中国科学技术大学
University of Science and Technology of China

社科研讨课

人口增长预测模型

学院：数学科学学院

成员：杨隽祎，黄小科，郑伟一

日期：2002/5/9

Contents

1 介绍

近年来人口增长成为一大热点，作为人口大国，对人口的预测影响着我国政策的制定与实施，进行恰当的建模与数据处理，在此切实应用的问题中，熟练对知识的掌握与应用。

2 问题

中国是一个人口大国，人口问题始终是制约我国发展的关键因素之一。根据已有数据，运用数学建模的方法，对中国人口做出分析和预测。

从中国的实际情况和人口增长的上述特点出发，参考相关数据，建立中国人口增长的数学模型，并由此对中国人口增长的中短期和长期趋势做出预测，特别要指出模型中的优点与不足之处。

3 讨论

本题来源于实际问题，需要真实有效的数据。确定好方向后，我们三人便从网络寻找数据，例如近些年来人口总数，出生率，死亡率，各年龄段人口总数等。

此次我们做了稍微调整，讨论学习相关数据处理，学习讨论模型的合理性，多角度切入，在结果处给出模拟结果与真实情况的对比，以此评定模型的合理性，进而分析相关结论。

4 困难

尽管有上学期运筹学建模题目的引导，我们组仍然面对诸多问题。

1. 模型建立较为多样，如何选择合适的模型。
2. 较为专业的内容难以涉猎。
3. 自身知识的有所欠缺，编程能力较为薄弱
4. 疫情影响，前期准备阶段组员之间线下联系受阻。

5 解决

认识到上述问题，我们组讨论后决定，改变上学期分组方式，采取三人讨论给出若干相关模型，再由组员分别切入不同模型，同时编程处理数据，最终讨论各个模型的优劣。

其次我们数据均来自国家统计局（《中国统计年鉴》）。模型知识学习自姜启源老师的《数学模型》及其译制的《数学建模》。

6 模型一

6.1 建模

模型：灰色预测模型-GM(1,1)

将输入的数据记作一列数据（记为：$x_0 = (x_0(1), ..., x_0(n))$），引入累加，逆累加，均值，级比生成算子：

累加生成：将原序列的数据依次累加的到生成序列（记为：$x_1 = (x_1(1), ..., x_1(n))$）

逆累加生成：将生成序列相邻数据作差生成新序列（记为：$y_1 = (y_1(1), ..., y_1(n))$）

均值生成：生成序列相邻数据作均值生成新序列（记为：$z_1 = (z_1(2), ..., z_1(n))$）

级比生成：原序列相邻数据作比生成新序列（记为：$\sigma = (\sigma(2), ..., \sigma(n))$）

若级比满足

$$\sigma(k) \in (e^{-\frac{2}{n+1}}, e^{\frac{2}{n+1}})$$

则该序列可以进行灰色预测
数学表达：
$x_1(k) = \sum_{m=1}^{k} x_0(m)$　　$y_1(k) = x_1(k) - x_1(k-1)$
$z_1(k) = \frac{(x_1(k)+x_1(k-1))}{2}$　　$\sigma(k) = \frac{x_0(k-1)}{x_0(k)}$
则定义序列间基本关系：$x_0(k) + az_1(k) = b$
其中设

$$P = \begin{pmatrix} a \\ b \end{pmatrix} \quad Y = \begin{pmatrix} x_0(2) \\ x_0(3) \\ ... \\ x_0(n) \end{pmatrix} \quad B = \begin{pmatrix} -z_1(2) & 1 \\ -z_1(3) & 1 \\ ... & ... \\ -z_1(n) & 1 \end{pmatrix} \tag{1}$$

最小二乘法估计参数列：$\hat{P} = (\hat{a}, \hat{b})^T = (B^T B)^{-1} B^T Y$
利用离散数据序列建立近似的微分方程模型：$\frac{dx_1}{dt} + ax_1 = b$
解得时间相应函数：$x_1(t) = (x_0(1) - \frac{b}{a})e^{-at} + \frac{b}{a}$
由此可得原始数据的预测值：

$$\hat{x}_0(k+1) = \hat{x}_1(k+1) - \hat{x}_1(k) = (1 - e^{\hat{a}})(x_0 1 - \frac{\hat{b}}{\hat{a}})e^{-\hat{a}k}$$

精度检验引入残差，相对误差的定义：
残差：$q(k) = x_0(k) - \hat{x}_0(k)$
相对误差：$\epsilon(k) = \frac{q(k)}{x_0(k)} \times 100\%$
精度：$p^0 = (1 - \frac{1}{n-1}\sum_{k=2}^{n} |\epsilon(k)|) \times 100\%$

6.2 应用

将已知的2001年至2010年的我国总人口数记为原始序列x_0
计算得出级比序列，经判断可以进行灰色预测。
将序列数据带入GM(1,1)模型中求解得：

$$a = -0.005354008906189196$$

$$b = 127547.90292940235$$

故得到时间响应函数为：

$$x_1(t) = 23950505.363530155e^{0.005354008906189196} - 23822878.363530155$$

整理为：

$$x_1(t) = 23950505e^{0.005354009} - 23822878$$

根据已知的十年数据我们可以预测得到2001年至2021年的数据，如下表格：
其中计算可得精度：$p^0 = 99.9616543322243\%$
此模型能够较为准确的预测我国人口数量。

年份	真实值	预测值	相对误差
2001	127627	127627	0
2002	128575.4	128453	0.000953002
2003	129265.7	129227	0.000299146
2004	129959.6	129988	0.000218444
2005	130657.3	130756	0.000755014
2006	131358.7	131448	0.000679391
2007	132063.9	132129	0.000492861
2008	132772.8	132802	0.000219516
2009	133485.6	133450	0.000266939
2010	134202.2	134091	0.00082947
2011	134922.7	134916	4.94609E-05
2012	135647	135922	0.002023298
2013	136375.2	136726	0.00256576
2014	137107.3	137646	0.003913606
2015	137843.4	138326	0.003489208
2016	138583.3	139232	0.00465879
2017	139327.3	140011	0.004883077
2018	140075.3	140541	0.003313784
2019	140827.3	141008	0.001281808
2020	141583.3	141212	0.00262916
2021	142343.3	141260	0.007669134

6.3 运行结果

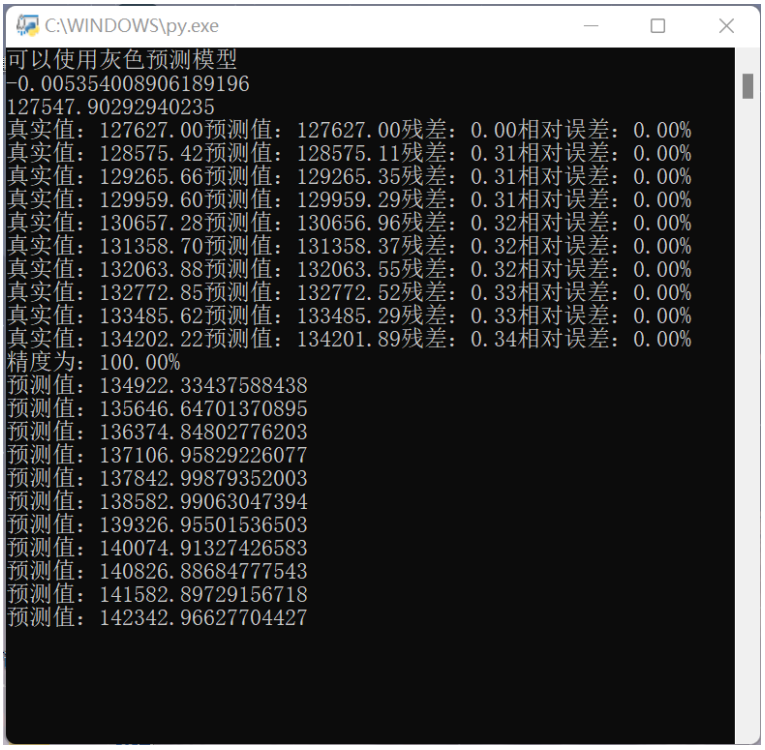


Figure 1: python实现

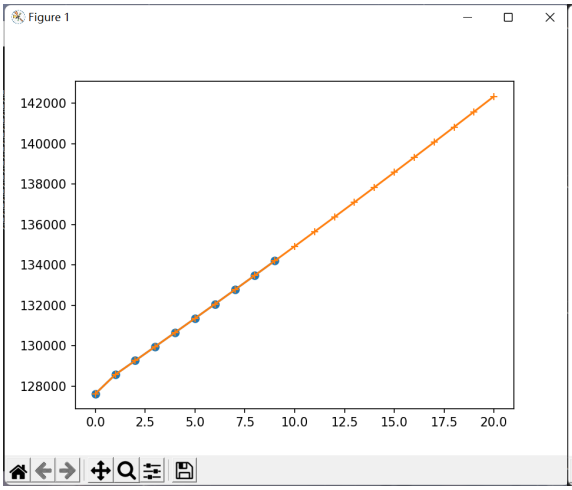


Figure 2: 拟合效果1

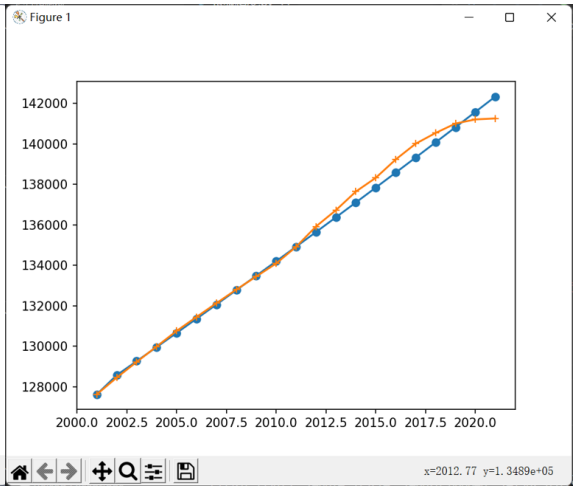


Figure 3: 拟合效果2

7 模型二

7.1 分析

不同年龄的人在死亡和生育方面存在差异。根据如下表所示的人口统计资料（2015年），建立描述人口增长的模型，并对每个时间段和每个年龄组计算出相应的增长率，同时计算总人口数及每个时间段末各年龄组的人数与总人数的百分比，计算年限为19个5年的时间段。

	A	B	C	D
1	指标	女婴出生率	女性人口存活率	女性人数/万人
2	**0-4岁**	0	0.9967	3733
3	**5-9岁**	0	0.99837	3480
4	10-14岁	0	0.9978	3277
5	15-19岁	0.04595	0.99672	3500
6	20-24岁	0.2748	0.99607	4840
7	25-29岁	0.37155	0.99472	6394
8	30-34岁	0.22655	0.9924	5056
9	35-39岁	0.093	0.98867	4793
10	40-44岁	0.02685	0.98274	5794
11	45-49岁	0.0155	0.97437	6134
12	50-54岁	0	0.96258	5176
13	**55-59岁**	0	0.94562	3807
14	60-64岁	0	0.91522	3936
15	65-69岁	0	0.86806	2761
16	70-74岁	0	0.80021	1861
17	75-79岁	0	0.69239	1399
18	80-84岁	0	0.77312	919
19	85以上	0		897

7.2 模型假设

1、讨论对象是年龄区间长为5年（年龄超过85岁的人不按此年龄区间分组）的年龄组人数，按年龄依次分为18组。

2、同一年龄组的女性和男性有相同的存活率，且存活率不变，表中没有的数据视作存活率为0.

3、同一年龄组的女性生男孩和生女孩的概率相同，且出生率不变，表中没有的数据视作出生率为0.

7.3 符号约定

t——时间，单位为年
$x_k(t)$————t时刻第k个年龄组的人口数量，单位为万人
$x_k(0)$————初始时刻第k个年龄组的人口数量，单位为万人
$x(t)$————t时刻的人口数量，单位为万人
b_k——第k组妇女的生育率
b_{wk}——第k组妇女生育女婴的比率，其中$b_{wk}=\frac{b_k}{2}$
s_k————第k组人口的存活率

7.4 建模

t时刻的人口数量为

$$\mathsf{x}(t)=\begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ ... \\ x_n(t) \end{pmatrix} \tag{2}$$

考虑在t时刻到$t+5$时刻的人口变化。由假设知$\frac{x_k(t)}{2}$表示第k个年龄组在t时刻的妇女数于是第k个年龄组妇女在t到t+5时刻生育且存活的人数为

$$b_k\frac{x_k(t)}{2}=b_{wk}x_k(t) \tag{3}$$

此外，在t+5时刻第一个年龄组的人口是由在t到t+5时刻期间出生且活到t+5时刻的人构成的。由数据可计算出

$$x_1(t+5)=b_{w1}x_1(t)+b_{w2}x_2(t)+...+b_{w18}x_{18}(t) \tag{4}$$

由于部分组女性无生育能力，故一部分$b_{wk}=0$。在t+5时刻，第2至第17个年龄组的人数由相应在t时刻的第1至第16个年龄组的存活人数构成，故由定义有

$$x_k(t+5)=s_{k-1}x_{k-1}(t) \tag{5}$$

式（4）和（5）就是人口增长的年龄结构模型，可用矩阵表示为

$$\mathsf{x}(t+5)=\mathsf{G}\mathsf{x}(t) \tag{6}$$

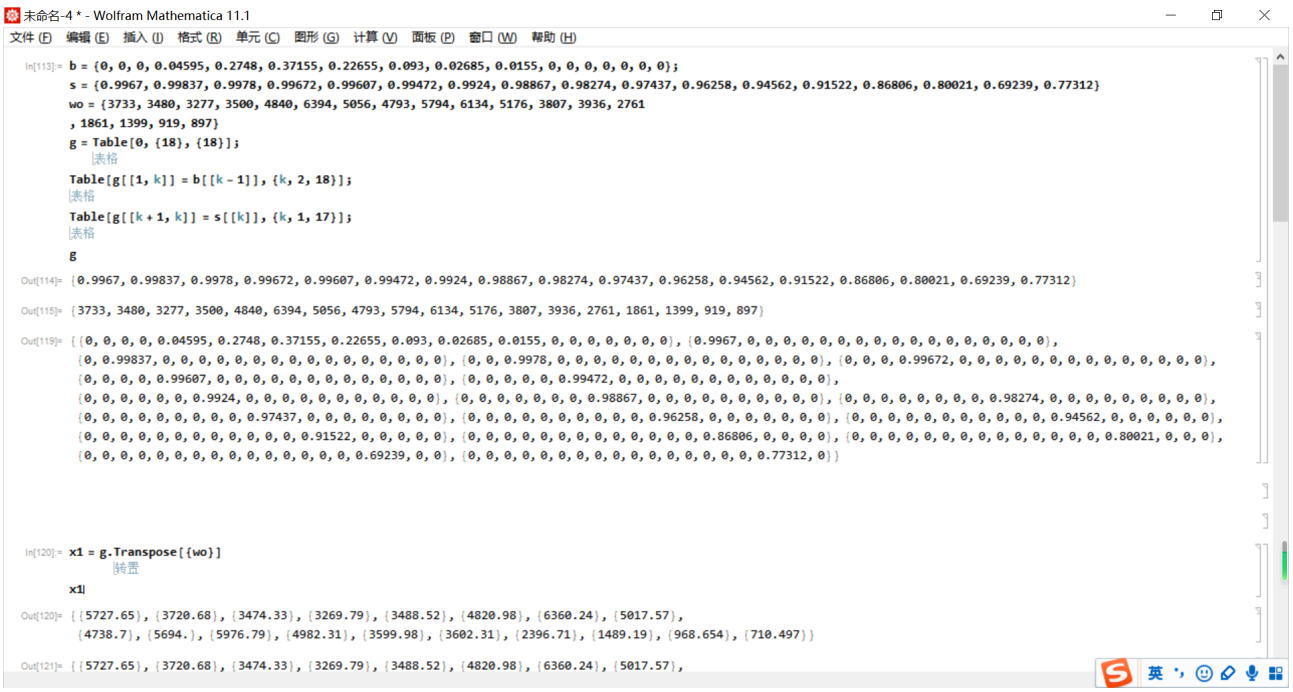
其中

$$\mathsf{G}=\begin{pmatrix} b_{w1} & b_{w2} & b_{w3} & ... & b_{w17} & b_{w18} \\ s_1 & 0 & 0 & ... & 0 & 0 \\ 0 & s_2 & 0 & ... & 0 & 0 \\ 0 & 0 & s_3 & ... & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & ... & s_{17} & 0 \end{pmatrix} \tag{7}$$

称G为增长矩阵。给定初始时刻人口总数x(0)，由式（5）可以递推出确定人口增长的矩阵方程:

$$\mathsf{x}(5n)=\mathsf{G}^n\mathsf{x}(0) \tag{8}$$

利用式（7）可以计算出任意时间段的人口分布情况.



7.5 计算结果

根据上述算法流程，用Mathematica软件进行计算，如图所示
计算得到

$$x(5)=2\begin{pmatrix}5727.65\\3720.68\\3474.33\\3269.79\\3488.52\\4820.98\\6360.24\\5017.57\\4738.7\\5694.\\5976.79\\4982.31\\3599.98\\3602.31\\2396.71\\1489.19\\968.654\\710.497\end{pmatrix}=\begin{pmatrix}11455.3\\7441.36\\6948.66\\6539.58\\6977.04\\9641.96\\12720.5\\10035.1\\9477.39\\11388.\\11953.6\\9964.63\\7199.95\\7204.61\\4793.43\\2978.38\\1937.31\\1420.99\end{pmatrix}$$

各分量相加得总人口数为140078（万人）

7.6 反思与讨论

1、2020年实际总人口数为141178万人，精度约为99.22%。故此模型对于预测人口数较为准确。
2、模型的优点：
(1)建立的模型与实际紧密联系，结合实际情况对所提出的问题进行求解，可迁移应用。
(2)通过利用数学工具和mathematica的运用，严格地对模型求解，具有科学性。
(3)模型能比较准确地预测我国中短期人口增长情况，如总人口、年龄结构分布等。
(4)对于我国人口增长情况的预测及人口控制提供了有价值的参考。
3、模型的不足：
(1)Leslie模型最初的假设过于理想，而且每年的数据持续变化，不一定相同。
(2)Leslie模型没有考虑社会因素（如生孩政策）、自然因素（如自然灾害）等一系列因素。
(3)数据要求高，实现难度大。

8 总结

灰色预测模型(GM(1,1))能仅根据已知的若干数据，准确的拟合将来的人数，与已知的数据预测相比，准确性与便利性较高，配合上python的应用，能够实现简便准确的预测。不过缺点也较为明显，这个模型首先需要对数据分析，判断是否能够运用这个模型，对于那些我们想要去预测的，但是不满足条件的，则无法使用。

Leslie模型的逻辑简单易懂，便于理解使用，对人口预测这一特定问题有较强的针对性，思路迁移可以将此问题拓展开，解决其他数据的预测问题，受数据本身的限制较少。但是Leslie模型对已知数据的种类和准确度有着极高的要求，增长率、消亡率、总数等都要求精确到各个年龄段，而且预测的范围也受年龄段大小的影响，局限性较强。

9 反思

从薄利多销问题到人口预测，两道题目均是运筹学范畴。无论是建模还是对数据的处理，误差分析，都要求我们对曲线的拟合讨论，这恰恰是二者最重要的关系。相比薄利多销问题，人口预测问题对模型处理，编程能力都提出了更高的要求。其中的Leslie模型更是能将现有的线代知识加以应用，深化认知。

10 插曲

尝试使用另一种混合预测的方法，思路更简单一些，根据已知的出生率和死亡率，去预测未来的出生率和死亡率，根据所得到的所有的数据，用最初年份的总人口作乘法运算得到对应年份的人口，不过经实验发现，这个方法得误差较大。思考过后发现其中可能存在如下原因：

1)对于已知的总数据中，我们有总人口，出生率，死亡率，但是发现两年总人口的对比计算得到的增长率并不等于已知的出生率-死亡率得到的自然增长率，单纯有后者数据必然存在误差。

2)使用灰色预测模型去预测出生率，死亡率有较大的误差(虽然不超过1%)，在乘法趋近于指数运算时，小小的误差都会造成结果发生较大偏差。

3)或者可能是还存在其他纰漏我们尚未发现。

11 附录——代码

11.1 代码——GM(1,1)

```
#GM(1,1)
from numpy import *
import math
import sys
import copy
import matplotlib.pyplot as plt
n = input("total known  data")
n=int(n)
k = input("number of predicted  data")
k = int(k)
s1 = pow(math.e,(-2)/(n+1))
s2 = pow(math.e,2/(n+1))
#x0-origin data
x0=[]
for i in range (n):
  X = input()
  X = float(X)
  x0.append(X)
#save for judgement
c=[]
for i in range(0,n-1):
  C = x0[i]/x0[i+1]
  c.append(C)
#judge
for temp in c:
  if(temp>s1 and temp < s2):
    continue
  else:
    print("NO")
    sys.exit(0)
print("YES")
#x1
x1=[]
X1 = 0
for i in range(n):
  X1=X1+x0[i]
  x1.append(X1)
#z1
z=[]
for i in range(n-1):
  Z = (x1[i+1]+x1[i])/2.0
  z.append(Z)
#Y,D
y=copy.deepcopy(x0)
del(y[0])
Y=matrix(y)
Y=Y.T
b0=copy.deepcopy(z)
B = matrix(b0)
B = (-1)*B
B = B.T
#problem(solve)
V = ones((n-1, 1))
D = c_[B,V]
# ab
A = (((D.T)*D).I)*(D.T)*Y
a = A[0,0]
b = A[1,0]
print(a)
print(b)
#m
```

```
m = []
m.append(x1[0])
for i in range(1,k):
  M = (x0[0]-b/a)*pow(math.e,((-1)*a*i))+b/a
  m.append(M)
#e(predicted data)
e = []
for i in range(k):
  if(i==0):
    e.append(m[i])
  else:
  E = m[i]-m[i-1]
  e.append(E)
#analysis
o1=[]
o2=[]
ave = 0.0
for i in range(n):
  O = (x0[i]-e[i])
  o1.append(O)
  O = O/x0[i]
  ave = ave + O
  o2.append(O)
ave = O/(n-1)
p = 1 - fabs(ave)
#print
for i in range(n):
  print(" true{:.2f} predict{:.2f}  {:.2f} epsilo{:.2%}".format(x0[i],e[i],o1[i],o2[i])
                                                    )
print(" p0{:.2%}".format(p))
#figure
xpoints1 = []
for i in range(n):
  xpoints1.append(i)
xpoints2 = []
for i in range(k):
  xpoints2.append(i)
for i in range(n,k):
  print(" predict{}".format(e[i]))
#save the data
fp = open("1.txt",'w+')
for item in e:
  fp.write("%f"%item)
  fp.write('\n')
fp.close()
plt.plot(xpoints1, x0,'o-')
plt.plot(xpoints2, e,'+-')
plt.show()
```

```
#further analysis
from numpy import *
import math
import matplotlib.pyplot as plt
from scipy import optimize
import xlwt
fp1 = open("1.txt",'r')#predict
fp2 = open("2.txt",'r')#true
n = 0
o1=[]
for line in fp1.readlines():
  line = line.strip('\n')
```

```
    line = float(line)
    o1.append(line)
    n=n+1
print(o1)
o2=[]
for line in fp2.readlines():
    line = line.strip('\n')
    line = float(line)
    o2.append(line)
print(o2)
fp1.close()
fp2.close()
e1=[]
e2=[]
ave = 0.0
for i in range(n):
    O = (o2[i]-o1[i])
    e1.append(O)
    O = fabs(O/o2[i])
    ave = ave + O
    e2.append(O)
ave = O/(n)
p = 1 - ave
#save in excel
book = xlwt.Workbook()
sheet = book.add_sheet('data')
col = (' year',' true',' predicted','  epsilo')
for i in range(4):
sheet.write(0,i,col[i])
y=2001
for i in range(1,n+1):
    sheet.write(i,0,y)
    y=y+1
for i in range(1,n+1):
    sheet.write(i,1,o1[i-1])
    sheet.write(i,2,o2[i-1])
    sheet.write(i,3,e2[i-1])
    sheet.write(0,4,p)
savepath = r'C:\Users\24514\Desktop\excel.xls'
book.save(savepath)
xpoint = arange(2001,2022,1)
plt.plot(xpoint,o1,'o-')
plt.plot(xpoint,o2,'+-')
plt.show()
```

11.2 代码——Leslie矩阵

```
import xlwt
import xlrd
from numpy import *
import copy
import matplotlib.pyplot as plt
xls = xlrd.open_workbook('data.xls')
sheet = xls.sheet_by_name('total')
x0 = []
bwk = []
sk = []
for i in range(38,56):
    a = sheet.cell_value(i,4)
    a = float(a)
```

```
    b = sheet.cell_value(i,2)
    c = sheet.cell_value(i,3)
    x0.append(a)
    bwk.append(b)
    sk.append(c)
del(sk[17])
X0 = array(x0)
Sk = array(sk)
Bwk = array(bwk)
Bwk = Bwk.T
#G
D = zeros((17,1))
C = diag(Sk)
B = c_[C,D]
B = B.T
G = c_[Bwk,B]
G = G.T
X0 = X0.T
res = dot(G,X0)
sum = 0
for item in res:
    sum = sum + item
sum = sum*2/0.015
book = xlwt.Workbook()
sheet = book.add_sheet('data')
for i in range(1,19):
    sheet.write(i,0,res[i-1])
sheet.write(0,0,'result')
sheet.write(19,0,'sum')
sheet.write(19,1,sum)
savepath = r'C:\Users\24514\Desktop\ excel.xls'
book.save(savepath)
```