**2016年秋matlab期中大作业（40分，共5道题，每题8分）**

**姓名： 霍峻杰**

**学号： 1140320206**

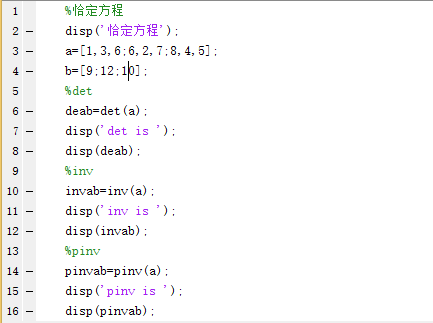
**学院： 计算机科学与技术学院**

**专业： 信息安全**

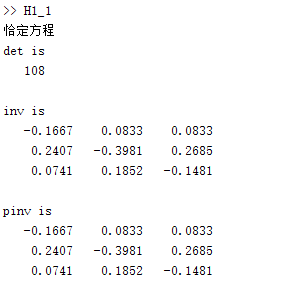
1. 解线性方程 （**第二章线性方程**）

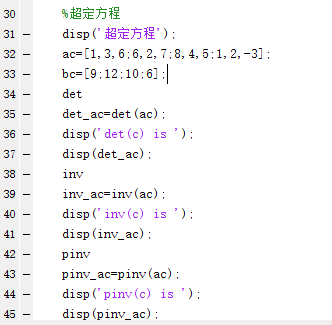
和 

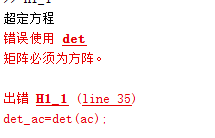
1. 分别求上两个系数矩阵的行列式det、逆inv、伪逆pinv
2. 恰定方程：
   1. 运行代码：

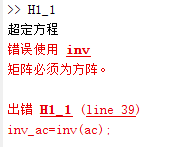


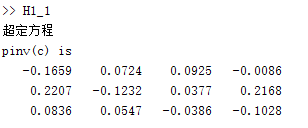
* 1. 运行结果：



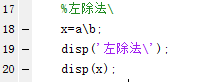
1. 超定方程：
   1. 运行代码：  
      
   2. 运行结果：



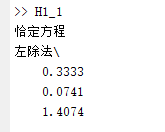




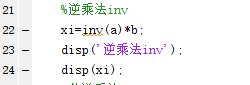
1. 分别采用左除法、逆乘法和伪逆乘法求解；
2. 恰定方程：
   1. 左除法：
      1. 运行代码：



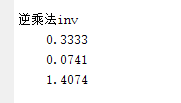
* + 1. 运行结果：



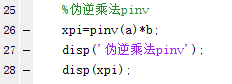
* 1. 逆乘法：
     1. 运行代码：



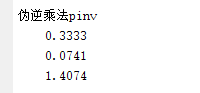
* + 1. 运行结果：



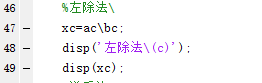
* 1. 伪逆乘法：
     1. 运行代码：



* + 1. 运行结果：



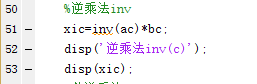
1. 超定方程：
   1. 左除法：
      1. 运行代码：



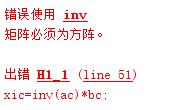
* + 1. 运行结果：



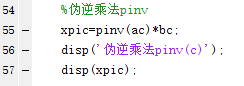
* 1. 逆乘法：
     1. 运行代码：



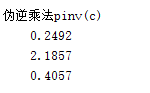
* + 1. 运行结果：



* 1. 伪逆乘法：
     1. 运行代码：



* + 1. 运行结果：



1. 对比两个方程，解释逆乘法和伪逆乘法区别。（8分）

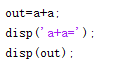
逆和伪逆的区别：

如果矩阵A不是一个方阵，或者A是一个非满秩的方阵时，矩阵A没有逆矩阵，伪逆函数在求解这种系数矩阵为严重病态问题时可避免“伪解”的产生，即找到一个与A的转置矩阵A‘同型的矩阵B，使得：A·B·A=A ；B·A·B=B；此时称矩阵B为矩阵A的伪逆，也称为广义逆矩阵。从结果中的可看出逆矩阵和伪逆矩阵区别在于如果方阵行列式为零，得到的结果不同。方阵行列式不为零，逆矩阵和伪逆矩阵的计算值相同，反之则不同

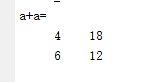
1. 四个矩阵a 、b、c和d，求matlab运算 （**第二章矩阵数组运算**）

， ，，

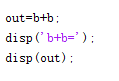
1. 哪些矩阵可以做加法运算（至少3个）
2. a+a,
3. 运行代码：



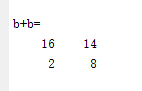
1. 运行结果：



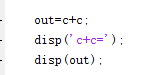
1. b+b
   1. 运行代码：



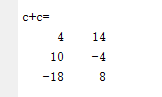
* 1. 运行结果：



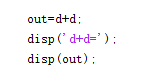
1. c+c
   1. 运行代码：



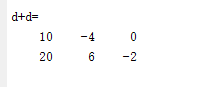
* 1. 运行结果：



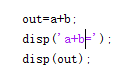
1. d+d
   1. 运行代码：



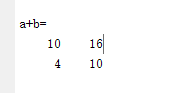
* 1. 运行结果：



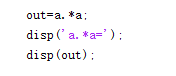
1. a+b
   1. 运行代码：



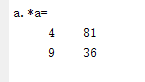
* 1. 运行结果：



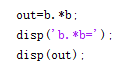
1. 哪些可以点乘数组运算（至少3个）
2. a.\*a
   1. 运行代码：



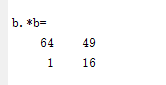
* 1. 运行结果：



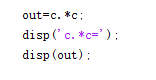
1. b.\*b
   1. 运行代码：



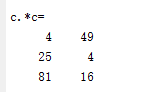
* 1. 运行结果：



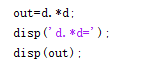
1. c.\*c
   1. 运行代码：



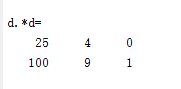
* 1. 运行结果：



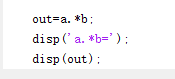
1. d.\*d
   1. 运行代码：



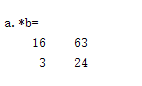
* 1. 运行结果：



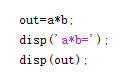
1. a.\*b
   1. 运行代码：



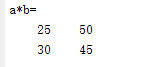
* 1. 运行结果：



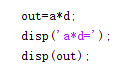
1. 哪些可以矩阵乘法运算（至少3个）
2. a\*b
   1. 运行代码：



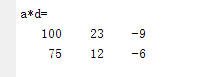
* 1. 运行结果：



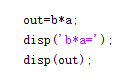
1. a\*d
   1. 运行代码：



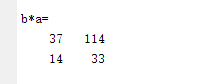
* 1. 运行结果：



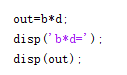
1. b\*a
   1. 运行代码：



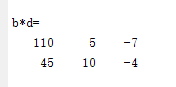
* 1. 运行结果：



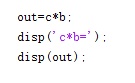
1. b\*d
   1. 运行代码：



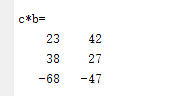
* 1. 运行结果：



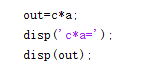
1. c\*b
   1. 运行代码：



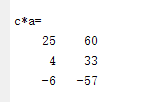
* 1. 运行结果：



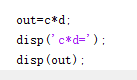
1. c\*a
   1. 运行代码：



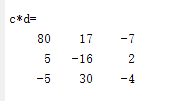
* 1. 运行结果：



1. c\*d
   1. 运行代码：



* 1. 运行结果：



1. 说明数组运算和矩阵运算的特点。（8分）

数组运算：点运算，维数相同，各对应元素运算

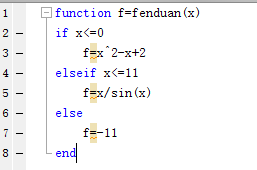
单纯数值汇集起来参与的批量运算

矩阵运算：严格遵守线性代数的规则运算

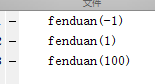
矩阵作为整体参与运算

3. **已知计算分段函数**（**第三章编程，条件语句**）

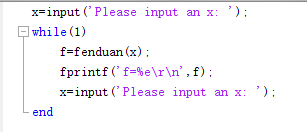
1. 要求采用条件语句if编写脚本文件和函数文件计算分段函数文件；
2. 函数文件：



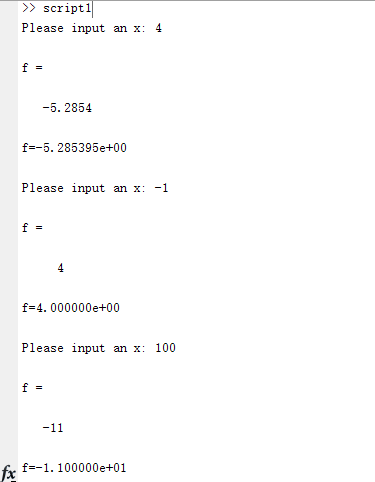
1. 脚本文件：



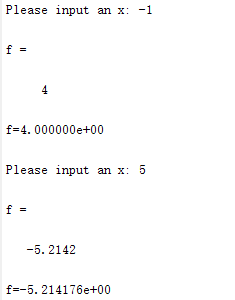
1. x由input指令从键盘给出,数据的输出f(x)值采用fprintf('f=%e',f)
2. 运行代码：



1. 运行结果：



1. 运行脚本文件计算x=-1的f值和运行函数文件计算x=5的f值



1. 说明脚本文件和函数文件区别

区别：

脚本文件：

1. 多条命令的综合体
2. 没有输入输出变量
3. 使用matlab基本工作空间
4. 没有函数声明行

函数文件：

1. 常用于扩充matlab函数库
2. 可以包含输入输出变量
3. 运算中生成的所有变量都放在函数工作空间
4. 包含函数声明行
5. **多项式运算 （第四章多项式）**

y =x4 + 4 x + 5

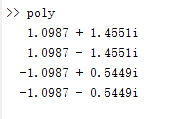
已知：第一个多项式是

1. 表示多项式p1=[]



1. 采用roots求根





1. 采用polyval求x=6时y值





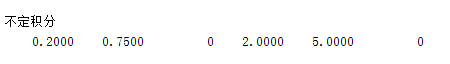
1. 采用polyder对多项式一阶求导





1. 采用polyint对多项式不定积分

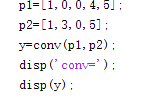




y=x3 + 3 x2 + 5

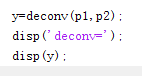
已知第二个多项式为

1. 求两个多项式的乘积conv（p1，p2）





1. 求两个多项式的除法deconv（p1，p2）

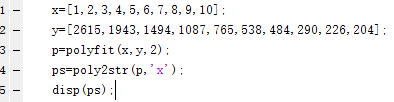




1. **第四章差值拟合**

某年美国旧车价格的调查资料如下表，其中*xi*表示轿车的使用年数，*yi*表示相应的平均价格。

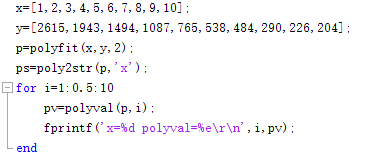
1. 2次多项式拟合（polyfit）
2. 运行代码：



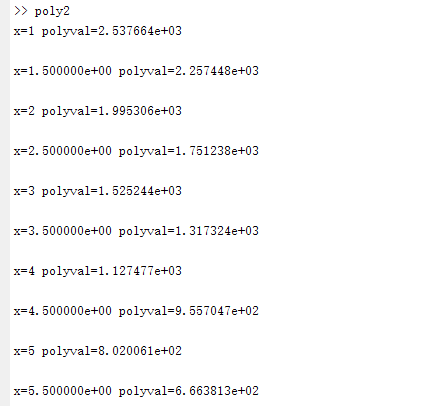
1. 运行结果：

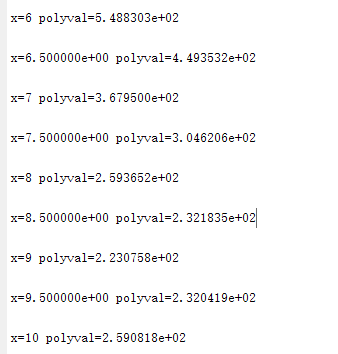


1. 采用2次拟合多项式预测每隔0.5年的价格(polyval)，再用插值法（interpl）预测一下；
2. 2次拟合多项式：
   1. 运行代码：

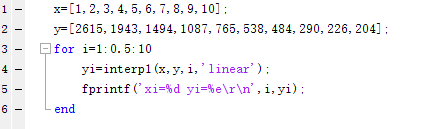


* 1. 运行结果：

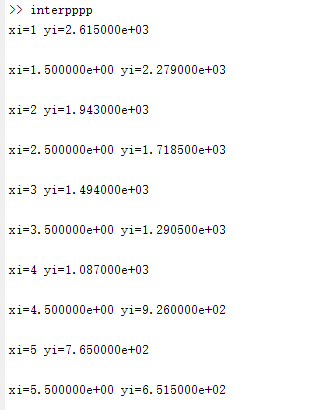


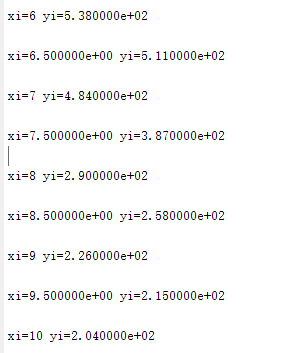


1. 插值法:
   1. 运行代码：

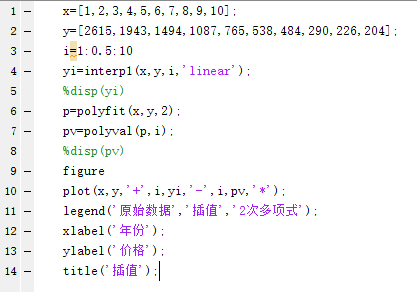


* 1. 运行结果：

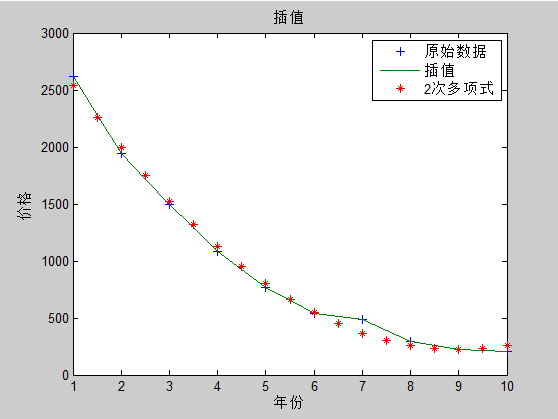




1. 绘制年数-价格曲线，要求用plot不同线型（LineSpec）绘制原始数据点、插值数据和2次多项式曲线，标注坐标轴为‘年份’和‘价格 ’（xlabel、ylabel）、图形标题（title）为‘插值’标注图例legend ，说明是否阶数越到高越好
2. 运行代码：



1. 运行结果：



1. 阶数不是越大越好，因为多项式次数要适当，过低误差大，过高波动明显
2. 说明插值和拟合的特点

若要求所求曲线通过所给的所有数据点，是插值问题。

若不要求曲线通过索引数据点，而是要求他反应对象的整体变化趋势，即为数据拟合。

函数插值与曲线拟合都是要根据一组数据构造一个函数作为近似，由于近似的要求不同，二者的数学方法上是完全不同的。

***xi***

**1**

**2**

**3**

**4**

**5**

**6**

**7**

**8**

**9**

**10**

***yi***

**2615**

**1943**

**1494**

**1087**

**765**

**538**

**484**

**290**

**226**

**204**