计 算 方 法

实验指导与实验报告

姓名 张瑞

学号 1190201421

院系 计算学部

专业 计算机系

哈尔滨工业大学

**实验报告一**

|  |
| --- |
| 题目（摘要）  利用拉格朗日插值多项式求的近似值  输 入：个数据点，；插值点  输 出：在插值点的近似值  前 言（目的和意义）  实验目的：利用拉格朗日插值多项式求的近似值。  实验意义：熟悉拉格朗日插值方法，学会用拉格朗日插值法求解函数近似值，利用计算机程序解决现实中的计算问题。  数学原理  给定平面上个不同的数据点，，，；则满足条件  ，  的次拉格朗日插值多项式  是存在唯一的。若，且函数充分光滑，则当时，有误差估计式  ，  程序设计流程  程序流程：  1 置；  2 当时，做2.1—2.4  2.1 置；  2.2 对，置  2.3 置  2.4 置  3 输出  4 停机 |
| 用Matlab实现的具体代码如下：  function [ y ] = Lagrange( x0,y0,x )  %Lagrange Lagrange插值法求解插值点近似值  %x0,y0为数据点及其函数值;x,y为插值点及其近似值  if length(x0)==length(y0)  n=length(x0);  y=0.0;  for k=1:n  l=1.0;  for j=1:n  if j~=k  l=l\*(x-x0(j))/(x0(k)-x0(j));  end  end  y=y+l\*y0(k);  end  else  disp("输入数据点错误");  end  end  实验结果、结论与讨论  问题1 拉格朗日插值多项式的次数越大越好吗？  考虑下面两个拉格朗日插值问题：  （1）设，，考虑等距节点的拉格朗日插值多项式，即将区间进行等分，记，，，构造，利用拉格朗日插值多项式作为的近似值。分别取，，，同时计算在，，，，处的函数值。   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 插值点x | 真实值f(x) | P(x)(n=5) | P(x)(n=10) | P(x)(n=20) | | 0.750000 | 0.640000 | 0.528974 | 0.678990 | 0.636755 | | 1.750000 | 0.246154 | 0.373325 | 0.190580 | 0.238446 | | 2.750000 | 0.116788 | 0.153733 | 0.215592 | 0.080660 | | 3.750000 | 0.066390 | -0.025954 | -0.231462 | -0.447052 | | 4.750000 | 0.042440 | -0.015738 | 1.923631 | -39.952449 |   （2）设，，考虑等距节点的拉格朗日插值多项式，即将区间进行等分，记，，，构造，利用拉格朗日插值多项式作为的近似值。分别取，，，同时计算在，，，处的函数值。 |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | 插值点x | 真实值f(x) | P(x)(n=5) | P(x)(n=10) | P(x)(n=20) | | -0.950000 | 0.386741 | 0.386798 | 0.386741 | 0.386741 | | -0.050000 | 0.951229 | 0.951248 | 0.951229 | 0.951229 | | 0.050000 | 1.051271 | 1.051290 | 1.051271 | 1.051271 | | 0.950000 | 2.585710 | 2.585785 | 2.585710 | 2.585710 |   结论：拉格朗日插值多项式的次数n并非越大越好，上面两个问题中的（1）即是一个反例，当插值点为4.75时，随着n的增加，偏差反而越来越大，逼近效果并不理想。这一现象即为龙格现象。要慎重使用次数过高的插值多项式；适当选择插值用的节点，例如用切比雪夫插值节点；也可以采用分段线性插值和三次样条插值等方式获得较好的逼近效果。  问题2 插值区间越小越好吗？  考虑下面两个拉格朗日插值问题：  （1）设，，考虑等距节点的拉格朗日插值多项式，即将区间进行等分，记，，，构造，利用拉格朗日插值多项式作为的近似值。分别取，，，同时计算在，，，处的函数值。   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 插值点x | 真实值f(x) | P(x)(n=5) | P(x)(n=10) | P(x)(n=20) | | -0.950000 | 0.525624 | 0.517147 | 0.526408 | 0.525620 | | -0.050000 | 0.997506 | 0.992791 | 0.997507 | 0.997506 | | 0.050000 | 0.997506 | 0.992791 | 0.997507 | 0.997506 | | 0.950000 | 0.525624 | 0.517147 | 0.526408 | 0.525620 |   （2）设，，考虑等距节点的拉格朗日插值多项式，即将区间进行等分，记，，，构造，利用拉格朗日插值多项式作为的近似值。分别取，，，同时计算在，，，处的函数值。   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 插值点x | 真实值f(x) | P(x)(n=5) | P(x)(n=10) | P(x)(n=20) | | -4.750000 | 0.008652 | 1.147035 | -0.001957 | 0.008652 | | -0.250000 | 0.778801 | 1.302152 | 0.778686 | 0.778801 | | 0.250000 | 1.284025 | 1.841210 | 1.284144 | 1.284025 | | 4.750000 | 115.584285 | 119.621007 | 115.607360 | 115.584285 |   结论：对于同一个函数，在等分段数相同情况下，插值区间越小的，插值点处的偏差越小，逼近效果越理想。因为同一函数等分段数相同，插值区间为小区间时，函数更不容易发生剧烈变化，更不容易出现较大摆动，则此时进行拟合逼近的效果更好更准确。  问题3 在区间考虑拉格朗日插值问题，为了使得插值误差较小，应如何选取插值节点？  考虑下面两个拉格朗日插值问题：  （1）设，，考虑非等距节点的拉格朗日插值多项式，记，，构造，利用拉格朗日插值多项式作为的近似值。分别取，，，同时计算在，，，处的函数值。   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 插值点x | 真实值f(x) | P(x)(n=5) | P(x)(n=10) | P(x)(n=20) | | -0.950000 | 0.525624 | 0.523881 | 0.525682 | 0.525624 | | -0.050000 | 0.997506 | 0.987881 | 0.997509 | 0.997506 | | 0.050000 | 0.997506 | 0.987881 | 0.997509 | 0.997506 | | 0.950000 | 0.525624 | 0.523881 | 0.525682 | 0.525624 |   （2）设，，考虑非等距节点的拉格朗日插值多项式，记，，构造，利用拉格朗日插值多项式作为的近似值。分别取，，，同时计算在，，，处的函数值。   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 插值点x | 真实值f(x) | P(x)(n=5) | P(x)(n=10) | P(x)(n=20) | | -0.950000 | 0.386741 | 0.386754 | 0.386741 | 0.386741 | | -0.050000 | 0.951229 | 0.951272 | 0.951229 | 0.951229 | | 0.050000 | 1.051271 | 1.051314 | 1.051271 | 1.051271 | | 0.950000 | 2.585710 | 2.585727 | 2.585710 | 2.585710 |   结论：这一问的插值节点的选取使得结果的精确程度显著提高，查阅资料发现，这种节点的选取方式是切比雪夫插值节点，节点的分布为两端密集中间稀疏，这种方式能尽可能地分散插值误差，最大限度地降低龙格现象。  问题4 考虑拉格朗日插值问题，内插比外推更可靠吗？  考虑下面两个拉格朗日插值问题：  （1）设，关于以，，为节点的拉格朗日插值多项式，利用拉格朗日插值多项式作为的近似值。同时计算在，，，处的函数值。  （2）设，关于以，，为节点的拉格朗日插值多项式，利用拉格朗日插值多项式作为的近似值。同时计算在，，，处的函数值。  （3）设，关于以，，为节点的拉格朗日插值多项式，利用拉格朗日插值多项式作为的近似值。同时计算在，，，处的函数值。  （4）设，关于以，，为节点的拉格朗日插值多项式，利用拉格朗日插值多项式作为的近似值。同时计算在，，，处的函数值。   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 插值点x | 真实值f(x) | P(x)(1) | P(x)(2) | P(x)(3) | P(x)(4) | | 5 | 2.236068 | 2.266667 | 3.115751 | 4.439112 | 5.497172 | | 50 | 7.071068 | -20.233333 | 7.071795 | 7.284961 | 7.800128 | | 115 | 10.723805 | -171.90000 | 10.167033 | 10.722756 | 10.800493 | | 185 | 13.601471 | -492.73333 | 10.038828 | 13.535667 | 13.600620 |   结论：内插即插值点的值介于已知函数值的节点之间，外推即插值点的值大于或小于所有的已知节点。由上述实验我们可以看出，内插时的误差小于外推的误差，则可以认为内插比外推可靠。 |

**实验报告二**

|  |
| --- |
| 题目（摘要）  利用四阶龙格—库塔(Runge—Kutta)方法求解微分方程初值问题  输 入：  输 出：初值问题的数值解，  前 言（目的和意义）  实验目的：利用四阶龙格—库塔(Runge—Kutta)方法求解微分方程初值问题。  实验意义：熟悉四阶龙格—库塔方法，根据实际问题建立数学模型，借助计算机程序解决现实问题。  数学原理  给定常微分方程初值问题  记，，利用四阶龙格—库塔方法      可逐次求出微分方程初值问题的数值解，  程序设计流程  程序流程：  1 置  2 对，做2.1—2.4 |
| 2.1 置  2.2 置  2.3 输出  2.4 置  3 停机  用Matlab实现的具体代码如下：  function [ x,y ] = RungeKutta( a,b,alpha,N )  %RungeKutta 四阶龙格库塔方法求解微分方程初值问题  %x,y为初值问题数值解，a,b为区间，alpha为a处的值，N为等分的段数  x=ones(1,N);  y=ones(1,N);  x0=a;  y0=alpha;  h=(b-a)/N;  for n=1:N  K1=h\*fun(x0,y0);  K2=h\*fun(x0+h/2,y0+K1/2);  K3=h\*fun(x0+h/2,y0+K2/2);  K4=h\*fun(x0+h,y0+K3);  x1=x0+h;  y1=y0+(K1+2\*K2+2\*K3+K4)/6;  x(n)=x1;  y(n)=y1;  x0=x1;  y0=y1;  end  实验结果、结论与讨论  （以下各题中，xn和yn均为使用四阶龙格库塔法求出的结果，y为使用精确的方程求出的结果）  问题1  （1）  准确解    N=5:  xn= [0.2,0.4,0.6,0.8,1.0]  yn= [-1.2,-1.4,-1.6,-1.8,-2.0]  y=[-1.2,-1.4,-1.6,-1.8,-2.0]  N=10:  xn=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]  yn=[-1.1,-1.2,-1.3,-1.4,-1.5,-1.6,-1.7,-1.8,-1.9,-2.0]  y=[-1.1,-1.2,-1.3,-1.4,-1.5,-1.6,-1.7,-1.8,-1.9,-2.0]  N=20:  xn=[0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,0.85,0.90,0.95,1.00]  yn=[-1.05,-1.10,-1.15,-1.20,-1.25,-1.30, -1.35,-1.40,-1.45,-1.50,-1.55,-1.60,-1.65,-1.70,-1.75,-1.80,-1.85,-1.90,-1.95,-2.00]  y=[-1.05,-1.10,-1.15,-1.20,-1.25,-1.30, -1.35,-1.40,-1.45,-1.50,-1.55,-1.60,-1.65,-1.70,-1.75,-1.80,-1.85,-1.90,-1.95,-2.00]  （2）  准确解    N=5:  xn=[0.2,0.4,0.6,0.8,1.0]  yn=[0.833339035623039,0.714292130463543,0.625005893608534,0.555560687934186,0.500004406158226]  y=[0.83333333,0.71428571,0.625,0.55555556,0.5]  N=10:  xn=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]  yn=[0.909091186332220,0.833333728843072,0.769231205753286,0.714286153892761,0.666667091065863,0.625000400949174,0.588235668580839,0.555555903183214,0.526316111264258,0.500000297580231]  y=[0.90909091,0.83333333,0.76923077,0.71428571,0.66666667,0.625,0.58823529,0.55555556,0.52631579,0.5]  N=20:  xn=[0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,0.85,0.90,0.95,1.00]  yn=[0.952380963026982,0.909090926812539,0.869565239726747,0.833333358572267,0.800000026948111,0.769230797052093,0.740740768850604,0.714285742277112,0.689655200006180,0.666666693669981,0.645161316611732,0.625000025496636,0.606060630719991,0.588235317919164,0.571428594368805,0.555555577643246,0.540540561792882,0.526315809913613,0.512820532474715,0.500000018897452]  y=[0.95238095,0.90909091,0.86956522,0.83333333,0.8,0.76923077,0.74074074,0.71428571,0.68965517,0.66666667,0.64516129,0.625,0.60606061,0.58823529,0.57142857,0.55555556,0.54054054,0.52631579,0.51282051,0.5]  分析：（1）中数值解和解析解相同，（2）中略有不同。因为四阶龙格库塔方法以线性算法来近似求解数值解，（1）中的准确解满足线性，故数值解和解析解相同，而（2）却不满足。  问题2  （1）    准确解  N=5:  xn=[1.4,1.8,2.2,2.6,3.0]  yn= [2.613942792503426,10.776313166418577,30.491654203794226,72.585598606012210,156.2251982758480]  y= [2.620359551235833,10.793624660490630,30.524581287519347,72.639283956228240,156.3052958525575]  N=10:  xn=[1.2,1.4,1.6,1.8,2.0,2.2,2.4,2.6,2.8,3.0]  yn= [0.866379111974020,2.619740520468712,5.719895279538559,10.792017597489250,18.680852364517300,30.521598135366503,47.832365832693654,72.634503537672020,107.6088519911855,156.2982574428725]  y= [0.866642535759602,2.620359551235833,5.720961525596334,10.793624660490630,18.683097081886410,30.524581287519347,47.836192620571520,72.639283956228280,107.6147011502820,156.3052958525578]  N=20:  xn=[1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9,2.0,2.1,2.2,2.3,2.4,2.5,2.6,2.7,2.8,2.9,3.0]  yn= [0.345910287306440,0.866621692728884,1.60718134766403,2.62031130587181,3.96760189798804,5.72087932424446,7.96377179260462,10.7935017836485,14.3229357275887,18.6829265676522,24.0249894196646,30.5243558898292,38.3834586600260,47.8359047809372,59.1510038275214,72.6389257808317,88.6565733309189,107.614264389308,129.983333115654,156.304771880838]  y=[0.345919876539739,0.866642535759604,1.60721507818074,2.62035955123584,3.96766629422780,5.72096152559635,7.96387347784498,10.7936246604907,14.3230815358910,18.6830970818865,24.0251864509190,30.5245812875194,38.3837143134216,47.8361926205717,59.1513258265278,72.6392839562285,88.6569697448568,107.614701150282,129.983812379676,156.305295852558]  （2）    准确解    N=5:  xn=[1.4,1.8,2.2,2.6,3.0]  yn=[-1.553988998095238,-1.383617289911493,-1.293401526919330,  -1.237540157935232,-1.199547958457927]  y=[-1.555555555555556,-1.384615384615385,-1.294117647058824,  -1.238095238095238,-1.200000000000000]  N=10:  xn=[1.2,1.4,1.6,1.8,2.0,2.2,2.4,2.6,2.8,3.0]  yn=[-1.714245180451154,-1.555522884849619,-1.454519749200756,  -1.384594506286678,-1.333315856075274,-1.294102660572944,-1.263144798904635,  -1.238083621168147,-1.217380873320439,-1.199990539708786]  y=[-1.714285714285714,-1.555555555555556,-1.454545454545455,  -1.384615384615385,-1.333333333333334,-1.294117647058824,-1.263157894736842,  -1.238095238095238,-1.217391304347826,-1.200000000000000]  N=20:  xn=[1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9,2.0,2.1,2.2,2.3,2.4,2.5,2.6,2.7,2.8,2.9,3.0]  yn= [-1.83333282942593,-1.71428516984133,-1.62499950017127,-1.55555511105261,  -1.49999960571033,-1.45454510284195,-1.41666635053680,-1.38461509824095,  -1.35714259583368,-1.33333309332710,-1.31249977826594,-1.29411744113621,  -1.27777758565039,-1.26315771473711,-1.24999983073609,-1.23809507839538,  -1.22727257614238,-1.2173911609365,-1.20833319690865,-1.19999986992714]  y=[ -1.83333333333333,-1.71428571428571,-1.62500000000000,-1.55555555555556,  -1.50000000000000,-1.45454545454545,-1.41666666666667,-1.38461538461538,  -1.35714285714286,-1.33333333333333,-1.31250000000000,-1.29411764705882,  -1.27777777777778,-1.26315789473684,-1.25000000000000,-1.23809523809524,  -1.22727272727273,-1.21739130434783,-1.20833333333333,-1.20000000000000]  分析：对实验2，N越大越精确。因为在相等区间间隔内，N越大，意味着划分的段越多，每一段的间隔越短，线性逼近时产生的偏差越小，逐渐累计的误差也会减小，最终结果就越精确。  问题3  （1）  准确解    N=5:  xn=[0.2,0.4,0.6,0.8,1.0]  yn= [1.760000000000000,8.813333333333338,43.680000000000035,217.2933333333336,1084.320000000002]  y= [0.046105212962911,0.160111820875968,0.360002048070785,0.640000037511725,1.000000000687051]  N=10:  xn=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]  yn= [0.122777777777778,0.079259259259259,0.104753086419753,0.166584362139918,0.253861454046639,0.362953818015546,0.492651272671849,0.642550424223950,0.812516808074650,1.002505602691550]  y= [0.055111761078871,0.046105212962911,0.090826250725555,0.160111820875968,0.250015133309921,0.360002048070784,0.490000277176240,0.640000037511725,0.810000005076660,1.000000000687051]  N=20:  xn=[0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,0.85,0.90,0.95,1.00]  yn=[0.127552083333333,0.0569466145833333,0.0401570638020833,0.0466734822591146,0.0650546391805013,0.0910100730260213,0.122930860718091,0.160213656102618,0.202632204371815,0.250101659972764,0.302590205823120,0.360085910517003,0.422584299777210,0.490083695749787,0.562583469239504,0.640083384298147,0.722583352445139,0.810083340500261,0.902583336020931,1.00008333434118]  y=[0.125126480390481,0.0551117610788709,0.0390956894559547,0.0461052129629114,0.0647459823330285,0.0908262507255555,0.122803960655185,0.160111820875967,0.202541136601362,0.250015133309921,0.302505567233597,0.360002048070784,0.422500753443136,0.490000277176240,0.562500101967440,0.640000037511725,0.722500013799793,0.810000005076660,0.902500001867599,1.00000000068705]  （2）    准确解    N=5:  xn=[0.2,0.4,0.6,0.8,1.0]  yn= [5.197338106220029,25.376170704380762,125.4868152611297,625.3120955171343,3123.795150947159]  y= [0.216984969683795,0.389753804936553,0.564648617607389,0.717356203434698,0.841470986869050]  N=10:  xn=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]  yn= [0.433138996497194,0.309660468004797,0.332324666705135,0.401413971263983,0.483074341470546,0.565435279659871,0.643989004482751,0.716722347060589,0.782499151201269,0.840525720595564]  y= [0.235168699883441,0.216984969683795,0.297998958838006,0.389753804936553,0.479470938533966,0.564648617607389,0.644218518766410,0.717356203434697,0.783326924857463,0.841470986869050]  N=20:  xn=[0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,0.85,0.90,0.95,1.00]  yn=[0.424978518601945,0.240456222130599,0.202168439043111,0.218438663412305,0.254811651108751,0.298291022215189,0.343928551050959,0.389795336350344,0.435096173332859,0.479462622859192,0.522688087921477,0.564628638372323,0.605165986304276,0.644193762568869,0.681612525466934,0.717328037859623,0.751250763421640,0.783295813201471,0.813383053836822,0.841437268860268]  y=[0.417858610442121,0.235168699883441,0.199225200841463,0.216984969683795,0.254141906253608,0.297998958838006,0.343809689421006,0.389753804936553,0.435088943915317,0.479470938533965,0.522703930631449,0.564648617607389,0.605188666065447,0.644218518766410,0.681639065925655,0.717356203434698,0.751280446539670,0.783326924857463,0.813415510392170,0.841470986869050]  （3）    准确解    N=5:  xn=[0.2,0.4,0.6,0.8,1.0]  yn= [0.298646212750134,0.927219870027348,2.835477338896382,10.710885330937337,47.941446381632670]  y= [0.242655268594923,0.580943900770567,1.028845666272092,1.596505340600251,2.287355287178843]  N=10:  xn=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]  yn= [0.112055109130374,0.245116514424435,0.401778096678299,0.584096956579228,0.793822052967138,1.032418305342644,1.301014988350537,1.600321012017492,1.930521033784067,2.291156923060078]  y= [0.110332988730204,0.242655268594923,0.398910553778490,0.580943900770567,0.790439083213615,1.028845666272092,1.297295111875269,1.596505340600251,1.926673303972716,2.287355287178842]  N=20:  xn=[0.05,0.10,0.15,0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.75,0.80,0.85,0.90,0.95,1.00]  yn=[0.0525950399557424,0.110408986281839,0.173709390516527,0.242749000926038,0.317771691558205,0.399013552467328,0.486702069624884,0.581054489098231,0.682275772497343,0.790556293022089,0.906069325028779,1.02896834412956,1.15938414166495,1.29742175278506,1.44315719602998,1.59663402222759,1.75785967098403,1.92680163375363,2.10338342334123,2.28748035067471]  y=[0.0525416560751489,0.110332988730204,0.173622339477193,0.242655268594923,0.317672971871986,0.398910553778490,0.486595151043810,0.580943900770567,0.682161747414882,0.790439083213615,0.905949216944811,1.02884566627209,1.15925926935155,1.29729511187527,1.44302926629324,1.59650534060025,1.75773083480017,1.92667330397272,2.10325632777116,2.28735528717884]  分析：对实验3，N较小时，数值解和解析解的偏差将会很大。因为四阶龙格库塔方法用线性算法来近似求解数值解，而实验3中的函数本身就不是线性的，线性逼近会有误差产生，再加上N较小，意味着划分后的每一段间隔都将较大，逼近效果不理想，误差大。 |

**实验报告三**

|  |
| --- |
| 题目（摘要）  利用牛顿迭代法求的根  输 入：初值，精度，最大迭代次数  输 出：方程根的近似值或计算失败标志  前 言（目的和意义）  实验目的：利用牛顿迭代法求的根  实验意义：熟悉牛顿迭代法，学会用牛顿迭代法求解方程根的近似值，利用计算机程序解决现实中的计算问题。  数学原理  求非线性方程的根，牛顿迭代法计算公式      一般地，牛顿迭代法具有局部收敛性，为保证迭代收敛，要求，对充分小的，。如果，，，那么，对充分小的，当时，由牛顿迭代法计算出的收敛于，且收敛速度是2阶的；如果，，，那么，对充分小的，当时，由牛顿迭代法计算出的收敛于，且收敛速度是1阶的。  程序设计流程  程序流程：  1 置  2 当时，做2.1—2.4  2.1 置,  如果 ，输出；停机  如果 ，输出失败标志；停机  2.2 置  2.3 置  如果 ，输出；停机  2.4 置 ，  3 输出失败标志  4 停机  用Matlab实现的具体代码如下：  function [ x ] = Newton( alpha,epsilon1,epsilon2,N )  %Newton 牛顿迭代法求解方程根  %alpha为初值，epsilon1和epsilon2为精度，N为最大迭代次数  x0=alpha；  for n=1:N  F=f(x0);  DF=df(x0);  if abs(F)<epsilon1  x=x0;  return;  end  if abs(DF)<epsilon2  x='fail';  return;  end  x1=x0-F/DF;  Tol=abs(x1-x0);  if abs(Tol)<epsilon1  x=x1;  return;  end  x0=x1;  end  x='fail';  return;  end  实验结果、结论与讨论  问题1：（1），，，，  由迭代得x=0.739085178106010  精确解x= 0.73908513321516064165531208767387  （2），，，，  由迭代得x= 0.588532742847979  精确解x= 0.5885327439818610774324520457029  讨论：确定初值时，需要在f(x)=0的根附近选择一点作为初始迭代值。在实际问题中，为了方便计算，可以考虑二分法等方法初步确定根的近似值，作为初值代入。  问题2：（1），，，，  由迭代得x= 0.567143165034862  精确解x= 0.567143290409784  （2），，，，  由迭代得x= 0.566605704128158  精确解x= 0.567143290409784  讨论：（2）中有重根，（1）和（2）的答案本应相同。但是（2）中重根的出现，使得第二个方程的求解精度小于（1）中的求解精度，导致最终迭代结束时，（2）中所得结果精度变低了，与（1）中结果不同。  问题3：  递推公式的代码模板如下：（以勒让德多项式为例）  function [ p ] = ditui( n )  %ditui 递推求表达式  syms x;  if n==0  p=1;  elseif n==1  p=x;  else  p=((2\*n-1)\*x\*ditui(n-1)-(n-1)\*ditui(n-2))/n;  end  end  （1）由下面的递推公式可以生成勒让德（Legendre）多项式    ①试确定和  ②确定，求得所有零点，精度  0.9324695142，0.6612093865，0.2386191861  ①P2(x)= (3\*x^2)/2 - 1/2  P3(x)= (5\*x\*((3\*x^2)/2 - 1/2))/3 - (2\*x)/3  P4(x)= 3/8 - (9\*x^2)/8 - (7\*x\*((2\*x)/3 - (5\*x\*((3\*x^2)/2 - 1/2))/3))/4  P5(x)= (8\*x)/15 - (4\*x\*((3\*x^2)/2 - 1/2))/3 - (9\*x\*((7\*x\*((2\*x)/3 - (5\*x\*((3\*x^2)/2 - 1/2))/3))/4 + (9\*x^2)/8 - 3/8))/5  ② P6(x)=(35\*x\*((2\*x)/3 - (5\*x\*((3\*x^2)/2 - 1/2))/3))/24 - (11\*x\*((4\*x\*((3\*x^2)/2 - 1/2))/3 - (8\*x)/15 + (9\*x\*((7\*x\*((2\*x)/3 - (5\*x\*((3\*x^2)/2 - 1/2))/3))/4 + (9\*x^2)/8 - 3/8))/5))/6 + (15\*x^2)/16 - 5/16  迭代结果如下：  0.932469514332961  -0.932469514332961  0.661209586595914  - 0.661209586595914  0.238619535037660  - 0.238619535037660  （2）由下面的递推公式可以生成切比雪夫勒让德（Chebyshev）多项式    ①试确定和  ②确定，求得所有零点，精度  ，  ①P2(x)= 2\*x^2 - 1  P3(x)= 2\*x\*(2\*x^2 - 1) - x  P4(x)= 1 - 2\*x^2 - 2\*x\*(x - 2\*x\*(2\*x^2 - 1))  P5(x)= x - 2\*x\*(2\*x^2 - 1) - 2\*x\*(2\*x\*(x - 2\*x\*(2\*x^2 - 1)) + 2\*x^2 - 1)  ② P6(x)= 2\*x\*(x - 2\*x\*(2\*x^2 - 1)) - 2\*x\*(2\*x\*(2\*x^2 - 1) - x + 2\*x\*(2\*x\*(x - 2\*x\*(2\*x^2 - 1)) + 2\*x^2 - 1)) + 2\*x^2 - 1  迭代结果如下：  0.258819045102525  -0.258819045102525  0.965925826291030  -0.965925826291030  0.707106782535483  -0.707106782535483  （3）由下面的递推公式可以生成拉盖尔（Laguerre）多项式    ①试确定和  ②求得所有零点，精度  0.2635603197，1.4134030591，3.5964257710，7.0858100059，12.6408008443  ①P2(x)= (x - 1)\*(x - 3) - 1  P3(x)= 4\*x - ((x - 1)\*(x - 3) - 1)\*(x - 5) - 4  P4(x)= (x - 7)\*(((x - 1)\*(x - 3) - 1)\*(x - 5) - 4\*x + 4) - 9\*(x - 1)\*(x - 3) + 9  P5(x)= 16\*((x - 1)\*(x - 3) - 1)\*(x - 5) - 64\*x - (x - 9)\*((x - 7)\*(((x - 1)\*(x - 3) - 1)\*(x - 5) - 4\*x + 4) - 9\*(x - 1)\*(x - 3) + 9) + 64  ② 迭代结果如下：  0.263560320250284  1.413403065338242  3.596434404829551  7.085809999942143（取初值7.08581）  12.640800003947867（取初值12.6408）  （4）由下面的递推公式可以生成埃尔米特（Hermite）多项式    ①试确定和  ②确定，求得所有零点，精度  2.3506049737，1.3358490740，0.4360774119  ①P2(x)= 4\*x^2 - 2  P3(x)= 2\*x\*(4\*x^2 - 2) - 8\*x  P4(x)= 12 - 24\*x^2 - 2\*x\*(8\*x - 2\*x\*(4\*x^2 - 2))  P5(x)= 64\*x - 16\*x\*(4\*x^2 - 2) - 2\*x\*(2\*x\*(8\*x - 2\*x\*(4\*x^2 - 2)) + 24\*x^2 - 12)  ② P6(x)= 20\*x\*(8\*x - 2\*x\*(4\*x^2 - 2)) - 2\*x\*(16\*x\*(4\*x^2 - 2) - 64\*x + 2\*x\*(2\*x\*(8\*x - 2\*x\*(4\*x^2 - 2)) + 24\*x^2 - 12)) + 240\*x^2 - 120  迭代结果如下：  2.350603533295554  -2.350603533295554  1.335848607526927  -1.335848607526927  0.436077377907361（取初值0.4360774）  -0.436077377907361（取初值-0.4360774）  讨论：使用牛顿迭代法求方程根的近似解时，可能会出现（3）和（4）中这样的情况，需要初值极其靠近准确值，才能使迭代的结果收敛到准确值，这是由于在方程根的极小领域内迭代函数的一阶导数都不满足收敛条件造成的。 |