画着卡通人物

描述已自动生成

课 程 设 计

数学建模实验

设计题目

傅益忠

学生姓名

2022210826

学 号

信息与计算科学22-2班

专业班级

苏敏

指导教师

**2024** 年 6 月 5 日

合肥工业大学课程设计任务书

|  |  |  |  |
| --- | --- | --- | --- |
| **设计题目** | **数学建模实验** | **成绩** |  |
| **课**  **程**  **设**  **计**  **主**  **要**  **内**  **容** | **课程设计题目一览：**  引言   * 1. 最优捕鱼策略   2. 黄河小浪底调水调沙问题   3. 国土面积问题   4. 轮船排队系统   5. （MCM-88）铁路平板车问题  1. 投资的收益与风险 2. 设备更新问题    1. 销售点问题    2. 阿波罗运动轨迹问题   引言：  所有代码和数据表格均被本人开源至网站  <https://github.com/xiao-en-5970/MATLAB-homework>  如果有复现代码结果的需要，可从网站中获取源代码并于matlab中运行   * 1. **最优捕鱼策略**：     **假设和初始条件**  设最初鱼的数量为 ，其中初始所有年龄段的鱼数量均为 。  捕捞强度系数：   * 对3龄鱼的捕捞强度系数为 * 对4龄鱼的捕捞强度系数为   鱼的自然死亡率为 。  不同龄鱼的重量：  繁殖数量：  卵的存活率：  迭代次数  设收益 P 初始化为0向量，长度为100：  **动态模型**  设第4龄鱼的捕捞强度为 矩阵定义为：    **计算过程**  对于 到 100：   1. 初始化状态向量 2. 进行 N 次迭代： 并更新第1龄鱼的数量： 3. 计算收益 P(x)：   **结果**  为了保持鱼数量不变，鱼的初始投入数量分别为 。 最大收益为：  通过上述模型和计算，我们可以得到在不同捕捞强度 x 下的最大收益，并确保鱼群数量的动态平衡。    **代码**  %% 对数据的假设  % 假设最开始只有1龄鱼  clc,clear;  % 初始鱼数量->这个不重要，总会收敛的（确信）  S=[1000000000;1000000000;1000000000;1000000000];  % 捕捞强度系数  power\_3 = 0.42;  power\_4 = 1;  % 死亡率  death\_rate = 0.8;  % 鱼的重量  kg = [5.07,11,55,17.86,22.99];  % 繁殖数量  breed\_4 = 1.109\*10^5;  breed\_3 = breed\_4/2;  % 卵的存活率  survive\_rate = 1.22\*10^11;  % 设置精度  format long  % 迭代次数  N=1000;  % P显然是收益，也就是鱼重量  P = zeros(100,1);  %% 计算  % 设4龄捕鱼强度为x  clc;  for x = 1:100  A = [0,0,breed\_3,breed\_4;  1-death\_rate,0,0,0;  0,1-death\_rate,0,0;  0,0,(1-death\_rate)\*(1-power\_3\*x/100),0];  for i = 1:N  S = A\*S;  S(1) = survive\_rate\*S(1)/(S(1)+survive\_rate);  P(x) = P(x) + S(3)\*kg(3)\*power\_3\*x/100+S(4)\*kg(4)\*power\_4\*x/100;  end      end  fprintf("为了保持鱼数量不变，鱼的初始投入数量应该分别为：");  S  fprintf("最大收益为：");  max(P)  **1.2黄河小浪底调水调沙问题**      **单位时间和排沙量关系如图**    **排沙量和水流量关系如图**    **代码**  %% 数据处理  clc,clear;  data = readtable("data\_1\_2.csv");  %time代表从6.29的8点，每过12h作为一个单位时间的值  time = table2array(data(:,"Var1"));  % 水流量  water\_stream=table2array(data(:,"Var2"));  % 沙含量  sand\_content = table2array(data(:,"Var3"));  %%  % 绘图  % 用原始数据作图  plot(time,water\_stream,"r-",LineWidth=3);  figure  plot(time,sand\_content,"b-",LineWidth=3);  %% （1）给出估计任意时刻的排沙量及总排沙量的方法；  % xx用于预测从6.29的8点之后的每一小时的沙流量yy，用三次样条插值（我觉得拉格朗日不行）  xx = (1:0.125:24)';  % 沙流量等于水流量乘含水量  sand\_stream = sand\_content.\*water\_stream;  yy = spline(time,sand\_stream,xx);  plot(xx,yy,"b-",LineWidth=3)  legend("排沙量 kg/s");  xlabel("单位时间 h");  ylabel("排沙量 kg/s");  % 而显然总排沙量等于每小时排沙量乘3600，然后累加  total\_sand\_stream = zeros(size(xx));  total\_sand\_stream(1) = sand\_stream(1)\*3600;  for i = 2:size(total\_sand\_stream)  total\_sand\_stream(i) = total\_sand\_stream(i-1)+yy(i)\*3600;  end  figure  plot(xx,total\_sand\_stream,"r-",LineWidth=3);  legend("总排沙量 kg");  xlabel("单位时间 h");  ylabel("总排沙量 kg");  %% （2）确定排沙量与水流量的关系。  sand\_stream = water\_stream.\*sand\_content;  plot(water\_stream,sand\_stream);  % 排沙量等于水流量乘含沙量  **数据表格**    **1.3国土面积问题**      **国土范围如图所示**    **最终计算得出**    **代码**  %% 数据处理  clc,clear;  data = readtable("data\_1\_3.csv");  x = table2array(data(:,"x"));  y1= table2array(data(:,"y1"));  y2= table2array(data(:,"y2"));  rate=40/18;  real\_square = 41288;  %% 初始数据绘图  plot(x,y1,"r-",LineWidth=2);  hold on  plot(x,y2,"r-",LineWidth=2);  hold off  %% 对数据进行三次样条插值  clc;  xx = 7:0.1:158;  yy1=spline(x,y1,xx);  yy2=spline(x,y2,xx);  % 插值之后的数值绘图  plot(xx,yy1,"r-",LineWidth=2);  hold on  plot(xx,yy2,"b-",LineWidth=2);  legend("y1","y2");  xlabel("x/mm");  ylabel("y/mm");  total\_square = sum((yy2-yy1)\*0.1\*rate\*rate);  fprintf("计算面积为：%.6f\n",round(total\_square,6));  diff = (total\_square-real\_square)/real\_square\*100;  fprintf("与真实面积的误差为：百分之%.3f\n",round(diff,3));  **数据表格**    **2.1轮船排队问题**          **计算结果**    **代码**  %% 数据处理  clc,clear;  % 读取csv表  data = readtable("data\_2\_1.csv");  % 间隔时间，例如15-24，这边我取均值20  between\_time = data.betweenTime;  % 1200艘经验数据中的对应间隔时间的数量  between\_num = data.betweenNum;  % 总共数量为1200  total\_num = data.totalNum(1);  % 装箱时间，在45-49中取平均为47  unloading\_time = data.unloadingTime(1:9);  %对应经验数据中的数量  unloading\_num = data.unloadingNum(1:9);  % 对0——1200的每一个数进行数值对时间的映射，方便随机值取值对应  hash\_between = zeros(total\_num,1);  hash\_unloading = zeros(total\_num,1);  p =1;  for i = 1:13  for j = 1:between\_num(i)  hash\_between(p) = between\_time(i);  p = p +1;  end  end  p =1;  for i = 1:9  for j = 1:unloading\_num(i)  hash\_unloading(p) = unloading\_time(i);  p = p +1;  end  end  % 先随机出1000条船进港口  N=1000;  % 做出一个随机的间隔时间和装载时间表  between\_rand\_time = hash\_between(round((1200-1)\*rand(N,1),0)+1);  % 第一艘船不需要等待  between\_rand\_time(1) = 0;  unloading\_rand\_time = hash\_unloading(round((1200-1)\*rand(N,1),0)+1);  % 船到达时间  arrive\_time = zeros(N,1);  for i = 2:N  arrive\_time(i) = arrive\_time(i-1)+between\_rand\_time(i);  end  % 等待时间  waiting\_time = zeros(N,1);  % 在港时间  living\_time = zeros(N,1);  % 最新来的船序号  new\_ship = 1;  % 正在服务的船序号  cur\_ship = 1;  % 正在服务的船的进度  cur\_proc = 0;  % 标准时间  cur\_time = 0;  %平均等待时间  avg\_wait\_time = 0;  %平均等待在港时间  avg\_live\_time = 0;  %% 平均等待时间  % 等待时间  waiting\_time = zeros(N,1);  % 最新来的船序号  new\_ship = 1;  % 正在服务的船序号  cur\_ship = 1;  % 正在服务的船的进度  cur\_proc = 0;  % 标准时间  cur\_time = 0;  %将时间离散化以每分钟为单位  % 当装载还没结束的时候  clc;  while cur\_ship<=N  %如果现在正在装载的船还没完  if cur\_ship ~= 0  if cur\_proc<unloading\_rand\_time(cur\_ship)  %则继续装  cur\_proc = cur\_proc+1;  else  % 如果不空闲,下一艘船继续  if cur\_time<new\_ship  cur\_ship = cur\_ship+1;  else  cur\_ship = 0;  end  cur\_proc = 0;  end  end  % 如果有排队发生，则排队时间+1  if cur\_ship~=0 && cur\_ship<new\_ship  waiting\_time(cur\_ship+1:new\_ship) = waiting\_time(cur\_ship+1:new\_ship)+1;  end  % 如果空闲且最新的船为最后一个，则表明接完了，break掉  if cur\_ship == 0 && new\_ship == N  break;  end  % 检测是否该时刻有新船来  if new\_ship<N && cur\_time>=arrive\_time(new\_ship+1)  if cur\_ship == 0  cur\_ship = new\_ship;  end  new\_ship = new\_ship+1;  end  %无论如何，时间都将继续流逝  cur\_time = cur\_time+1;  end  % 平均等待时间  avg\_wait\_time = sum(waiting\_time)/N  % 平均在港口时间（其实就是等待时间加上装载时间）  avg\_live\_time = sum(waiting\_time+unloading\_rand\_time)/N  fprintf("平均等待时间为：%f\n",avg\_wait\_time);  fprintf("平均在港时间为：%f\n",avg\_live\_time);  **数据表格**    **2.2（MCM-88）铁路平板车问题**                          **代码**  %% 数据预处理  clc,clear;  data = readtable("data\_2\_2.csv");  width = data.width;  kilo = data.kilo;  num = data.num;  % 建立变量cij表示i车上cj的数量  % 则有c11,c12,c13,c14,c15,c16,c17,c21,c22,c23,c24,c25,c26,c27  % 以上十四个变量  % 对567的约束  C567 = 302.7;  % 车总质量  TOTAL\_KILO = 40000;  % 车总厚度  TOTAL\_WIDTH = 1020;  % 对变量的各种约束，包括质量超载，厚度超出，物品件数，C567的单独约束  A=[kilo',0,0,0,0,0,0,0;  0,0,0,0,0,0,0,kilo';  width',0,0,0,0,0,0,0;  0,0,0,0,0,0,0,width';  1,zeros(1,6),1,zeros(1,6);  0,1,zeros(1,5),0,1,zeros(1,5);  0,0,1,zeros(1,4),0,0,1,zeros(1,4);  zeros(1,3),1,zeros(1,3),zeros(1,3),1,zeros(1,3);  zeros(1,4),1,zeros(1,2),zeros(1,4),1,zeros(1,2);  zeros(1,5),1,zeros(1,1),zeros(1,5),1,zeros(1,1);  zeros(1,6),1,zeros(1,6),1;  zeros(1,4),1,1,1,zeros(1,7);  zeros(1,11),1,1,1];  % 约束的最大值  b = [TOTAL\_KILO;  TOTAL\_KILO;  TOTAL\_WIDTH;  TOTAL\_WIDTH;  num;  C567;  C567;  ];  % 期望求出物品的总厚度最大，则-width最小  f = [-width',-width'];  % 物品数量最小值为0  lb = [zeros(14,1)];  % 最大值不超过件数  ub = [num;num];  % intlinprog函数需要知道哪些变量只能为整数，以下是需要为整数的索引  intcon = 1:14;  %% 计算  % 核心，调用intlinprog函数求出约束条件下的f最小值2\*TOTAL\_WIDTH+value  [x,value,ef,op] = intlinprog(f,intcon,A,b,[],[],lb,ub);  smallest\_space = round(2\*TOTAL\_WIDTH+value,2);  fprintf("最小浪费空间为%f\n",smallest\_space);  **数据表格**    **3.1投资的收益与风险**    表格  描述已自动生成          最终求出图像      **风险越高，收益越大，但是在图中，风险a接近0.05的时候，风险增加，收益提高不明显**  **代码**  %% 数据预处理  clc,clear;  data = readtable("data\_3\_1.csv");  r = data.r;  q = data.q;  p = data.p;  u = data.u;  n = data.n;  r0 = data.r0;  N=15;  %% 计算​​​​  a=0;  hold on;  while a<0.2  c=[-0.05,-(data.r-data.p)'/100];  A=[zeros(N,1),diag((data.q/100)')];  b=a\*ones(N,1);  Aeq=[1,(data.p/100+1)'];  beq=1;  LB=zeros(N+1,1);  [x,Q]=linprog(c,A,b,Aeq,beq,LB);  Q=-Q;  plot(a,Q,'\*r');  a=a+0.001;  end  xlabel('a(风险)'),ylabel('Q(收益)')  fprintf("风险越高，收益越大，但是在图中，风险a接近0.05的时候，风险增加，收益提高不明显\n")  **数据表格**    **3.2设备更新问题**    **思路：**        **计算结果：**    **代码**  %% 数据预处理  clc;clear;  data = readtable("data\_3\_2.csv");  buy\_price = data.purchase\_price;  fix\_price = data.fix\_price;  %设Vij为第i年购入机器到第j年的花费  v = zeros(6);  for i = 1:5  for j = 1:6  if i<j  v(i,j) = buy\_price(i)+sum(fix\_price(1:j-i));  elseif i>j  v(i,j) = inf;  end  end  end  %% 计算v16的最短路  clc;  [price,pass]=dijkstra(v,1,6);  pass = fliplr(pass);  fprintf("最小花费为:%f\n最小路径为:",price);  pass  fprintf("表示从第一年用到第三年换新，然后一直用到第六年初，为最优解\n");  **数据表格**    **4.1销售点问题**    **思路：**                **结果：**    %% 数据预处理  % 邻接矩阵A  clc,clear;  A = [  0,20,Inf,Inf,15,Inf;  20,0,20,60,25,Inf;  0,20,0,30,18,Inf;  Inf,60,30,0,Inf,Inf;  15,25,18,Inf,0,15;  Inf,Inf,Inf,Inf,15,0;  ];  % 创建i到j的最短距离的矩阵F  F = zeros(6);  %% 计算  clc;  % 对每两个点跑一遍dijkstra求个最短路  for i = 1:6  for j = 1:6  if i==j  continue;  end  F(i,j)=dijkstra(A,i,j);  end  end  % 跑出最短距离矩阵之后，对每个点向周围的点找出最远距离的点  for i = 1:6  fprintf("V%f对其他点的最远距离为%f\n",i,max(F(i,:)));  end  fprintf("综上，V3到其他点的最远距离最小，V3建立销售点\n");  **4.2阿波罗运动轨迹问题**              **运算结果：**    **代码：**  %% 计算(数值）  clc,clear;  % 解方程  syms x(t) y(t) u lam r1 r2  u = 1/82.45;  lam = 1 - u;  x = y(1);  dx = y(2);  y\_pos = y(3);  dy = y(4);  r1 = sqrt((x + u)^2 + y\_pos^2);  r2 = sqrt((x + lam)^2 + y\_pos^2);  d2x = 2 \* dy + x - lam \* (x + u) / r1^3 - u \* (x - lam) / r2^3;  d2y = -2 \* dx + y\_pos - lam \* y\_pos / r1^3 - u \* y\_pos / r2^3;  [t, sol] = ode45(@(t, y) odefun(t, y, u, lam),[0 10],[1.2; 0; 0; -1.0494]);  x\_sol = sol(:, 1);  y\_sol = sol(:, 3);  % 可视化结果  figure;  plot(y\_sol, x\_sol, 'r',LineWidth=2);  title('轨迹');  % 定义ODE函数  function dydt = odefun(t, y, u, lam)  x = y(1);  dx = y(2);  y\_pos = y(3);  dy = y(4);    r1 = sqrt((x + u)^2 + y\_pos^2);  r2 = sqrt((x + lam)^2 + y\_pos^2);    d2x = 2 \* dy + x - lam \* (x + u) / r1^3 - u \* (x - lam) / r2^3;  d2y = -2 \* dx + y\_pos - lam \* y\_pos / r1^3 - u \* y\_pos / r2^3;    dydt = [dx; d2x; dy; d2y];  end | | |
| **指**  **导**  **教**  **师**  **评**  **语** | **建议：从学生的工作态度、工作量、设计（论文）的创造性、学术性、实用性及书面表达能力等方面给出评价。**  **签名： 2022年 月 日** | | |