



计算机与软件工程学院

上机报告

**（ 2021/2021 学年 第 1 学期 ）**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | **数值计算（C++）** | | | | | |
| 课程代码 | **190901319** | | | | | |
| 上机时间 | 2021 | 年 | 11 | 月 | 10 | 日 |
| 指导单位 | 物联网工程系 | | | | | |
| 任课教师 | 李显勇 | | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 学生姓名 | 李子涵 | | |
| 学 号 | 3120190971401 | | |
| 成 绩 |  | 年级专业 | 计算机科学与技术 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **实验名称** | **【实验2】拟合函数** | | | **实验地点** | 8-322 |
| **实验类型** | **验证** | **实验学时** | **2** | **实验日期** | 2021-11-10 |
| **实验目的和要求**   1. 了解**拟合函数**基本理论，编程实现该算法； 2. 了解**多项式拟合**基本理论，编程实现该算法； 3. 提交以下文档（用**学号+姓名**作为文件夹名，将所有内容放于该文件夹中）：   1）实验报告  2）程序代码 | | | | | |
| **实验环境(实验设备)**  windows XP及以上版本；PC；Matlab7.0及以上版本；其它高级语言 | | | | | |
| **实验原理及内容**  **一、实验原理**  根据**拟合函数，多项式拟合**相关知识和算法编程完成本实验  **二、实验内容**  已知1920年-1970年美国人口如下   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 年份 | 1920 | 1930 | 1940 | 1950 | 1960 | 1970 | | 人口（千人） | 105711 | 123203 | 131669 | 150697 | 179323 | 203212 |  1. 用二次多项式拟合数据，并用此估计1965年和2002年的人口数； 2. 观察数据特点，选择合适的拟合函数对数据进行拟合，并用此估计估计1965年和2012年的人口数。   **三、实验过程（可以文字说明+运行结果截图）**  准备函数代码：（实现解方程与预测人口数的部分）  '''  文件名：西华大学数值计算C++实验报告——实验二  当前版本：1.0  完成作者：李子涵  学号：3120190971401  完成日期：2021.11.10  实验环境：Windows10;Python 3.7.10  注：所有系数列表均为升幂排列  '''  import numpy as np  from numpy.core.fromnumeric import transpose  import math  def solve(coefficient):      """      解三元（二元）一次方程组      1x + 2y + 3z = 4      5x + 7y + 6z = 7      9x + 3y + 6z = 9      示例：      coefficient=[[1, 2, 3, 4], [5, 7, 6, 7], [9, 3, 6, 9]]      """      #coe=[[5,-0.5,1.875,9.32],[-0.5,1.875,-0.6875,2.2925],[1.875,-0.6875,1.3828,2.7138]]#系数列表      #coe=[2,-1,1,10,3,2,-1,16,1,6,-1,28]        #分别表示每一个式子的系数      flag=len(coefficient)#二元or三元标志      first=coefficient[0]      second=coefficient[1]      if(flag==3):          last=coefficient[2]      def  elimination(first\_coe,second\_coe):          """          消去一项          输入参数：第一个式子和第二个式子的系数          返回参数：消去第一个未知数后的一个式子的系数          """          new=[]          #若第二个式子无此未知数则直接用第二个式子的系数          if(second\_coe[0]==0):              second\_coe.pop(0)              new=second\_coe          #若有就先消去未知数          else:              cnt=0              for i in second\_coe:                  new.append(first\_coe[cnt]-i\*first\_coe[0]/second\_coe[0])                  cnt+=1              new.pop(0)#去掉0项          return new      #对第一二个式子进行运算，消去x      new\_first=elimination(first,second)#用1，2式消去第一个未知数得到的一个式子        if(flag==3):          #对第一三个式子进行运算，消去x          new\_second=elimination(first,last)#消元x多项式后的第二个式子          #对新得到的两个式子消去y          new\_last=elimination(new\_first,new\_second)#消去y后的式子系数      #三元代入计算z,y,x      if(flag==3):          z=new\_last[-1]/new\_last[-2]          y=(new\_first[-1]-new\_first[-2]\*z)/new\_first[0]#由前式计算y          x=(first[-1]-first[-2]\*z-first[-3]\*y)/first[0]#由前式计算x          solution=[x,y,z]      #二元代入计算y,x      if(flag==2):          y=new\_first[-1]/new\_first[-2]          x=(first[-1]-first[-2]\*y)/first[0]          solution=[x,y]      return solution  def Predict(ans\_list,x):      """      由拟合函数预测第x年的人口数      输入参数：      ans\_list:拟合函数的系数      """      ans=0      for i in ans\_list:          ans=ans+i\*math.pow(x,ans\_list.index(i))      return ans   1. 二次项拟合代码：   def Quadratic(x,y,n):      """      二次多项式拟合      """      Y=y.T#将Y转为列向量      transposed\_A=np.matrix([np.ones(n),x,x\*x])#构造矩阵A(转置)      matrix\_A=transposed\_A.T#转置得到A      matrix\_ATA=transposed\_A@matrix\_A#矩阵乘法AAT，法方程的左边系数      matrix\_ATY=transposed\_A@Y#法方程右边系数      coe=matrix\_ATA.tolist()#将左边系数矩阵的未知数部分转为矩阵        #将右边系数添加至对应的左边系数行中，构成方程的系数矩阵      cnt=0      for i in coe:          i.append(matrix\_ATY[cnt,0])          cnt+=1      so=solve(coe)#解三元一次方程      print("二次多项式拟合结果为:\n"+str(so[0]),end="")      if(so[1]>0):          print("+",end="")      print(str(so[1])+"x",end="")      if(so[2]>0):          print("+",end="")      print(str(so[2])+"x²")      pre=Predict(so,1965.)      print("1965年的人口数约为："+str(int(pre)))      pre=Predict(so,2002.)  print("2002年的人口数约为："+str(int(pre)))  #主函数  if \_\_name\_\_=='\_\_main\_\_':      pairs=[1920.,105711.,1930.,123203.,1940.,131669.,1950.,150697.,1960.,179323.,1970.,203212.]#依次列出各点的x,y值      #pairs=[-1,0.22,-0.5,0.8,0,2.,0.25,2.5,0.75,3.8]      #pairs=[-3,4,-2,2,-1,3,0,0,1,-1,2,-2,3,-5]      #pairs=[-1.15,0.22,-0.5,0.8,0.1,2,0.25,2.5,0.75,3.8]      if(len(pairs)%2!=0):          print("输入有误")          exit()      n=int(len(pairs)/2)      x=np.array([pairs[i] for i in range(len(pairs)) if i%2==0])      y=np.matrix([pairs[i] for i in range(len(pairs)) if i%2!=0])      Quadratic(x,y,n)  结果截图：（单位千）    （2）通过观察曲线大致形态，选用线性拟合函数或指数函数拟合较好。    线性：  def Linear(x,y,n):      """      线性拟合      """      Y=y.T#将Y转为列向量      transposed\_A=np.matrix([np.ones(n),x])#构造矩阵A(转置)      matrix\_A=transposed\_A.T#转置得到A      matrix\_ATA=transposed\_A@matrix\_A#矩阵乘法AAT，法方程的左边系数      matrix\_ATY=transposed\_A@Y#法方程右边系数      coe=matrix\_ATA.tolist()#将左边系数矩阵的未知数部分转为矩阵        #将右边系数添加至对应的左边系数行中，构成方程的系数矩阵      cnt=0      for i in coe:          i.append(matrix\_ATY[cnt,0])          cnt+=1      so=solve(coe)#解二元一次方程      print("线性拟合结果为:\n"+str(so[0]),end="")      if(so[1]>0):          print(" + ",end="")      print(str(so[1])+"x")      pre=Predict(so,1965)      print("1965年的人口数约为："+str(int(pre)))      pre=Predict(so,2012)      print("2012年的人口数约为："+str(int(pre)))  指数：  def Exponent(x,y,n):      """      指数拟合      """      #将y值转为ln(y)      for i in range(n):          y[0,i]=math.log(y[0,i])          a=1      Y=y.T#将Y转为列向量      transposed\_A=np.matrix([np.ones(n),x])#构造矩阵A(转置)      matrix\_A=transposed\_A.T#转置得到A      matrix\_ATA=transposed\_A@matrix\_A#矩阵乘法AAT，法方程的左边系数      matrix\_ATY=transposed\_A@Y#法方程右边系数      coe=matrix\_ATA.tolist()#将左边系数矩阵的未知数部分转为矩阵        #将右边系数添加至对应的左边系数行中，构成方程的系数矩阵      cnt=0      for i in coe:          i.append(matrix\_ATY[cnt,0])          cnt+=1      so=solve(coe)#解二元一次方程      print("指数增长拟合结果为:\ne^"+str(so[0]),end="")      if(so[1]>0):          print(" + ",end="")      print(str(so[1])+"x")      pre=Predict(so,1965)      print("1965年的人口数约为："+str(int(math.exp(pre))))      pre=Predict(so,2012)      print("2012年的人口数约为："+str(int(math.exp(pre))))  #主函数：  if \_\_name\_\_=='\_\_main\_\_':      pairs=[1920.,105711.,1930.,123203.,1940.,131669.,1950.,150697.,1960.,179323.,1970.,203212.]#依次列出各点的x,y值      #pairs=[-1,0.22,-0.5,0.8,0,2.,0.25,2.5,0.75,3.8]      #pairs=[-3,4,-2,2,-1,3,0,0,1,-1,2,-2,3,-5]      #pairs=[-1.15,0.22,-0.5,0.8,0.1,2,0.25,2.5,0.75,3.8]      if(len(pairs)%2!=0):          print("输入有误")          exit()      n=int(len(pairs)/2)      x=np.array([pairs[i] for i in range(len(pairs)) if i%2==0])      y=np.matrix([pairs[i] for i in range(len(pairs)) if i%2!=0])      Linear(x,y,n)  Exponent(x,y,n)  结果截图：（单位千） | | | | | |

|  |
| --- |
| **四、实验小结**（包括问题和解决方法、心得体会、意见与建议等）  问题：  （1）构造矩阵时要分清法方程中AT 与A  （2）此小节使用了较多的矩阵及矩阵运算，需要熟悉numpy模块  （3）各种拟合函数需要不断地调试，才能得出完全正确的结果。  解决方法：   1. 通过在网上查询相关的方法并学习 2. 在写代码时，还要一边熟悉计算的步骤与过程，才能写出正确的算法   心得体会：  用计算机语言将拟合算法实现使我更加深了对该算法的理解，并进一步熟悉了求解过程以及矩阵的运算，同时将数学计算过程转换为代码的过程也让我理解到通过数学推导过程可以优化算法，降低其时间与空间复杂度。此外，使用面向对象的语言应该时刻有面向对象的思想，要先确定结构再开始写代码，节省后期整理代码、封装函数的时间。 |