MATRIX ROUTINE APPLICATION PROGRAM: SIMPLEX PROBLEM

The LP program solves the linear maximisation and minimisation problems with a slightly modified Simplex method.Slack variables are not explicitly used,to save memory space.The Simplex maximisation problem can be described as follows:

A number of positive variables are submitted to a number of constraints of the form:

$g_i(x_1,x_2,\ldots,x_n) \leqslant b_i$ for i=1,2,...,m

where the g_i are linear functions.

The problem stated is to find a set of solutions $(x_1,x_2,\ldots,x_n)$ for the constraints g_i so as to maximize a given linear function $f(x_1,x_2,\ldots,x_n)$.

A sample problem:

$$x_1 + 3x_2 \leqslant 300$$
$$x_1 + x_2 \leqslant 160$$
$$2x_1 + x_2 \leqslant 170$$

maximize $f(x_1,x_2)=20x_1 + 30x_2$

The problem can be represented in matrix form as:

$$\left[\begin{array}{cc|c} 1 & 3 & 300 \\ 1 & 1 & 160 \\ 2 & 1 & 170 \\ \hline 20 & 30 & 0 \end{array}\right]$$

Thus, a simplex problem with m constraints and n variables can be mapped on an (m+1)x(n+1) matrix.

The LP routine published here requires such an input matrix. The bottom row stands for the function to maximize.

REMARK: If $f(x_1,x_2,\ldots,x_n)$ contains a constant it must be entered as the bottom right element of the input matrix but with opposite sign!

e.g. if $f(x_1,x_2)=20x_1 + 30x_2 + 100$,the bottom row of the input matrix becomes: $\lfloor 20 \quad 30 \mid -100 \rfloor$

LPI and LPO are initialisation routines for the general input -output routine for dimensioned arrays I have written. You can always use your own favorite input scheme instead.

Take care to initialize the following data before a run of LP:

R07: starting address of the matrix

R08: number of columns=number of variables +1

R09: number of rows=number of constraints +1

Flag 2: clear for maximisation,set for minimisation.

Size requirements: To keep track of the basic and slack variables involved in the algoritm ,an extra row of data is used.The starting address of the input matrix must be 16 or higher.

Back to the sample problem.A run of LP on the input matrix yields the following matrix:

$$\begin{bmatrix} -0.2 & 0.4 & 0 \\ -0.4 & -0.2 & 32 \\ 0.6 & -0.2 & 0 \\ 42 & 86 & 3420 \end{bmatrix}$$

The upper 3 rows are hardly of any further interest to the user.The bottom row stands for the solution as follows: $x_1=42$ and $x_2=86$ give a maximum of 342Ø for the function $f(x_1,x_2)$.This can be verified by substituting this solution into the constraints g_i.

For the minimisation problem,the constraints are of the form $g_i(x_1,x_2,...,x_n) \geqslant b_i$.The objective function f must be minimised.As described above,the problem can again be mapped on a matrix.

The LP routine essentially solves the maximisation problem. To solve the minimisation problem,it is transformed into the former one.This is achieved by first transposing the input matrix,a few changes in the slack variable takeover routine, and finally transposing the matrix again.This is,of course, not the fastest way to solve the minimisation problem,but it saves a lot of extra slack variable registers which are needed when using other methods.Flag 2 is used to select either the first or the second of both problem types.

The routine which transposes the matrix in the 2^{nd} case has been written as a stand alone routine.Here all credits go to JOHN KENNEDY (PPC 918),who wrote an excellent TP program. The version published here only differs from his original version in that it uses the function REGMOVE instead of the PPC ROM "BM".The TP routine transposes any matrix specified by RØ7,RØ8 and RØ9,like the matrix routines.The contents of RØ8 and RØ9 are exchanged by TP,of course.

Technical details:

Data registers:

RØ7: starting address of matrix

RØ8: number of columns=number of variables +1

RØ9: number of rows=number of constraints +1

R1Ø: ISG constant for row selection

R11: ISG pointer to constraint constants

R12: ISG pointer to objective function coëfficiënts.

R13: save pivot address

R14: ISG constant to extra row.

alpha registers M,N and O are used for scratch and loop control.

PPC ROM ROUTINES USED:

"BC","BX","M2","M3","M4","M5"

flag 2: used

display mode:not used

angular mode:not used

A good sample problem for those intending to analyze the system is the following:

$$x_1 + 2x_2 + x_3 \leqslant 2$$
$$2x_1 + 3x_2 + x_3 \leqslant 3$$
$$x_1 + x_2 + 4x_3 \leqslant 4$$

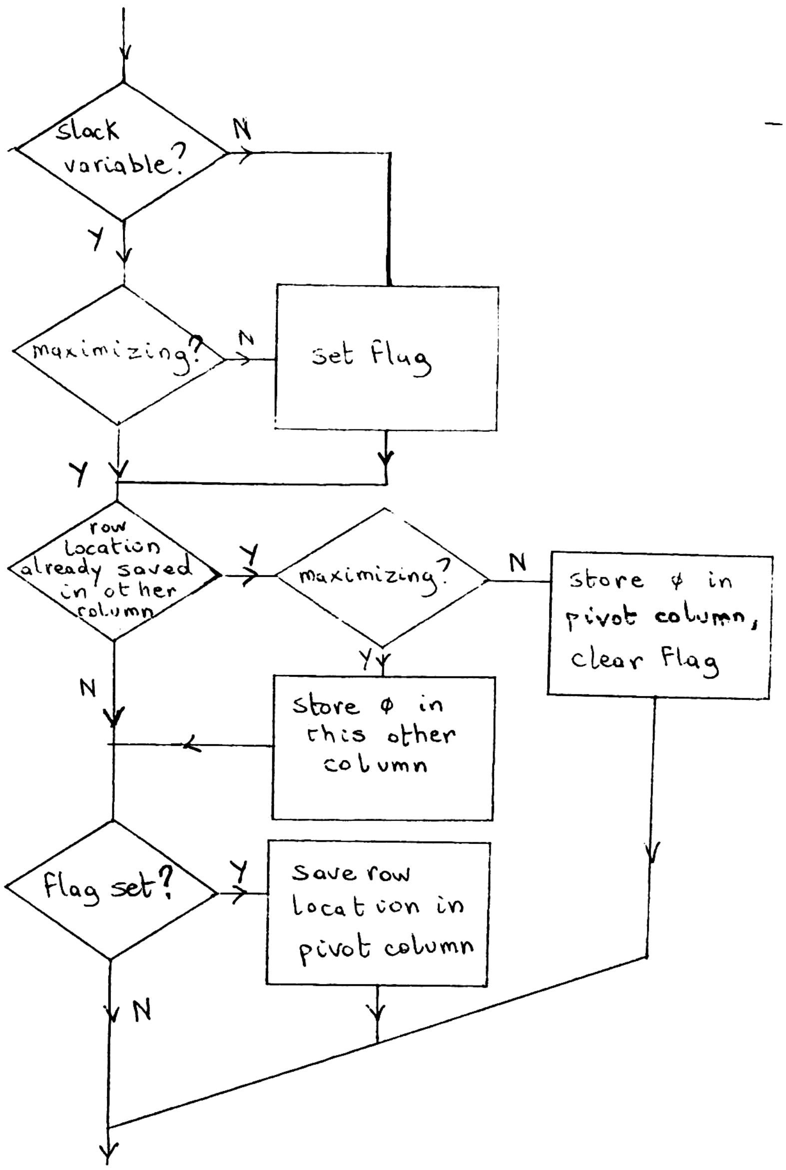
Maximize $18x_1 + 24x_2 + 16x_3$

The solution is $x_1 = 8/7$ $x_2 = Ø$ $x_3 = 5/7$ maximum=32

PHILIPPE ROUSSEL PPC4367BC1

Perhaps the method used needs some more explanation. It can be understood by observing the following: at the start of the solution method, all slack variable coefficients are either set to 1 or 0. When the matrix column pertaining to a slack variable is changed for the first time, the column pertaining to one of the main variables becomes a unity base vector (such as <0,0,1,0>). Such columns are not explicitly needed. They simply indicate that the main variable related to it "has been taken over" by a slack variable. By keeping track of this takeover process in an extra matrix row, the unit vectors are no longer explicitly necessary in the system matrix. Especially for larger systems, the number of storage locations needed can be reduced using the above method, which is certainly interesting when implementing the Simplex method on small systems like the HP-41C*.

slack variable takeover IN LP.RPN S/R @LINE 129.




```
RCL 14
- - - + - - -
STO 0
CLX
RCL IND 0
X≠0?
FS? 02
SF 05
RDN
STO 13
XROM "M2
RCL 14
SIGN
RCL 13
GTO 00
LBL 07
RCL IND L
X<>Y
X≠Y?
GTO 00
CLX
FS? 02
GTO 05
STO IND L
GTO 06
LBL 00
ISG L
GTO 07
LBL 06
RCL 13
LBL 05
FS?C 05
STO IND 0
- - - - - - - -
1
STO 0
LBL 02
```

Sample procedure

1	(2)	1	1	0	0	|	(2)
2	3	1	0	1	0	|	3
1	1	4	0	0	1	|	4
	↑		slack variables				
18	(24)	16	0	0	0		0

largest objective function coeff.

Find smallest a_{in} / a_{i2} with $a_{in} \geq 0$ & $a_{i2} > 0$

1/2	1	1/2	1/2	0	0	1	r_1 / a_{12}
(1/2)	0	−1/2	−3/2	1	0	(0)	$r_2 - a_{22} \cdot r_1$
1/2	0	7/2	−1/2	0	1	3	$r_3 - a_{32} \cdot r_1$
(6)	0	4	−12	0	0	−24	$r_4 - a_{42} \cdot r_1$
	↑ x_2						

Find smallest a_{in} / a_{i1} ...

0	1	1	2	−1	0	1	$r_1 - a_{11} r_2$
1	0	−1	−3	2	0	0	r_2 / a_{21}
0	0	(4)	1	−1	1	(3)	$r_3 - a_{31} r_2$
0	0	(10)	6	−12	0	−24	$r_4 - a_{41} r_2$
↑ x_1							

0	1	0	(7/4)	−3/4	−1/4	(1/4)
1	0	0	−11/4	7/4	1/4	3/4
0	0	1	1/4	−1/4	1/4	3/4
0	0	0	(7/2)	−19/2	−10/4	−63/2
		↑ x_3				

$x_2 = 1/4$

$x_1 = 3/4$

$x_3 = 3/4$

is not solution

indicated by positive objective function coeff. left

0	4/7	0	1	−3/7	−1/7	1/7
1	11/7	0	0	4/7	−1/7	8/7
0	−1/7	1	0	−1/7	2/7	5/7
0	−2	0	0	−8	−2	−32

$x_2 = 0$

$x_1 = 8/7$

$x_3 = 5/7$

→ max = 32

```
01♦LBL "LPO"
02 CF 10
03 RCL 09
04 E3
05 /
06 RCL 09
07 +
08 GTO 00

09♦LBL "LPI"
10 "C↑Y↑B?"
11 PROMPT
12 SF 10
13 STO 07
14 RDN
15 1
16 +
17 STO 08
18 X<>Y
19 1
20 +
21 STO 09

22♦LBL 00
23 STO 01        ✓As 04
24 2
25 STO 00        ✓Put 02
26 RCL 08
27 STO 02        ✓As 05
28 RCL 07
29 "a"
30 CF 01
31 CF 04
32 FS?C 10       For LP/LPI S/R
33 GTO "IN"      ←
34 GTO "OUT"
```

'SIM' LINE 01

```
35♦LBL "LP"
36 CF 10
37 FS? 02
38 XEQ "TP"      T
39 RCL 07
40 DSE X
41 RCL 08
42 RCL 09
43 1
44 -
45 *
46 +
47 RCL X
48 E3
49 /
50 ST+ Z
51 +
52 RCL 08
53 1
54 -
55 E-3
56 *
57 ST+ Y
58 X<>Y
59 STO 12
60 LASTX
61 +
62 +
63 RCL 08
64 +
65 STO 14
66 1
67 +
68 XROM "BC"     Z
69 X<>Y
70 RCL 08
71 E5
72 /
73 +
74 STO 10
75 RCL 08
76 +
77 STO 11
78 ISG 12

79♦LBL 15
80 RCL 12
81 XROM "BX"     X
82 X<>Y
83 X<=0?
84 GTO 03
85 RCL [         00→16
86 INT
87 XROM "M4"     4
88 RCL 10
89 +
90 STO [         00→16
91 RCL 08
92 E3
93 /
94 +
95 STO \         01→17
96 RCL 11
97 STO ]         02→18
98 CLX
```

'SIM' LINE 125

```
99♦LBL 01
100 RCL IND [    00→16
101 X<=0?
102 GTO 00
103 RCL IND ]    02→18
104 X≠0?
105 GTO 05
106 STO ]        02→18
107 GTO 06

108♦LBL 05
109 /
110 X<Y?
111 GTO 00

112♦LBL 06
113 RCL [        00→16
114 STO 13

115♦LBL 00
116 X<>Y
117 ISG [        00→16
118 ""
119 ISG ]        02→18
120 GTO 01
121 1
122 X<> IND 13
123 1/X
124 RCL 13
125 INT
126 XROM "M4"    4
127 RCL 14
128 +
129 STO ]        02→18   SLACK VARIABLE TAKEOVER, SEE FLOWCHART
130 CLX
131 RCL IND ]    02→18
132 X≠0?
133 FS? 02
134 SF 05
135 RDN
136 STO 13
137 XROM "M2"    2
138 RCL 14
139 SIGN
140 RCL 13
141 GTO 00

142♦LBL 07
143 RCL IND L
144 X<>Y
145 X≠Y?
146 GTO 00
147 CLX
148 FS? 02
149 GTO 05
150 STO IND L
151 GTO 06

152♦LBL 00
153 ISG L
154 GTO 07

155♦LBL 06
156 RCL 13
```

'SIM' LINE 215

```
157♦LBL 05
158 FS?C 05
159 STO IND ]    02→18
160 1
161 STO ]        02→15

162♦LBL 02
163 RCL ]        02→18
164 RCL 13
165 X=Y?
166 GTO 00
167 0
168 X<> IND \    01→17
169 CHS
170 X≠0?
171 XROM "M3"    3

172♦LBL 00
173 ISG ]        02→18
174 ""
175 ISG \        01→17
176 GTO 02
177 GTO 15

178♦LBL 03
179 -1
180 RCL 09
181 XROM "M2"    2
182 FC? 02
183 RCL 12
184 FS? 02
185 RCL 11
186 XROM "BC"    Z
187 RCL 14
188 1
189 +
190 STO 13
191 DSE 12

×192♦LBL 04
193 RCL IND 13
194 X=0?
195 GTO 00
196 RCL 08
197 XROM "M5"    5
198 RCL 13
199 RCL 14
200 -
201 RCL 12
202 +
203 RCL IND Y
204 X<> IND Y
205 STO IND Z

206♦LBL 00
207 ISG 13
208 GTO 04
209 FS? 02
210 XEQ "TP"     T
211 END
329b=47R+0b
```

```
01♦LBL "TP"
02 RCL 07        StartAdr         17
03 ENTER↑
04 STO 01        ←07              17
05 STO 02        ←07              17
06 STO 04        ←07              17
07 RCL 09        +Rows+1          5
08 ST+ 04        ←As Int          22
09 +                              22
10 E3            For Isg.         .022
11 /
12 +                              17.022
13 RCL 08        Columns+1        5
14 ST- 04                         17
15 ST- 04                         12
16 STO 03        Col+1            5
17 RCL 09        Rows+1           5
18 *                              25
19 ST+ 04                         37
20 E6                             .000025
21 /
22 +                              17.022025
23 REGMOVE
24 ISG 04                         35

×25♦LBL 01       Skips By Force.
26 RCL 09        Rows+1           5
27 STO 00                         5
28 ST+ 02                         22
29 RCL 02                         22
30 RCL 03        Col+1            5

31♦LBL 02
32 RCL IND Y
33 STO IND 01
34 RDN
35 ISG 01
36 ""
37 +
38 LASTX
39 DSE 00
40 GTO 02
41 DSE 03
42 GTO 03
43 RCL 08
44 X<> 09
45 STO 08
46 RTN

47♦LBL 03
48 RCL 09
49 STO 00
50 RCL 04
51 STO 05
52 1
53 ST- 00
54 +
55 STO 06

56♦LBL 04
57 RCL 05
58 RCL 06
59 E3
60 /
61 +
62 RCL 03
63 E6
64 /
65 +
66 REGMOVE
67 DSE 05
68 RCL 03
69 ST- 05
70 ST- 06
71 DSE 00
72 GTO 04
73 2
74 ST+ 04
75 GTO 01
76 END
108b=15R+3b
```

R:07 Start Adr
R:08 Columns+1
R:09 Rows+1
R:10 Isg For Row
R:11 Isg Pointer For Constraint Constants
R:12 Isg Pointer For Objective Func Coefficients
R:13 Save Pivot Adr.
R:14 Isg Constant For Extra Row.

SYNTHETICS

P.J. Roussel's Version Of 'Sim' Called 'LP' For Linear Programming.

Non-Syntho' (RPN) version of 'LP', renamed 'SIM'

```
01♦LBL "SIM"
CLX  "MIN=1"  PROMPT
X≠0?  SF 02  FS? 02
XROM "T"  RCL 07  DSE X
RCL 08  RCL 09  1  -  *
+  RCL X  1 E3  /  ST+ Z
+  RCL 08  1  -  1 E-3
*  ST+ Y  X<>Y  STO 12
LASTX  +  +  RCL 08  +
STO 14  1  +  SIGN  CLX

40♦LBL 13
STO IND L  ISG L  GTO 13
X<>Y  RCL 08  1 E5  /  +
STO 10  RCL 03  +
STO 11  ISG 12

54♦LBL 15
RCL 12  STO 00  STO 01
STO 02  RCL IND X
ENTER↑  ENTER↑  RDN

63♦LBL 08
CLX  RCL IND Z  X>Y?
GTO 10  R↑  X>Y?  GTO 11
RDN

72♦LBL 09
ISG Z  GTO 08  X<>Y  R↑
GTO 14

78♦LBL 10
X<>Y  CLX  RCL Z  STO 00
GTO 09

84♦LBL 11
CLX  RCL T  STO 01  X<>Y
RDN  GTO 09

91♦LBL 14
X<>Y  X<=0?  GTO 03
RCL 00  INT  RCL 07  -
RCL 03  X<>Y  STO 02
X<>Y  MOD  ST- 02  LASTX
ST/ 02  CLX  X<> 02
X<>Y  ISG Y  " "  ISG X
" "  RCL 10  +  STO 00
RCL 08  1 E3  /  +
STO 01  RCL 11  STO 02
CLX

125♦LBL 01
RCL IND 00  X<=0?
GTO 00  RCL IND 02  X≠0?
GTO 05  STO 02  GTO 06

134♦LBL 05
/  X<Y?  GTO 00
```

```
138♦LBL 06
RCL 00  STO 13

141♦LBL 00
X<>Y  ISG 00  CLD
ISG 02  GTO 01  1
X<> IND 13  1/X  RCL 13
INT  RCL 07  -  RCL 08
X<>Y  STO 02  X<>Y  MOD
ST- 02  LASTX  ST/ 02
CLX  X<> 02  X<>Y  ISG Y
" "  ISG X  " "  RCL 14
+  STO 02  CLX
RCL IND 02  X≠0?  FS? 02
SF 05  RDN  STO 13
RCL 08  *  RCL 07  +
RCL X  RCL 08  ST- Z
SIGN  -  1 E3  /  +
X<>Y

192♦LBL 12
ST* IND Y  ISG Y  GTO 12
RCL 14  SIGN  RCL 13
GTO 00

200♦LBL 07
RCL IND L  X<>Y  X≠Y?
GTO 00  CLX  FS? 02
GTO 05  STO IND L
GTO 06

210♦LBL 00
ISG L  GTO 07

213♦LBL 06
RCL 13

215♦LBL 05
FS?C 05  STO IND 02  1
STO 02

220♦LBL 02
RCL 02  RCL 13  X=Y?
GTO 00  0  X<> IND 01
CHS  X≠0?  XROM "3"

230♦LBL 00
ISG 02  CLD  ISG 01
GTO 02  GTO 15

236♦LBL 03
-1  RCL 09  RCL 08  *
RCL 07  +  RCL X  RCL 08
ST- Z  SIGN  -  1 E3  /
+  X<>Y

252♦LBL 16
ST* IND Y  ISG Y  GTO 16
FC? 02  RCL 12  FS? 02
RCL 11  SIGN  CLX
```

```
262♦LBL 17
STO IND L  ISG L  GTO 17
RCL 14  1  +  STO 13
DSE 12

271♦LBL 04
RCL IND 13  X=0?  GTO 00
RCL 08  X<> 08  ST- 08
*  ST+ 08  X<> L  X<> 08
1  -  RCL 07  +  RCL 13
RCL 14  -  RCL 12  +
RCL IND Y  X<> IND Y
STO IND Z

294♦LBL 00
ISG 13  GTO 04  FS? 02
XROM "T"  TONE 9  GE
END
```

No Labels. See 'M3'

```
GTO "3"
08♦LBL "3"
09 STO 00
10 RDN
11 XEQ 00
12 X<>Y
13 XEQ 00
14 RCL 00
15 SIGN
16 RDN
17 RCL IND Y
18 LASTX
19 *
20 ST+ IND Y
21 ISG Y
22 " "
23 ISG Z
24 GTO 02
25 RTN
26 RCL 08
27 *
28 RCL 07
29 +
30 RCL X
31 RCL 03
32 ST- Z
33 SIGN
34 -
35 1 E3
36 /
37 +
38 RTN
39♦LBL "4"
```

```
ST/ND?  Nops.
E000EFFF
 E032>E087 86
 E090>E09F 16
 E0D8>E0DF 8
 E153>E18F 61
 E1A3>E1A7 5
 E1E7>E1EB 5
 E211>E217 7
 E24E>E25F 18
 E313>E32F 29
 E445>E44F 11
 E55F>E56F 17
 E5ED>E5EF 3
 E668>E66F 8
 E6A3>E6AF 13
 E764>E76F 12
 E783>E78F 13
 E945>EFFA 1718
                16
XR>16/24
E002/00/E08F SIMPLEX
E004/01/E0A2 ⊤M4
E005/02/E0C0 ⊤M5
E008/03/E262 ⊤MI
E00A/04/E299 ⊤MO  Syntho'
E00C/05/E2F9 ⊤VW  Section
E00E/06/E332 ⊤LP  Originals
E010/07/E192 ⊤BC
E012/08/E1AA ⊤BX
E014/09/E1EE ⊤M2
E016/10/E21A ⊤M3
E018/11/E0E2 ⊤TRNS
E01A/12/E452 ⊤Simplex.
E01C/13/E772 ⊤Z  Bc
E01E/14/E57A ⊤Trns
E020/15/E5F2 ⊤2  M2
E022/16/E602 ⊤3  M3
E024/17/E633 ⊤4  M4
E026/18/E651 ⊤5  M5
E028/19/E672 ⊤X  Bx
E02A/20/E6B2 ⊤SI  Input
E02C/21/E6E9 ⊤SO  Output
E02E/22/E74A ⊤V  ViewX,Y.
E030/23/E792 ⊤SIMplex.
```

Sim.Rpn ↑

No Synthetics, Straight Line Version, Except For 'T' & '3'

XROM 'T' Has 77 Lines + 2 Calls To X Functions 'Regmove' (XROM 25,35)