|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | |
|  | | | | |
|  | | | | |
|  | 姓名 | ： | 牟鑫一 |  |
|  | 班号 | ： | 191174 |  |
|  | 学号 | ： | 20161001764 |  |
|  | 院（系） | ： | 计算机学院 |  |
|  | 专业 | ： | 计算机科学与技术 |  |
|  | 指导老师 | ： | 葛富东 |  |
|  |  |  |  |  |
| 2019 年 6 月 | | | | |

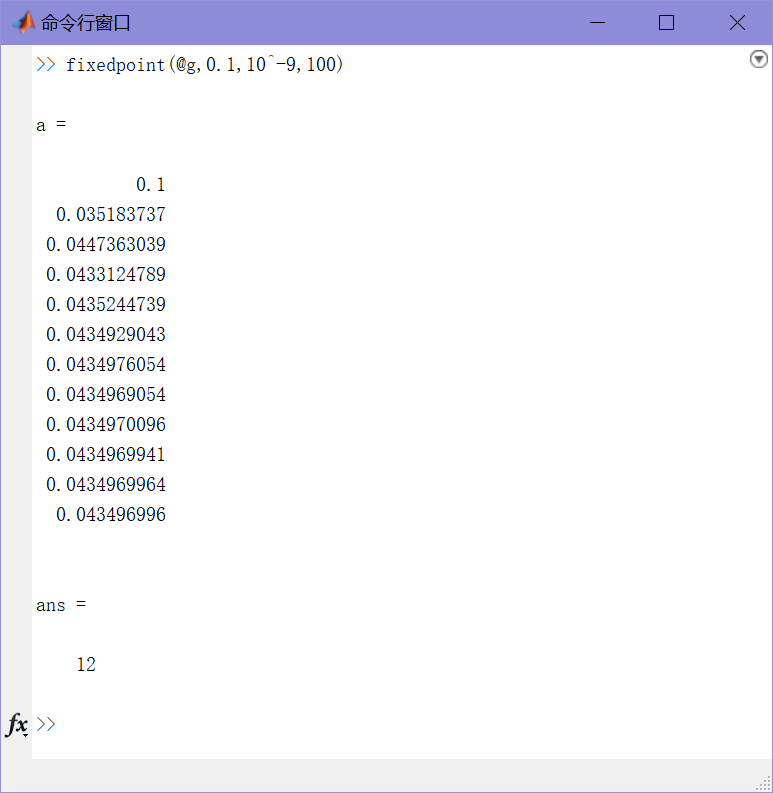
1. 求解非线性方程（精确到小数点后9位）
2. 运用不动点迭代法求解非线性方程 在区间 [0,1]内的解.

**fixpt.m**

1. function[k,p,err,P]=fixpt(g,p0,to1,max1)
3. % 测试: fixpt(@g,0.1,10^-9,100)
4. % p0初值
5. % tol精度
6. % maxl迭代次数
8. P(1)=p0;
9. for k=2:max1
10. P(k)=feval(g,P(k-1));
11. err=abs(P(k)-P(k-1));
12. relerr=err/(abs(P(k))+eps);
13. p=P(k);
14. if(err<to1)|(relerr<to1),break;
15. end
16. end
17. if k==max1
18. disp('maximum number of iterations exceeded')
19. end
20. P=P';
21. a=vpa(P,9)
22. end

**g.m**

1. function f=g(x)
2. f=exp(log(20)/(x-cos(x)));



1. 运用牛顿法求解非线性方程在区间 [0,1]内的解.

**newton\_iteration.m**

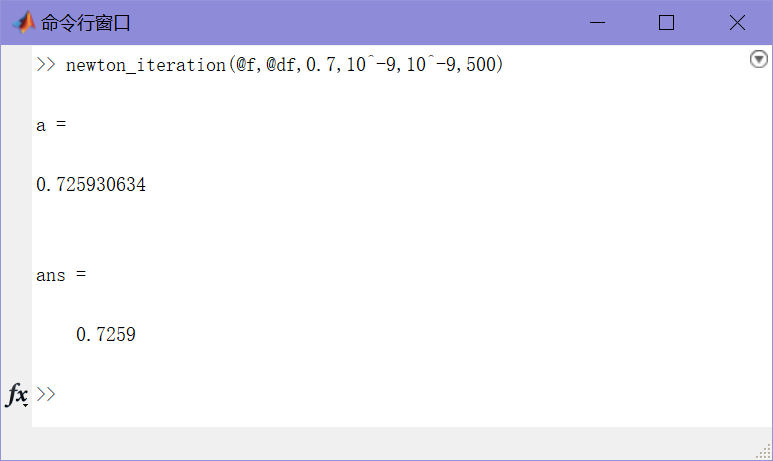
1. function [p0,err,k,y]=newton\_iteration(f,df,p0,delta,epsilon,max1)
2. for k=1:max1
3. p1=p0-f(p0)/df(p0);
4. err=abs(p1-p0);
5. relerr=2\*err/(abs(p1)+delta);
6. p0=p1;
7. y=f(p0);
8. if (err<delta)|(relerr<delta)|(abs(y)<epsilon),
9. break,
10. end
11. end
12. a=vpa(p0,9)
14. % 测试：newton\_iteration(@f,@df,0.7,10^-9,10^-9,500)

**f.m**

1. function  f=f(x)
2. f=(x^5)-(3\*x^3)-(2\*x^2)+2;

**df.m**

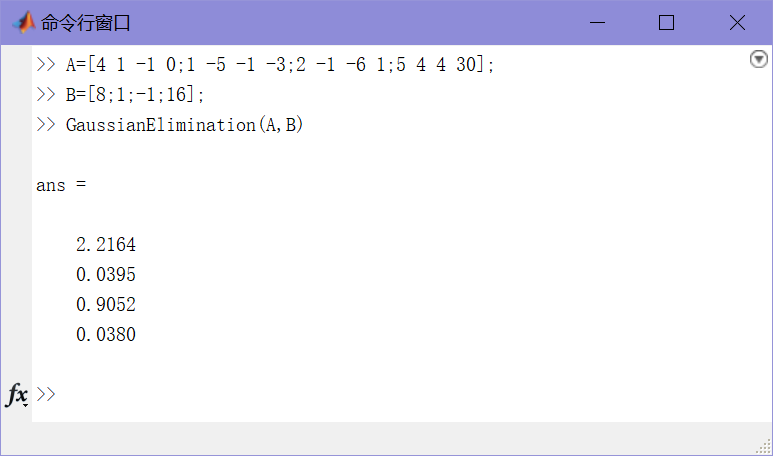
1. function  f=f(x)
2. f=(5\*x^4)-(9\*x^2)-(4\*x);



1. 求解线性方程组（分别用Gauss消去法、Jacobi迭代法和Gauss–Seidel迭代法求解，精确到小数点后4位）

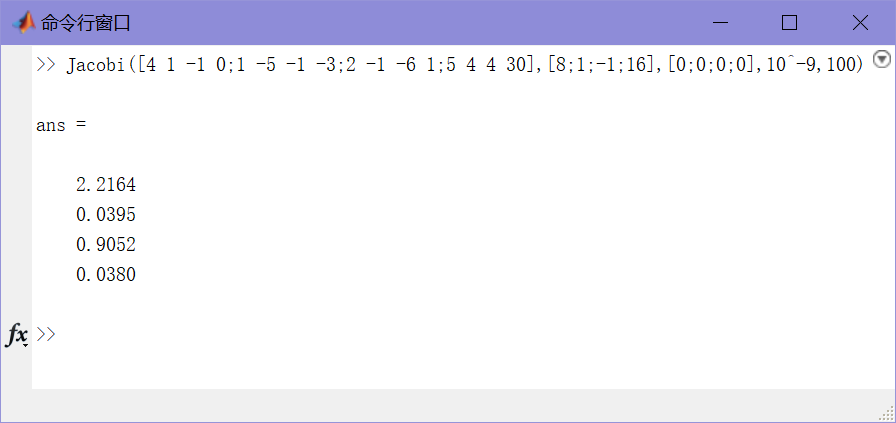
**GaussianElimination.m**

1. function X = GaussianElimination(A,B)
3. % A=[4 1 -1 0;1 -5 -1 -3;2 -1 -6 1;5 4 4 30];
4. % B=[8;1;-1;16];
5. % GaussianElimination(A,B)
7. [N N]=size(A);
8. X=zeros(N,1);
9. C=zeros(1,N+1);
11. Aug=[A B];
13. for p=1:N-1
14. [Y,j]=max(abs(Aug(p:N,p)));
15. C=Aug(p,:);
16. Aug(p,:)=Aug(j+p-1,:);
17. Aug(j+p-1,:)=C;
18. if Aug(p,p)==0
19. 'A was singular.  No unique solution';
20. break
21. end
22. for k=p+1:N
23. m=Aug(k,p)/Aug(p,p);
24. Aug(k,p:N+1)=Aug(k,p:N+1)-m\*Aug(p,p:N+1);
25. end
26. end
27. X=backsub(Aug(1:N,1:N),Aug(1:N,N+1));



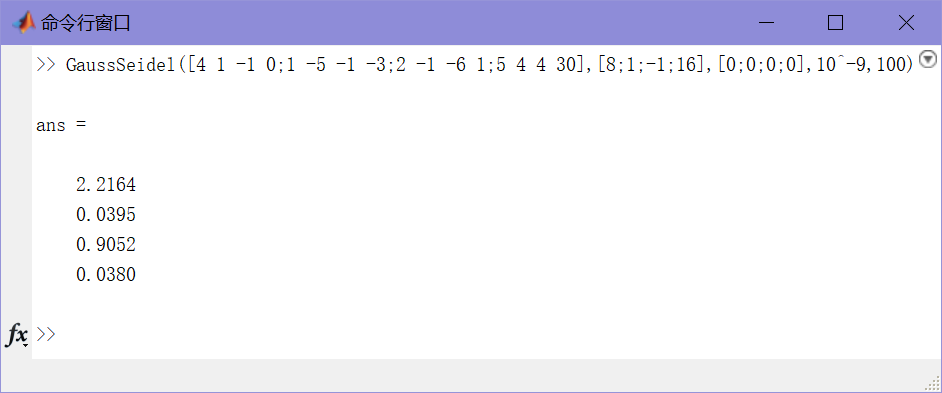
**Jacobi.m**

1. function X=Jacobi(A,B,P,delta, max1)
3. N = length(B);
4. for k=1:max1
5. for j=1:N
6. X(j)=(B(j)-A(j,[1:j-1,j+1:N])\*P([1:j-1,j+1:N]))/A(j,j);
7. end
8. err=abs(norm(X'-P));
9. relerr=err/(norm(X)+eps);
10. P=X';
11. if (err<delta)|(relerr<delta)
12. break
13. end
14. end
16. X=X';



**GaussSeidel.m**

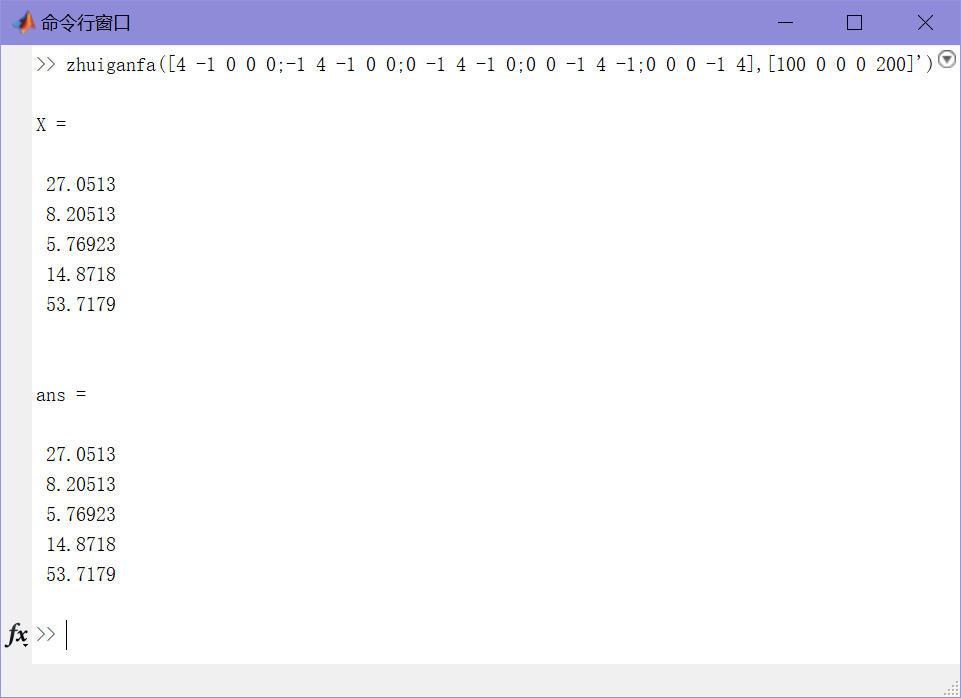
1. function X=GaussSeidel(A,B,P,delta, max1)
3. % GaussSeidel([4 1 -1 0;1 -5 -1 -3;2 -1 -6 1;5 4 4 30],[8;1;-1;16],[0;0;0;0],10^-9,100)
5. N = length(B);
7. for k=1:max1
8. for j=1:N
9. if j==1
10. X(1)=(B(1)-A(1,2:N)\*P(2:N))/A(1,1);
11. elseif j==N
12. X(N)=(B(N)-A(N,1:N-1)\*(X(1:N-1))')/A(N,N);
13. else
14. X(j)=(B(j)-A(j,1:j-1)\*X(1:j-1)'-A(j,j+1:N)\*P(j+1:N))/A(j,j);
15. end
16. end
17. err=abs(norm(X'-P));
18. relerr=err/(norm(X)+eps);
19. P=X';
20. if (err<delta)|(relerr<delta)
21. break
22. end
23. end
24. X=X';



1. 用追赶法求解下列方程组（6位有效数字）。

**zhuiganfa.m**

1. function X = zhuiganfa(A,B)
2. [N,N]=size(A);
3. X=zeros(N,1);   Y=zeros(N,1);
4. C=zeros(1,N);  R=1:N;
5. for p=1:N-1
6. [max1,j]=max(abs(A(p:N,p)));
7. C=A(p,:);
8. A(p,:)=A(j+p-1,:);
9. A(j+p-1,:)=C;
10. d=R(p);
11. R(p)=R(j+p-1);
12. R(j+p-1)=d;
13. if A(p,p)==0
14. 'A is singular.  No unique solution';
15. break
16. end
17. for k=p+1:N
18. mult=A(k,p)/A(p,p);
19. A(k,p) = mult;
20. A(k,p+1:N)=A(k,p+1:N)-mult\*A(p,p+1:N);
21. end
22. end
23. Y(1) = B(R(1));
24. for k=2:N
25. Y(k)= B(R(k))-A(k,1:k-1)\*Y(1:k-1);
26. end
27. X(N)=Y(N)/A(N,N);
28. for k=N-1:-1:1
29. X(k)=(Y(k)-A(k,k+1:N)\*X(k+1:N))/A(k,k);
30. end
31. X=vpa(X,6)



1. 某气象观测站测得某日6:00-18:00之间每隔2小时的温度如下：

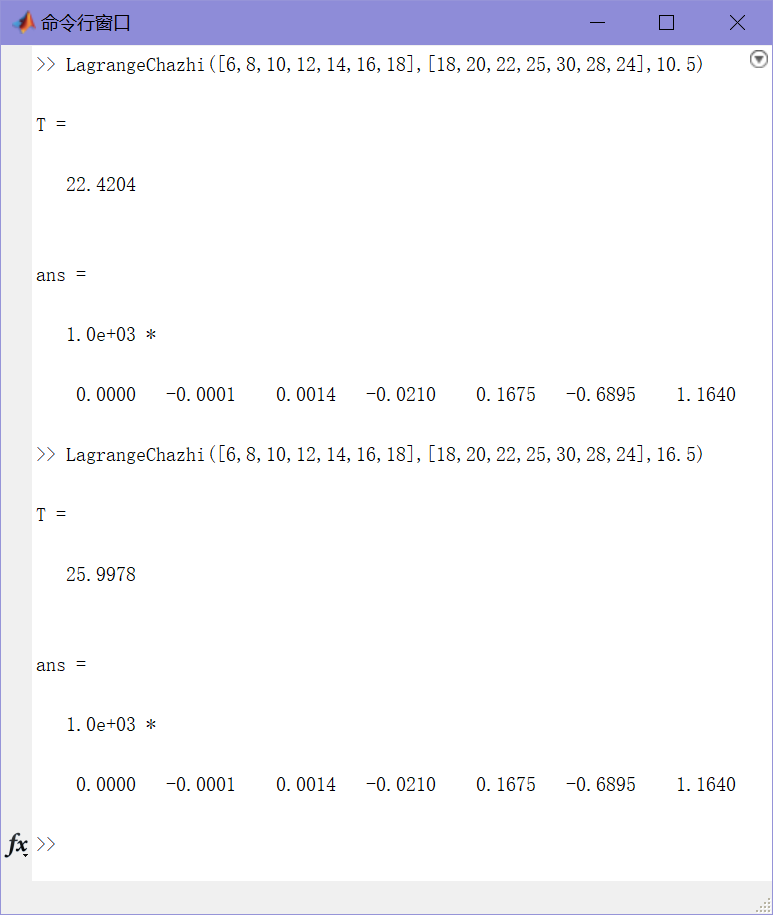
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 时间 | 6 | 8 | 10 | 12 | 14 | 16 | 18 |
| 温度 | 18 | 20 | 22 | 25 | 30 | 28 | 24 |

分别用Lagrange插值和Newton插值求出该日10:30, 16:30的温度。

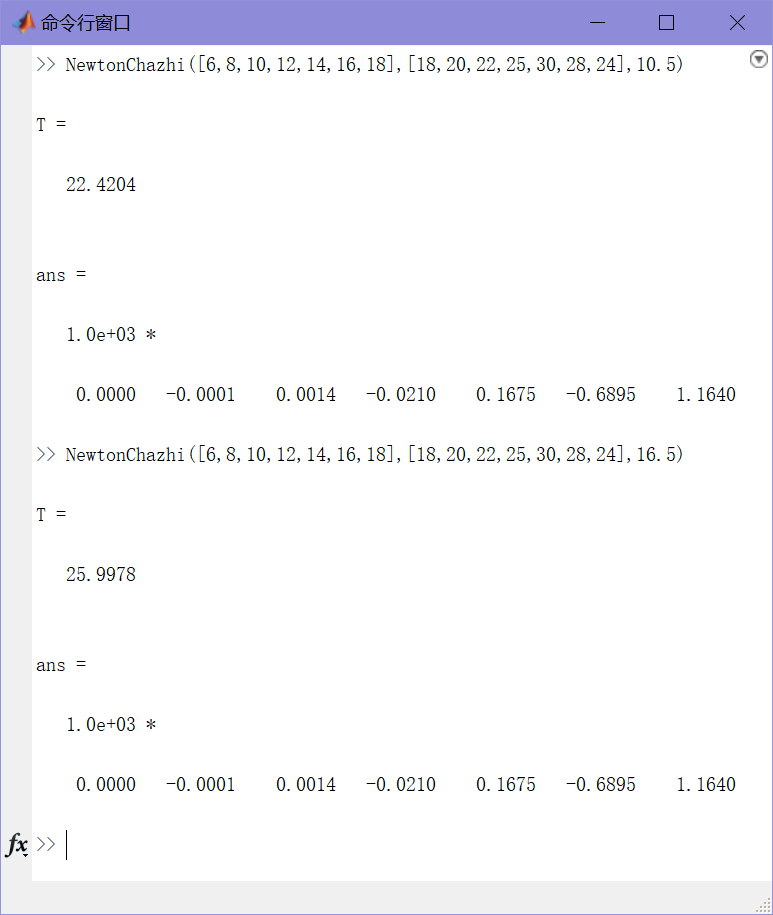
**Lagrange插值:**

**LagrangeChazhi.m**

1. function C=LagrangeChazhi(X,Y,t)
3. %Input    - X is a vector that contains a list of abscissas
4. %            - Y is a vector that contains a list of ordinates
5. %Output - C is a matrix that contains the coefficents of
6. %              the Lagrange interpolatory polynomial
7. % - L is a matrix that contains the Lagrange coefficient polynomials
8. % X=[0,1,2,4];  Y=[1,9,23,3];  C=lagrangeChazhi(X,Y)
9. w=length(X);
10. n=w-1;
11. L=zeros(w,w);
12. %Form the Lagrange coefficient polynomials
13. for k=1:n+1
14. V=1;
15. for j=1:n+1
16. if k~=j
17. V=conv(V,poly(X(j)))/(X(k)-X(j));
18. end
19. end
20. L(k,:)=V;
21. end
22. %Determine the coefficients of the Lagrange interpolator polynomial
23. C=Y\*L;
24. T=C(1)\*t^6+C(2)\*t^5+C(3)\*t^4+C(4)\*t^3+C(5)\*t^2+C(6)\*t+C(7)



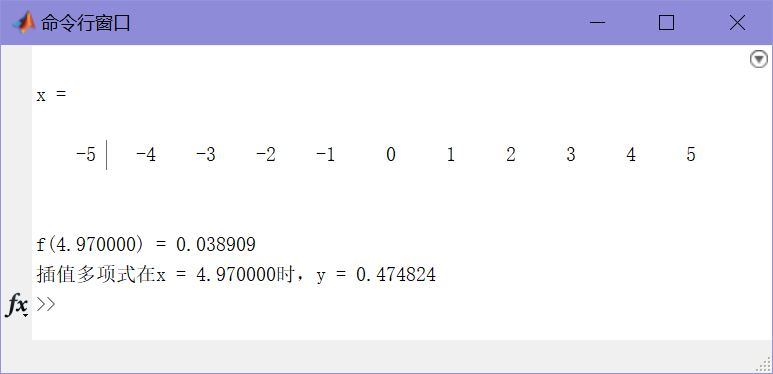
**Newton插值：**



1. 利用Lagrange插值多项式验证Runge现象，即10等分区间[-5,5]. 求函数关于分割点的插值多项式在处的值，并与比较。

直接运行以下代码：

1. clear
2. clc
3. x=-5:1:5
4. y=1./(1+x.^2);
5. plot(x,y);
6. xx=4.97;
7. yy=0;
8. for i=1:11
9. temp=y(i);
10. for j=1:11
11. if j~=i
12. temp=temp\*(xx-x(j))/(x(i)-x(j));
13. end
14. end
15. yy=yy+temp;
16. end
17. plot(x,yy);
18. fprintf('\nf(%f) = %f',xx,1/(1+xx\*xx));
19. fprintf('\n插值多项式在x = %f时，y = %f\n',xx,yy)



1. 经调查15个人，他们的体重与身高的数据如下：



试用数据建模的方法建立体重（y）与身高（x）的关系。

命令行输入：

1. >>> X=[0.75,0.86,0.96,1.08,1.12,1.26,1.35,1.51,1.55,1.60,1.63,1.67,1.71,1.78,1.85];
2. >>> Y=[10,12,15,17,20,27,35,41,48,50,51,54,59,66,75];
3. >>> cftool

设置X data为X，Y data为Y

