

Initial exercise E0

Charles Bos

v. 2024

1 Introduction

You are set to follow the course *Principles of Programming in Econometrics*. In order to get a bit of a headstart, you'll find attached a first complete program, written in the Python programming language.

During the course, we will study the concepts in detail. To help in understanding those concepts, first try to form yourself an idea: Why would the program be written as it is? What parts are meaningful? What would you have done the same/different/never in a lifetime?

2 Preparation

Section 3 contains the listing of the program. Read this program through, don't use a computer at all at this stage (maybe use a pocket calculator if you really want to).

Ask yourself the following questions, and answer them on a piece of scratch paper/a file in wordpad:

1. Where would execution of the program start?
2. What lines are comments, which are code?
3. What is the system in the naming of the variables?
4. After line 127, the value of `mC` is

$$\text{mC} = \begin{pmatrix} 10 & -7 & -4 & 28 \\ -7 & 59 & 18 & -145 \\ -4 & 18 & 58 & 56 \end{pmatrix}$$

What would its value be after line 132? What would you have written on line 7, instead of the '???'?

5. Matrices are indexed throughout the program. How does this work? Where does the index start?

6. There is something special with the numerics. Where do you encounter the numerical values 0, 1 and 2? Is there a difference in the region of the program where you encounter those numbers? Why?
7. This same program could have been written in some 40 lines of code (of which roughly half initialisation of `vY` and `mX`). What would be possible advantages of the present, rather extensive program, using 139 lines instead?
8. What is the logic between the ‘Inputs’ and ‘Outputs’ that are listed in the program? What goes where, what is the defining difference between the two?
9. (*More difficult*) At line 127, the matrix `mC` is changed. Why does this happen, what would have gone (hopelessly) wrong otherwise?

Think about these questions *before* watching the main set of videos; you are not supposed to answer them all precisely and correctly, that should be easy at the end of the course. You could write for yourself some basic answers, or list your doubt where you don’t see the answer. During the course, tick-off the doubts that are solved, or raise the questions with the instructors.

3 Program `e0_elim.py`

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  """
4  e0_elim.py
5
6  Purpose:
7      ???
8
9  Date:
10     2018/8/28, 2021/8/4
11
12 Author:
13     Charles Bos
14 """
15 #####
16 ### Imports
17 import numpy as np
18
19 #####
20 ### br= ElimElement(mC, i, j)
21 def ElimElement(mC, i, j):
22     """
23     Purpose:
24         Eliminate one element [i,j] of a matrix, subtracting multiples
25         of row j from row i

```

```

26
27     Inputs:
28         mC      iK x iK+iY matrix
29         i       integer, number of row to eliminate
30         j       integer, number of row with pivot
31
32     Outputs:
33         mC      iK x iK+iY matrix, with 0 created in location [i,j]
34
35     Return value:
36         br      boolean, True if all went well
37     """
38     if np.abs(mC[j,j]) < 1e-8:
39         return False
40
41     # Find factor multiplying row j
42     dF= mC[i,j] / mC[j,j]
43
44     # Subtract dF times row j from row i
45     mC[i,j:] = mC[i,j:] - dF*mC[j,j:]
46
47     return True
48
49     #####
50     ## br= ElimColumn(mC)
51     def ElimColumn(mC, j):
52         """
53         Purpose:
54             Eliminate one column[:,j] of a matrix, creating zeros below
55             the pivot at [j,j]
56
57         Inputs:
58             mC      iK x iK+iY matrix
59             j       integer, number of row with pivot
60
61         Outputs:
62             mC      iK x iK+iY matrix, with 0 created below [j,j]
63
64         Return value:
65             br      boolean, True if all went well
66         """
67         br= True
68         iK= np.size(mC, 0)
69         # i= 1
70         for i in range(j+1, iK):
71             # print ('Starting row ', i)
72             br= br and ElimElement(mC, i, j)
73             # print ('resulting in mC= \n', mC)
74

```

```

75     return br
76
77 #####
78 ### br= ElimGauss(mC)
79 def ElimGauss(mC):
80     """
81     Purpose:
82     Eliminate a matrix, creating zeros at lower triangular
83
84     Inputs:
85     mC      iK x iK+iY matrix
86
87     Outputs:
88     mC      iK x iK+iY matrix, with 0 created below main diagonal
89
90     Return value:
91     br      boolean, True if all went well
92     """
93     iK= np.size(mC, 0)
94
95     br= True
96     for j in range(iK):
97         print ('Starting iteration ', j)
98         br= br and ElimColumn(mC, j)
99         print ('resulting in mC= \n', mC)
100
101     return br
102
103 #####
104 ### main
105 def main():
106     print ('Running the original MAIN function from e0_elim')
107
108     # Magic numbers
109     mX= [ [1, 1, 3],
110           [1, -1, -3],
111           [1, -4, -1],
112           [1, 1, -1],
113           [1, 0, 2],
114           [1, 1, -2],
115           [1, 2, 3],
116           [1, 1, -2],
117           [1, -5, 1],
118           [1, -3, -4] ]
119     vY= [ 6, -1, 10, -3, 4,
120          -5, 1, -5, 19, 2]
121
122     # Transform inputs to matrices of floats
123     mX= np.array(mX)

```

```

124
125     iN= np.size(vY)
126     vY= np.array(vY).reshape(iN, 1)
127
128     # Prepare A= X'X, b= X'y, C= [A, b]
129     mA= mX.T@mX
130     vB= mX.T@vY
131     mC= np.hstack((mA, vB))
132
133     mC= mC.astype(float)
134
135     print ('Initial matrix [A | b]: \n', mC);
136
137     # Eliminate the mC matrix, resulting in [ mU | vC ]
138     ir= ElimGauss(mC)
139     print ('ElimGauss returns ir= ', ir,
140           ' with mC= \n', mC)
141
142     #####
143     ### start main
144     if __name__ == "__main__":
145         main()

```