**重 庆 大 学**

**学 生 实 验 报 告**

**实验课程名称 数学实验**

**开课实验室 DS1407**

**学 院 计算机学院 年级 2022级 专业班 06班**

**学 生 姓 名 楼洋 学 号 20221627**

**开 课 时 间 2023 至 2024 学年第 二 学期**

|  |  |
| --- | --- |
| **总 成 绩** |  |
| **教师签名** |  |

**数 学 与 统 计 学 院 制**

**开课学院、实验室： 计算机学院 实验时间 ： 2024 年 3 月 16 日**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **课程**  **名称** | **数学实验** | **实验项目**  **名 称** | **迭代、方程模型及其求解算法** | **实验项目类型** | | | | |
| **验证** | **演示** | **综合** | **设计** | **其他** |
| **指导**  **教师** | **龚劬** | **成 绩** |  |  |  |  |  |  |
| 实验目的  [1] 复习求解方程及方程组的基本原理和方法；  [2] 掌握迭代算法；  [3] 熟悉MATLAB软件编程环境；掌握MATLAB编程语句(特别是循环、条件、控制等语句)；  [4] 通过范例展现求解实际问题的初步建模过程；  通过该实验的学习，复习和归纳方程求解或方程组求解的各种数值解法（简单迭代法、二分法、牛顿法、割线法等），初步了解数学建模过程。这对于学生深入理解数学概念，掌握数学的思维方法，熟悉处理大量的工程计算问题的方法具有十分重要的意义。  基础实验1  问题重述  使用**fsolve**计算方程组的解时，为验证初值是否对解有影响，采用随机产生的100组随机数作为初始值，依次进行求解。  实验过程  F=@(x) [x(1)+x(2)^2-13;  log(2\*x(1)+x(2))-x(1)^x(2)+2];  x0=randi(20,100,2);  res=zeros(100,2);  for i=1:100  [x,fval,exitflag, output]=fsolve(F,[x0(i,1),x0(i,2)]);  if exitflag>0  res(i,1)=x(1);  res(i,2)=x(2);  end    end  实验结果及分析  图一:实验一运行结果  分析：  初值会对实验结果产生影响，因为有些方程组的解不止一个，当初值不同得到的解也会不同。  基础实验2  问题重述  （1）将方程*x*5 +5*x*3- 2*x* + 1 = 0 改写成各种等价的形式进行迭代，观察迭代是否给出收敛，并给出解释。  （2）编写用二分法求方程根的函数M文件，用该函数文件求（1）中方程的根，给出运行时间（可用MATLAB命令**tic**,**toc**）。  （3）使用牛顿法编程（function [x, fval]=niudun(fun, x0, er)）求方程的根（用**diff**求导函数时要将函数的自变量定义为符号变量，如：syms x，df=diff(fun,x); 求导函数在某点的函数值时需要用**subs**），用该函数文件求（1）中方程的根，给出运行时间（可用MATLAB命令**tic**,**toc**）  实验过程  x=-1;b=-1;  for k=1:20  a=2/x^3-5/x-1/x^4;  b=(1/2)\*x^5+(5/2)\*x^3+1/2;  x=(-4\*x^5-10\*x^3+1)/(2-5\*x^4-15\*x^2)  c=[a,b,x]  end  function root = bisection(f, a, b, tol)  tic;  fa = f(a);  fb = f(b);  if fa \* fb > 0  error('Function does not change sign on the interval.');  end  while (b - a) / 2 > tol  c = (a + b) / 2;  fc = f(c);  if fc == 0  break;  end  if fa \* fc < 0  b = c;  fb = fc;  else  a = c;  fa = fc;  end  end  root = (a + b) / 2;  toc;  end  function [x, fval] = niudun(fun, x0, er)  tic;    df = diff(fun, x); %导函数  fval = subs(fun, x, x0); %当前函数值  while abs(fval) > er  dfval = subs(df, x, x0);%df  x0 = x0 - fval / dfval;  fval = subs(fun, x, x0);  end  x = x0;  toc;  end  f = @(x) x^5 + 5\*x^3 - 2\*x + 1;  root = bisection(f, -2, 2, 1e-6);  syms x;  fun = x^5 + 5\*x^3 - 2\*x + 1;  [x, fval] = niudun(fun, 0, 1e-6);  实验结果及分析  实验结果：    图二：实验二运行结果  分析：  (1)将y=0变形得,这是y1,以及，这是y2,利用加速迭代使y1变形后得到,这是y3，通过迭代可以得到解为x=-0.7685,且y1、y2、y3均收敛，但y3收敛最快  (2)(3)分别通过二分法和牛顿法编写了求根的方程，根据结果，二分法所用时间比牛顿法快，且两者得到解为-0.7685  基础实验3  问题重述   1. 求方程ex-3x=0, 在[-1，1]上的近似解，使用**fzero，fsolve**分别进行求解。   （b）判定方程 x7+2x5+3x3+5x+7=0 有几个实根，并使用**roots，fzero，fsolve**分别进行求解。  （c）求解下列方程组    直接使用MATLAB命令：solve()和fsolve()对方程组求解。  （d）设非线性方程组为    其中已知，随机产生数据后，用fsolve解这个方程组。  可以不用全局变量，用匿名函数即可。  实验过程  res1=fsolve(@(x) exp(x)-3\*x,-1)  res2=fzero(@(x) exp(x)-3\*x,[-1,1])  fprintf("使用fsolve的结果为%f,使用fzero的结果为%f",res1,res2)  res3=roots([1,0,2,0,3,0,5,7])  res3=res3(res3==real(res3))  res4=fzero(@(x) x^7+2\*x^5+3\*x^3+5\*x+7,[-2,2]);  res5=fsolve(@(x) x.^7+2\*x.^5+3\*x.^3+5\*x+7,0);  fprintf("使用roots的结果为%f,使用fzero的结果为%f,使用fsolve的结果为%f",res3,res4,res5)  clear  syms x1 x2  eqns = [2\*x1 - x2 == exp(-x1), -x1 + 2\*x2 == exp(-x2)];  [s1,s2] = vpasolve(eqns, [x1, x2]);  res6=[s1,s2]  clear  syms x1 x2 x3  eqn=[x1^2-5\*x2^2+7\*x3^2==-12;3\*x1\*x2+x1\*x3-11\*x1==0;2\*x2\*x3+40\*x1==0]  [s1,s2,s3]=solve(eqn,[x1 x2 x3]);  res7=[s1,s2,s3]  F=@(x) [2\*x(1)-x(2)-exp(-x(1));-x(1)+2\*x(2)-exp(-x(2))];  res8=fsolve(F,[2,2])  F=@(x) [x(1)^2-5\*x(2)^2+7\*x(3)^2+12;3\*x(1)\*x(2)+x(1)\*x(3)-11\*x(1);2\*x(2)\*x(3)+40\*x(1)];  res9=fsolve(F,[2,2,2])  function F = myEquations(x, a, b, c)  F = zeros(10, 1);  for k = 1:10  sum\_xi = sum(x);  for j = 1:10  F(k) = F(k) + x(j) .\* (c(k, j) + log(abs(a(k) .\* x(j)) ./ sum\_xi));  end  F(k) = F(k) - b(k);  end  end  a=randi([20],1,10);  b=randi([20],1,10);  c=randi([20],10,10);  x0 = ones(10, 1);  options = optimoptions('fsolve', 'Display', 'iter');  [x, fval, exitflag, output] = fsolve(@(x) myEquations(x, a, b, c), x0, options);  disp(x);  实验结果及分析  实验结果：    图三：实验三运行结果  分析：   1. fzero,fsolve得出的解相同，都为0.6191 2. 方程有一个解，且三种方法都可以都到解为-0.834114 3. 用solve得到的解比较复杂，用fsolve可以直接得到解 4. 由于方程系数随机产生，每次运行结果都不一样   探究实验1  问题重述  考虑线性方程组A*x*=b,其中   1. 作出两个方程的图形，计算A的条件数。 2. 计算Ax=b的真解*x*true,计算如下1000个相近的方程组的解   ,i=1,2,…,1000  其中的元素是独立的正态分布变量，均值为0,标准差τ=0.0001.（你可以靠设置DeltaA=tau\*randn(2,2)来产生这些例子。作出1000个点,这个图展示了把作为A的近似的向前误差。在新的图形窗口作出1000个残差(向后误差).   1. 对线性方程组Ax=b重复(a),(b)的做法。其中 2. 讨论你的结果。为什么两个问题的向前误差图会如此的不同。向前误差图中你看到的有关条件数怎么样？向后误差图告诉了我们什么？   实验过程  clear  delta = 0.002;  A = [delta, 1; 1, 1];  b = [1; 0];  % 定义方程  f1 = @(x) (b(1) - A(1,1)\*x)/A(1,2);  f2 = @(x) (b(2) - A(2,1)\*x)/A(2,2);  % 绘制方程  fplot(f1);  hold on;  fplot(f2);  hold off;  % 计算交点  x\_intersect = fzero(@(x) f1(x) - f2(x),0);  y\_intersect = f1(x\_intersect);  % 计算Ax=b的解  x\_true = A\b;  % 初始化解的数组  solutions = zeros(2,1000);  % 观察不同方程组的解  figure  tau = 0.0001;  for i = 1:1000  DeltaA = tau\*randn(2,2);  A\_new = A + DeltaA;  solutions(:,i) = A\_new\b;    end  % 计算e(i)  e = solutions - x\_true;  plot([e(1,:),e(2,:)],'o');  figure;  r=b-A\*solutions;  plot([r(1,:),r(2,:)],'o');  A = [1+delta, -1; -1, 1+delta];  b = [2; -2];  x\_true = A\b;  figure  tau = 0.0001;  for i = 1:1000  DeltaA = tau\*randn(2,2);  A\_new = A + DeltaA;  solutions(:,i) = A\_new\b;    end  % 计算e(i)  e = solutions - x\_true;  plot([e(1,:),e(2,:)],'or');  figure;  r=b-A\*solutions;  plot([r(1,:),r(2,:)],'or');  实验结果及分析    图4：（a）两个方程的图形  图四：（a）两个方程的图形    图5：（b）向前误差和向后误差图    图6：（c）新的线性方程组的向前和向后误差图  分析：   1. 做出两个方程的图形如图4，这两个方程的交点即是方程组的解 2. 通过计算1000个相近的方程组的解，画出A的近似向前误差和向后误差的图 3. 对新的线性方程组进行b的操作，画出新的向前和向后误差图 4. 由图可知（b）的向前和向后误差更小，而（c）的向前和向后误差更大，说明了在讨论线性方程组的解时，条件数是一个条件数是一个重要的概念，它衡量了输入数据的微小变化对解的影响。条件数越大，系统对数据扰动的敏感度越高，即系统是病态的；条件数越小，系统对数据扰动的敏感度越低，即系统是良态的。   向前误差和向后误差是评估数值解质量的两种方法。向前误差关注的是数值解与真实解之间的差异，而向后误差关注的是数值解代入原方程后的残差。  对于向前误差图的不同，这可能是由于两个问题的条件数不同。如果一个问题的条件数远大于另一个问题的条件数，那么即使是相同的数据扰动，也会导致更大的向前误差。这表明，即使两个问题在数学上是等价的，它们的数值解的稳定性也可能大不相同。  向后误差图提供了关于数值解稳定性的信息。如果向后误差较小，这意味着数值解即使不是精确的，也是可接受的，因为它们在原方程中产生的残差很小。相反，如果向后误差较大，即使数值解看起来接近真实解，也可能不是一个好的解，因为它们在原方程中产生了较大的残差。  总的来说，条件数提供了一个量化指标来评估问题的病态程度，而向前误差和向后误差则提供了解的质量评估。在实际应用中，理解这些概念对于解决数值问题至关重要  教师签名  年 月 日 | | | | | | | | |