(' ');
        fprintf('迭代次数 n= %d',n);
        return;
    end
    n=n+1;
    x0=x1;
end
fprintf('输出失败');
return
```

问题 1（1）

```
disp('问题 1（1）');
syms x;
f(x)=cos(x)-x;
newton(f,pi/4,1e-6,1e-10,10);
```

问题 1（2）

```
disp('问题 1（2）');
syms x;
f(x)=exp(-x)-sin(x);
newton(f,0.6,1e-6,1e-10,10);
```

```
问题 2（1）
disp('问题 2（1）');
syms x;
f(x)=x-exp(-x);
newton(f,0.5,1e-6,1e-10,10);


问题 2（2）
disp('问题 2（2）');
syms x;
f(x)=x^2-2*x*exp(-x)+exp(-2*x);
newton(f,0.5,1e-6,1e-10,20);
```

第四部分：实验结果、结论与讨论

```
>> newton_1_1
问题1（1）
x*=0.7390851781
迭代次数n= 3>> newton_1_2
问题1（2）
x*=0.5885327428
迭代次数n= 3>> newton_2_1
问题2（1）
x*=0.5671431650
迭代次数n= 3>> newton_2_2
问题2（2）
```

```
>> newton_2_2
问题2（2）
x*=0.5666057041
迭代次数n= 8>>
```

思考题

1. 初值距离标准值越近越好，在实际计算中，可以先粗略的估计标准值的范围，最好是能精确到一位到两位小数，可以大大减少运算次数。
2. 正常来讲两问得到的零点是同一个，但是由于这个计算得到的是近似解而不是精确解，所以得到的答案有所不同。因为产生了舍入误差，且两个函数的导数不同，还有第一个函数是一个根，第二个函数是二重根，计算过程收敛速度

存在一阶和二阶的区别，所以得到的结果不同。

实验报告四

第一部分：问题分析 *（描述并总结出实验题目）*

摘要：

数值分析中，龙格库塔方法是用于非线性常微分方程的解的重要的一类隐式或显式迭代法。这些技术由数学家卡尔龙格和马丁威尔海姆库塔于 1900 年左右发明。

龙格库塔方法是一种子工程应用广泛的高精度单步算法，其中包括著名的欧拉法，用于数值求解微分方程。由于此算法精度高，采取措施对误差进行抑制，所以其实现原理也比较复杂。

目的：利用四阶龙格——库塔（Runge—Kutta）方法求解微分方程初值问题

意义：理解龙格库塔方法的原理，能够运用龙格库塔解决问题。

第二部分：数学原理

令初值问题表述如下。

$$y' = f(t,y), \quad y(t_0) = y_0$$

则，对于该问题的 RK4 由如下方程给出：

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

其中

$$k_1 = f(t_n, y_n)$$

$$k_2 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right)$$

$$k_3 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right)$$

$$k_4 = f(t_n + h, y_n + hk_3)$$

这样，下一个值（yn+1）由现在的值（yn）加上时间间隔（h）和一个估算的斜率的乘积 所决定。该斜率是以下斜率的加权平均：

- k1 是时间段开始时的斜率；
- k2 是时间段中点的斜率，通过欧拉法采用斜率 k1 来决定 y 在点 tn+h/2

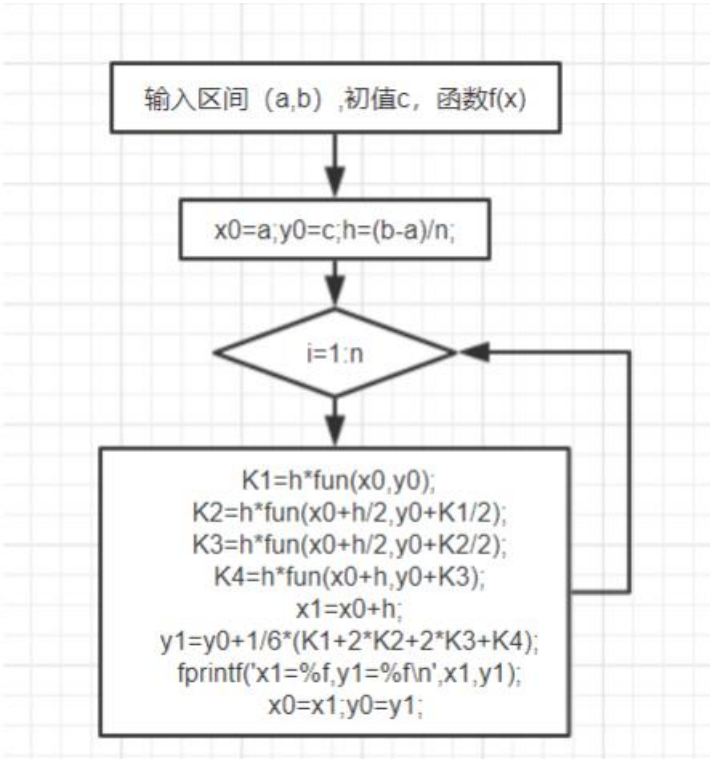
的值；

$$\text{slope} = \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}.$$

- k3 也是中点的斜率，但是这次采用斜率 k2 决定 y 值；
- k4 是时间段终点的斜率，其 y 值用 k3 决定。 当四个斜率取平均时，中点的斜率有更大的权值：

RK4 法是四阶方法，也就是说每步的误差是 h 阶，而总积累误差为 h 阶。注意上述公式对于标量或者向量函数（y 可以是向量）都适用。

第三部分：程序设计流程



代码如下：

```
function a=runge_kutta(f,x0,y0,b)
N=input('请输入 N 的值：');
syms x y;
h=(b-x0)/N;
for n=1:N
    F1=subs(f,'x',x0);
    F2=subs(F1,'y',y0);
    K1=double(h*F2);
    F3=subs(f,'x',x0+h/2);
    F4=subs(F3,'y',y0+K1/2);
    K2=double(h*F4);
    F5=subs(f,'x',x0+h/2);
    F6=subs(F5,'y',y0+K2/2);
    K3=double(h*F6);
    F7=subs(f,'x',x0+h);
    F8=subs(F7,'y',y0+K3);
    K4=double(h*F8);
    x1=x0+h;
    y1=double(y0+(K1+2*K2+2*K3+K4)/6);
    fprintf('x1=%.10f',x1);
    fprintf('y1=%.10f\n',y1);
    x0=x1;
    y0=y1;
end
```

问题 1（1）

```
disp('问题 1（1）');
syms x y;
f(x)=x+y;
runge_kutta(f,0,-1,1);
```

问题 1（2）

```
disp('问题 1（2）');
syms x y;
f(x)=-y^2;
runge_kutta(f,0,1,1);
```

问题 2（1）

```
disp('问题 2（1）');
syms x y;
f(x)=2*y/x+x*x*exp(x);
runge_kutta(f,1,0,3);
```

问题 2（2）

```
disp('问题 2（2）');
syms x y;
f(x)=(y^2+y)/x;
runge_kutta(f,1,-2,3);
```

问题 3（1）

```
disp('问题 3（1）');
syms x y;
f(x)=-20*(y-x^2)+2*x;
runge_kutta(f,0,1/3,1);
```

问题 3（2）

```
disp('问题 3（2）');
syms x y;
f(x)=-20*y+20*sin(x)+cos(x);
runge_kutta(f,0,1,1);
```

问题 3（3）

```
disp('问题 3（3）');
syms x y;
f(x)=-20*(y-exp(x)*sin(x))+exp(x)*(sin(x)+cos(x));
runge_kutta(f,0,0,1);
```

第四部分：实验结果、结论与讨论

问题 1（1）

```
问题1（1）
请输入N的值：5
x1=0.2000000000y1=-1.2000000000
x1=0.4000000000y1=-1.4000000000
x1=0.6000000000y1=-1.6000000000
x1=0.8000000000y1=-1.8000000000
x1=1.0000000000y1=-2.0000000000
>> runge_kutta_1_1
问题1（1）
请输入N的值：10
x1=0.1000000000y1=-1.1000000000
x1=0.2000000000y1=-1.2000000000
x1=0.3000000000y1=-1.3000000000
x1=0.4000000000y1=-1.4000000000
x1=0.5000000000y1=-1.5000000000
x1=0.6000000000y1=-1.6000000000
x1=0.7000000000y1=-1.7000000000
x1=0.8000000000y1=-1.8000000000
x1=0.9000000000y1=-1.9000000000
x1=1.0000000000y1=-2.0000000000
问题1（1）
请输入N的值：20
x1=0.0500000000y1=-1.0500000000
x1=0.1000000000y1=-1.1000000000
x1=0.1500000000y1=-1.1500000000
x1=0.2000000000y1=-1.2000000000
x1=0.2500000000y1=-1.2500000000
x1=0.3000000000y1=-1.3000000000
x1=0.3500000000y1=-1.3500000000
x1=0.4000000000y1=-1.4000000000
x1=0.4500000000y1=-1.4500000000
x1=0.5000000000y1=-1.5000000000
x1=0.5500000000y1=-1.5500000000
x1=0.6000000000y1=-1.6000000000
x1=0.6500000000y1=-1.6500000000
x1=0.7000000000y1=-1.7000000000
x1=0.7500000000y1=-1.7500000000
x1=0.8000000000y1=-1.8000000000
x1=0.8500000000y1=-1.8500000000
x1=0.9000000000y1=-1.9000000000
x1=0.9500000000y1=-1.9500000000
x1=1.0000000000y1=-2.0000000000
```

问题 1（2）

```
>> runge_kutta_1_2
问题1（2）
请输入N的值：5
x1=0.2000000000y1=0.8333390356
x1=0.4000000000y1=0.7142921305
x1=0.6000000000y1=0.6250058936
x1=0.8000000000y1=0.5555606879
x1=1.0000000000y1=0.5000044062
>> runge_kutta_1_2
问题1（2）
请输入N的值：10
x1=0.1000000000y1=0.9090911863
x1=0.2000000000y1=0.8333337288
x1=0.3000000000y1=0.7692312058
x1=0.4000000000y1=0.7142861539
x1=0.5000000000y1=0.6666670911
x1=0.6000000000y1=0.6250004009
x1=0.7000000000y1=0.5882356686
x1=0.8000000000y1=0.5555559032
x1=0.9000000000y1=0.5263161113
x1=1.0000000000y1=0.5000002976

>> runge_kutta_1_2
问题1（2）
请输入N的值：20
x1=0.0500000000y1=0.9523809630
x1=0.1000000000y1=0.9090909268
x1=0.1500000000y1=0.8695652397
x1=0.2000000000y1=0.8333333586
x1=0.2500000000y1=0.8000000269
x1=0.3000000000y1=0.7692307971
x1=0.3500000000y1=0.7407407689
x1=0.4000000000y1=0.7142857423
x1=0.4500000000y1=0.6896552000
x1=0.5000000000y1=0.6666666937
x1=0.5500000000y1=0.6451613166
x1=0.6000000000y1=0.6250000255
x1=0.6500000000y1=0.6060606307
x1=0.7000000000y1=0.5882353179
x1=0.7500000000y1=0.5714285944
x1=0.8000000000y1=0.5555555776
x1=0.8500000000y1=0.5405405618
x1=0.9000000000y1=0.5263158099
x1=0.9500000000y1=0.5128205325
x1=1.0000000000y1=0.5000000189
```

问题 2（1）

```
>> runge_kutta_2_1
问题2（1）
请输入N的值：5
x1=1.4000000000y1=2.6139427925
x1=1.8000000000y1=10.7763131664
x1=2.2000000000y1=30.4916542038
x1=2.6000000000y1=72.5855986060
x1=3.0000000000y1=156.2251982758
>> runge_kutta_2_1
问题2（1）
请输入N的值：10
x1=1.2000000000y1=0.8663791120
x1=1.4000000000y1=2.6197405205
x1=1.6000000000y1=5.7198952795
x1=1.8000000000y1=10.7920175975
x1=2.0000000000y1=18.6808523645
x1=2.2000000000y1=30.5215981354
x1=2.4000000000y1=47.8323658327
x1=2.6000000000y1=72.6345035377
x1=2.8000000000y1=107.6088519912
x1=3.0000000000y1=156.2982574429

>> runge_kutta_2_1
问题2（1）
请输入N的值：20
x1=1.1000000000y1=0.3459102873
x1=1.2000000000y1=0.8666216927
x1=1.3000000000y1=1.6071813477
x1=1.4000000000y1=2.6203113059
x1=1.5000000000y1=3.9676018980
x1=1.6000000000y1=5.7208793242
x1=1.7000000000y1=7.9637717926
x1=1.8000000000y1=10.7935017836
x1=1.9000000000y1=14.3229357276
x1=2.0000000000y1=18.6829265677
x1=2.1000000000y1=24.0249894197
x1=2.2000000000y1=30.5243558898
x1=2.3000000000y1=38.3834586600
x1=2.4000000000y1=47.8359047809
x1=2.5000000000y1=59.1510038275
x1=2.6000000000y1=72.6389257808
x1=2.7000000000y1=88.6565733309
x1=2.8000000000y1=107.6142643893
x1=2.9000000000y1=129.9833331157
x1=3.0000000000y1=156.3047718808
```

问题 2（2）

```
>> runge_kutta_2_2
问题2（2）
请输入N的值：5
x1=1.4000000000y1=-1.5539889981
x1=1.8000000000y1=-1.3836172899
x1=2.2000000000y1=-1.2934015269
x1=2.6000000000y1=-1.2375401579
x1=3.0000000000y1=-1.1995479585
>> runge_kutta_2_2
问题2（2）
请输入N的值：10
x1=1.2000000000y1=-1.7142451805
x1=1.4000000000y1=-1.5555228848
x1=1.6000000000y1=-1.4545197492
x1=1.8000000000y1=-1.3845945063
x1=2.0000000000y1=-1.3333158561
x1=2.2000000000y1=-1.2941026606
x1=2.4000000000y1=-1.2631447989
x1=2.6000000000y1=-1.2380836212
x1=2.8000000000y1=-1.2173808733
x1=3.0000000000y1=-1.1999905397
>> runge_kutta_2_2
问题2（2）
请输入N的值：20
x1=1.1000000000y1=-1.8333328294
x1=1.2000000000y1=-1.7142851698
x1=1.3000000000y1=-1.6249995002
x1=1.4000000000y1=-1.5555551111
x1=1.5000000000y1=-1.4999996057
x1=1.6000000000y1=-1.4545451028
x1=1.7000000000y1=-1.4166663505
x1=1.8000000000y1=-1.3846150982
x1=1.9000000000y1=-1.3571425958
x1=2.0000000000y1=-1.3333330933
x1=2.1000000000y1=-1.3124997783
x1=2.2000000000y1=-1.2941174411
x1=2.3000000000y1=-1.2777775857
x1=2.4000000000y1=-1.2631577147
x1=2.5000000000y1=-1.2499998307
x1=2.6000000000y1=-1.2380950784
x1=2.7000000000y1=-1.2272725761
x1=2.8000000000y1=-1.2173911609
x1=2.9000000000y1=-1.2083331969
x1=3.0000000000y1=-1.1999998699
```

问题 3（1）

```
>> runge_kutta_3_1
问题3（1）
请输入N的值：5
x1=0.2000000000y1=1.7600000000
x1=0.4000000000y1=8.8133333333
x1=0.6000000000y1=43.6800000000
x1=0.8000000000y1=217.2933333333
x1=1.0000000000y1=1084.3200000000
>> runge_kutta_3_1
问题3（1）
请输入N的值：10
x1=0.1000000000y1=0.1227777778
x1=0.2000000000y1=0.0792592593
x1=0.3000000000y1=0.1047530864
x1=0.4000000000y1=0.1665843621
x1=0.5000000000y1=0.2538614540
x1=0.6000000000y1=0.3629538180
x1=0.7000000000y1=0.4926512727
x1=0.8000000000y1=0.6425504242
x1=0.9000000000y1=0.8125168081
x1=1.0000000000y1=1.0025056027

>> runge_kutta_3_1
问题3（1）
请输入N的值：20
x1=0.0500000000y1=0.1275520833
x1=0.1000000000y1=0.0569466146
x1=0.1500000000y1=0.0401570638
x1=0.2000000000y1=0.0466734823
x1=0.2500000000y1=0.0650546392
x1=0.3000000000y1=0.0910100730
x1=0.3500000000y1=0.1229308607
x1=0.4000000000y1=0.1602136561
x1=0.4500000000y1=0.2026322044
x1=0.5000000000y1=0.2501016600
x1=0.5500000000y1=0.3025902058
x1=0.6000000000y1=0.3600859105
x1=0.6500000000y1=0.4225842998
x1=0.7000000000y1=0.4900836957
x1=0.7500000000y1=0.5625834692
x1=0.8000000000y1=0.6400833843
x1=0.8500000000y1=0.7225833524
x1=0.9000000000y1=0.8100833405
x1=0.9500000000y1=0.9025833360
x1=1.0000000000y1=1.0000833343
```

问题 3（2）

```
>> runge_kutta_3_2
问题3（2）
请输入N的值：5
x1=0.2000000000y1=5.1973381062
x1=0.4000000000y1=25.3761707044
x1=0.6000000000y1=125.4868152611
x1=0.8000000000y1=625.3120955171
x1=1.0000000000y1=3123.7951509472
>> runge_kutta_3_2
问题3（2）
请输入N的值：10
x1=0.1000000000y1=0.4331389965
x1=0.2000000000y1=0.3096604680
x1=0.3000000000y1=0.3323246667
x1=0.4000000000y1=0.4014139713
x1=0.5000000000y1=0.4830743415
x1=0.6000000000y1=0.5654352797
x1=0.7000000000y1=0.6439890045
x1=0.8000000000y1=0.7167223471
x1=0.9000000000y1=0.7824991512
x1=1.0000000000y1=0.8405257206
>> runge_kutta_3_2
问题3（2）
请输入N的值：20
x1=0.0500000000y1=0.4249785186
x1=0.1000000000y1=0.2404562221
x1=0.1500000000y1=0.2021684390
x1=0.2000000000y1=0.2184386634
x1=0.2500000000y1=0.2548116511
x1=0.3000000000y1=0.2982910222
x1=0.3500000000y1=0.3439285511
x1=0.4000000000y1=0.3897953364
x1=0.4500000000y1=0.4350961733
x1=0.5000000000y1=0.4794626229
x1=0.5500000000y1=0.5226880879
x1=0.6000000000y1=0.5646286384
x1=0.6500000000y1=0.6051659863
x1=0.7000000000y1=0.6441937626
x1=0.7500000000y1=0.6816125255
x1=0.8000000000y1=0.7173280379
x1=0.8500000000y1=0.7512507634
x1=0.9000000000y1=0.7832958132
x1=0.9500000000y1=0.8133830538
x1=1.0000000000y1=0.8414372689
```

问题 3（3）

```
>> runge_kutta_3_3
问题3（3）
请输入N的值：5
x1=0.2000000000y1=0.2986462128
x1=0.4000000000y1=0.9272198700
x1=0.6000000000y1=2.8354773389
x1=0.8000000000y1=10.7108853309
x1=1.0000000000y1=47.9414463816
>> runge_kutta_3_3
问题3（3）
请输入N的值：10
x1=0.1000000000y1=0.1120551091
x1=0.2000000000y1=0.2451165144
x1=0.3000000000y1=0.4017780967
x1=0.4000000000y1=0.5840969566
x1=0.5000000000y1=0.7938220530
x1=0.6000000000y1=1.0324183053
x1=0.7000000000y1=1.3010149884
x1=0.8000000000y1=1.6003210120
x1=0.9000000000y1=1.9305210338
x1=1.0000000000y1=2.2911569231

>> runge_kutta_3_3
问题3（3）
请输入N的值：20
x1=0.0500000000y1=0.0525950400
x1=0.1000000000y1=0.1104089863
x1=0.1500000000y1=0.1737093905
x1=0.2000000000y1=0.2427490009
x1=0.2500000000y1=0.3177716916
x1=0.3000000000y1=0.3990135525
x1=0.3500000000y1=0.4867020696
x1=0.4000000000y1=0.5810544891
x1=0.4500000000y1=0.6822757725
x1=0.5000000000y1=0.7905562930
x1=0.5500000000y1=0.9060693250
x1=0.6000000000y1=1.0289683441
x1=0.6500000000y1=1.1593841417
x1=0.7000000000y1=1.2974217528
x1=0.7500000000y1=1.4431571960
x1=0.8000000000y1=1.5966340222
x1=0.8500000000y1=1.7578596710
x1=0.9000000000y1=1.9268016338
x1=0.9500000000y1=2.1033834233
x1=1.0000000000y1=2.2874803507
```

思考题：

1.相同，因为实验 1 的函数都是简单的初等函数，截断误差 Tn 为 0。

2.N 越大越精确，N 越大，步长越小。计算次数越多，截断误差也越小。

3.本身函数的变化趋势比较复杂，N 较小时，得到的解与准确解相差很大，因为当步长大，计算次数少，误差也自然很大。