August 7, 2006

Balancing Matrixes with Externally Imposed Preconditions

Robert Michael Field

1. Introduction

A researcher who needs to balance a matrix often has data that are not part of the matrix itself, such as the value of—or part of the value of— particular cells, or the range within which the values must fall. The **precondition function** (which is part of the G7 software) allows the use of this data to constrain the value of cells, or groups of cells.

The function is invoked by the last parameter in the RAS, PSRAS and GVRAS commands. This parameter specifies a file (such as VA.pre) that contains a list of the cells that are to be constrained, the types of constraint to be applied, and the values to which they are to be constrained. These values can be expressed directly as a number, or as any expression that the G7 software can interpret, such as a series name from any bank or vam file.

2. Precondition Commands

There are two types of precondition. In the first type, a known value is removed from a cell, and from the corresponding row and column sums; the matrix is balanced; and finally the value is restored to the cell, row sum and column sum. In the second type, after each iteration, a value or condition is imposed until the matrix can be balanced without changing the value or violating the condition.

Precondition commands that apply to single cells

eq <rownum> <colnum> <value|expression>

This command forces the value of the specified cell to **equal** the value or expression. For example:

eq 7 12 127.5

forces the value of the 12th cell in the 7th row to equal 127.5
Or

eq 11 8 .5*c.VA2.3

forces the value of the 8th cell in the 11th row to equal half the value of the cell in matrix VA in vam file c.

This is done by (a) setting the specified cell to 0, (b) removing the value from both the row sum and the column sum, (c) balancing the matrix, and (d) restoring the value to the row and column sums and setting the cell equal to the value.

pt <rownum> <colnum> <value|expression>

This command preserves that **part** of the value of the specified cell to equal the value or expression. For example:

pt 7 12 51.3

This is done by (a) removing the value from the specified cell, the row sum, and the column sum, (b) balancing the matrix, and (c) restoring the value to the cell, the row sum and the column sum.

max <rownum> <colnum> <value|expression>

This command specifies the **maximum** value allowed in the specified cell.

min <rownum> <colnum> <value|expression>

This command specifies the **minimum** value allowed in the specified cell.

Precondition commands that apply to groups of cells

The following commands—which apply to part of a row, or part of a column, or a block of cells—scale the values in the cells indicated to the value or expression specified.

sc <firstrownum> <firstcolnum> <lastrownum> <lastcolnum> <value|expression>

This command **scales** the values in the cells specified to the value or expression specified. For example:

sc 3 1 3 5 c.GV3

The scales the values in the 1st through the 5th columns of the 3rd **row** to the value in the 3rd row of the vector GV in vam file c.
Or

sc 4 2 6 2 d.VA1

The values in the 4th through the 6th columns of the 2rd **column** are scaled to the value in the 1st row of the vector VA in vam file d.
Or

sc 2 2 4 6 1000

In the **block** that includes the 2nd through the 6th columns of the 2rd through the 4th rows, the values are scaled to 1000.

scmax <firstrownum> <firstcolnum> <lastrownum> <lastcolnum> <value|expression>

This command scales partial row, or partial column, or a block of cells if the sum of their values is greater than the value or expression specified.

scmin <firstrownum> <firstcolnum> <lastrownum> <lastcolnum> <value|expression>

This command scales a partial row, or a partial column, or a block of cells if the sum of their values is less than the value or expression specified.

3. A Demonstration

An examination of the G7 script "preconA.add" in the files that accompany this paper (found on p. 8 or in \precon\A) shows that it: (a) reads in the value added quadrant of a 3-sector input-output table; (b) balances the matrix without constraints—to serve as a basis of comparison; and (c) balances the matrix with a series of three constraints to show how they operate.

The first "show" presents the matrix balanced without constraints:

	VA	1	2	3	4
	2000	Primary	Secondary	Tertiary	FactorPay
1	EstDep	43.311	435.896	360.794	840.000
2	EstWages	1276.811	1808.285	1284.901	4370.000
3	EstTaxes	44.067	737.798	348.136	1130.000
4	EstSurplus	55.812	738.021	306.168	1100.000
5	ValueAdded	1420.000	3720.000	2300.000	7440.000

These results can be compared with the constrained results that follow.

The second show presents the matrix when it is balanced with the precondition in the file VA1.pre:

```
# Wages in secondary industry
eq  2 2  1800
```

This precondition forces the 2nd cell of the 2nd row (that is, wages in secondary industry) to equal 1800.000—which can be compared with the 1808.285 in the unconstrained solution, above.

The third show presents the matrix when it is balanced with the precondition in the file VA2.pre:

```
# Wages of cooperatives in tertiary industry
pt  2 3   600
```

This precondition preserves part of the value in the 3rd cell of the 2nd row (600) by removing it from the cell, the row total, and the column total before balancing the matrix and restoring it afterwards. The constrained value of the cell is 1237.485, as opposed to 1284.901 in the unconstrained result.

The fourth show presents the matrix when it is balanced with the precondition in the file VA3.pre:

```
# Scale total of depreciation in secondary
# and tertiary industry
sc  1 2  1 3  700
```

This precondition forces the sum of the values in the 2nd and 3rd cells of the 1st row to equal 700. It accomplished this by scaling the two cells to 700 after each iteration, until the matrix can be balanced without changing the value of their sum. The result can be verified by inspection that the values in the two cells add to 700.

4. Estimating Intermediate Input and Value Added for a 33-Sector Input-Output Table

A 1997 33-sector input-output table for Jiangxi Province is needed to use in a linked set of models for China's provinces, but only a 6-sector table and the intermediate input, final demand and gross output columns from a 38-sector table are available.[1] Fortunately, the figures in these two sources are consistent, and are most likely drawn from the 40-sector table prepared by the provincial statistical bureau.

The intermediate input and value added portion of the desired 33-sector table is shown on the following page with the data that can be transferred directly from the 6-sector or 38-sector sources. The key to estimating the values for the remaining cells is that every cell in the 6-sector table can be used as a control total for part of the table. For example, each shaded partial row or partial column and each unshaded block can be controlled to the value in a single cell in the 6-sector table.

[1] Both the table and the final demand data have been adjusted, but the adjustment is not relevant to the discussion of precondition commands in this paper.

		1	2							3	4	5	6						7
		Agri-cult-ure	Industry							Con-struc-tion	Trans & PTT	Com-merce	Other						Inter-medi-ate Input
		1	2	3	4		22	23	24	25	26	27	28	29	30	31	32	33	34
		Agri-cult-ure	Coal mng	Oil & Gas	FM & NF ore	---	Elec-tricity	Gas	Tap Water	Con-struc-tion	Trans & PTT	Com-merce	Fin & Ins	RE & Soc Ser	Health	Ed & Cult	Sci Res	Public Adm	Inter-medi-ate Input
1	Agriculture	111.8								0.0	2.7	17.4							315.2
2	Coal ming																		51.8
3	Oil & Gas																		25.6
4	FM&NF ore																		53.3
	----																		----
22	Electricity																		96.2
23	Gas																		1.1
24	Tap Water																		12.2
25	Construction	0.0								0.2	0.7	1.4							24.5
26	Trans&PTT	16.2								6.2	5.8	34.3							181.3
27	Commerce	16.0								0.7	3.7	8.3							162.4
28	Fin & Ins																		98.5
29	RE& SocSer																		86.9
30	Health																		10.7
31	Ed & Cult																		19.1
32	Sci Res																		13.3
33	Public Adm																		29.4
34	Depreciation	11.0								11.1	26.0	9.7							221.8
35	Wages	407.2								82.1	55.5	63.8							1070.6
36	Net Taxes	13.9								15.8	6.1	14.3							186.8
37	Surplus	38.7								1.7	30.3	15.9							235.0
38	Gross Output	760.5	59.7	0.0	49.1	---	79.7	7.8	14.0	316.5	225.0	275.7	93.4	165.6	72.5	1.1	38.5	0.1	4006.1

As an illustration, the values in the 2nd through the 24th cells of the 1st row of the 33-sector table can be controlled to the value in the 2nd cell of the 1st row of the 6-sector table. The precondition command is, as follows:

sc 1 2 1 24 d.jxiIO6sec1.2

which instructs G7 to scale the values in the cells indicated to the value found in the cell indicated in the matrix "jxiIO6sec" in vam file d.

Or the command:

sc 2 1 24 1 d.jxiIO6sec2.1

instructs G7 to scale the values in the 2nd through the 24th cells of the 1st column to the value found in "jxiIO6sec2.1" in vam file d.

Or finally, the command:

sc 2 2 24 24 d.jxiIO6sec2.2

instructs G7 to scale the values in the 2nd row of the 2nd column through the 24th row of the 24th column to the value found in "jxiIO6sec2.2" in vam file d.

Note that the complete kist of precondition commands in found in the file "\precon\B\jxiIO33sec.pre".

To see how this procedure works out in practice, one should run the file "preconB.add", which is found on p. 9 or in the directory "\precon\B". The first show allows the examination of the 6-sector input-output table, and the second, an examination of the intermediate input, final demand and gross output columns from a 38-sector table.

The G7 script then distributes the data from the 6-sector table to the 33-sector table and displays the result—which is rather sparse. It includes only inputs into agriculture, construction, transport and commerce, and the details of value added for the sectors agriculture, transport, and commerce, and total inputs. The script then adds gross value and intermediate input control totals for the industrial and other tertiary sectors and displays the result.

Finally, initial values for the blank cells are taken from the input-output table for all China, and the table is balanced in 14 iterations. It is only the complex pattern of constraints imposed by the precondition commands, that were discussed above, that permits reasonable estimates for the missing data.

4. FURTHER WORK

This method of using constraints while balancing a matrix will be used to control the 1997 59-sector input-output for all China to the level of the reported national accounts, and to prepare input-output tables for several other provinces. It will also be used to expand the currently available 40-sector make table to 59 sectors.

The greatest difficulty balancing a matrix with a large number of constraints is to write the precondition command file. In this moderate size example, the preconditions file was 82 lines long (see p.11). Because the individual lines vary only in the rows and columns to which they refer, it was extremely difficult to eliminate replication of lines or entry of incorrect row and column numbers. It would greatly facilitate the application of the these commands in balancing large tables if a program could be written that used the double do loops that are available in G7 scripts to prepare the command file.

Appendix B: preconA.add

```
# preconA.add

zap
close all
clear

vamcr VA.cfg VA
vam VA c
dvam c

# Read in the raw data
matin  rawdat  1997  1  5  1  4  12
#             Primary Secondary  Tertiary ValueAdded
Dep            58.50    563.70    409.00   840.00
Wages        1297.90   1759.90   1096.20  4370.00
Taxes          43.30    694.10    287.10  1130.00
Surplus        74.50    943.20    343.00  1100.00
ValueAdded   1420.00   3720.00   2300.00  7440.00

# Balance the matrix without constraints
mcopy VA = rawdat
ras  VA (r 1-4)(c 1-3)  VA 4  VA 5 1997
show VA y 1997

# Balance the matrix with the constraint found in the file VA1.pre
mcopy VA = rawdat
ras  VA (r 1-4)(c 1-3)  VA 4  VA 5 1997 -VA1.pre
show VA y 1997

# Balance the matrix with the partial constraint found in the file
VA2.pre
mcopy VA = rawdat
ras  VA (r 1-4)(c 1-3)  VA 4  VA 5 1997 -VA2.pre
show VA y 1997

# Balance the matrix with the constraint found in the file VA3.pre
mcopy VA = rawdat
ras  VA (r 1-4)(c 1-3)  VA 4  VA 5 1997 -VA3.pre
show VA y 1997
```

Appendix B: preconB.add

```
# jxiIO33sec.add
# To build a 33-sector IO table for Jiangxi

zap
clear
close all

yf = 4
fdates 1997 1997

vamcr jxiIO33sec.cfg jxiIO33sec
vam jxiIO33sec c
dvam c

vam rawdata d
# Show the 6-sector I-O table
show d.jxiIO6sec y 1997
# Show the 33-sector final demand matrix
show d.jxiFD38sec y 1997

# 1. Move data from the 6-sector I-O table to the 33-sector table
addty n
do{
  do{
    vf jxiIO33sec%1.%3 = d.jxiIO6sec%2.%4
    }%1 %2 (1 25-27 34)(1 3-5 7)m #Target and source columns

  }(1 25-27 34-38)(1 3-5 8-11 13)m #Target and source rows
addty y
# Show the 33-sector table with data and control totals from the 6-
sector table
show c.jxiIO33sec y 1997

# 2. Move intermediate input amd gross output for industry
# and other tertiary from 38-sector final demand
addty n
do{
  # Intermediate input
  vf jxiIO33sec%1.34 = d.jxiFD38sec%2.1
  # Gross otuptut
  vf jxiIO33sec38.%1 = d.jxiFD38sec%2.8
  }(2-20 22-24 28 30-33)(2-20 23-25 32 35-38)m #Target and source rows
addty y

# Manufacturing, nec
vf jxiIO33sec21.34 = d.jxiFD38sec21.1 + d.jxiFD38sec22.1
vf jxiIO33sec38.21 = d.jxiFD38sec21.8 + d.jxiFD38sec22.8

# Real Estate and Social Services
vf jxiIO33sec29.34 = d.jxiFD38sec33.1 + d.jxiFD38sec34.1
vf jxiIO33sec38.29 = d.jxiFD38sec33.8 + d.jxiFD38sec34.8
# Show the 33-sector table with all available data and control totals
show c.jxiIO33sec y 1997
```

```
# 3. Enter and scale all intermediate input and value added by column
addty n
do{
  do{
    vf jxiIO33sec%1.%3 = d.AllChina%2.%3 * (jxiIO33sec38.%3 /
d.AllChina40.%3)
    }%1 %2 (1-33)
  }(1-37)(1-33 35-38)m
addty y

# 4. Ras intermediate input and value added
#ras <matrix>[(r rgroup)][(c cgroup)]<row><col>[yr][r|c][-maxiter][-
precon]
psras jxiIO33sec (r 1-37)(c 1-33) jxiIO33sec 34 jxiIO33sec 38 1997
-jxiIO33sec.pre
# Show the 33-sector table that is balanced with constraints from the
6-sector table
show c.jxiIO33sec y 1997
```

Appendix C: jxiIO33sec.pre

```
# jxiIO33sec.pre

# Single cells in intermediate input
eq  1  1  d.jxiIO6sec1.1
eq 25  1  d.jxiIO6sec3.1
eq 26  1  d.jxiIO6sec4.1
eq 27  1  d.jxiIO6sec5.1

eq  1 25  d.jxiIO6sec1.3
eq 25 25  d.jxiIO6sec3.3
eq 26 25  d.jxiIO6sec4.3
eq 27 25  d.jxiIO6sec5.3

eq  1 26  d.jxiIO6sec1.4
eq 25 26  d.jxiIO6sec3.4
eq 26 26  d.jxiIO6sec4.4
eq 27 26  d.jxiIO6sec5.4

eq  1 27  d.jxiIO6sec1.5
eq 25 27  d.jxiIO6sec3.5
eq 26 27  d.jxiIO6sec4.5
eq 27 27  d.jxiIO6sec5.5

# Scale rows in intermediate input
sc  1 2   1 24  d.jxiIO6sec1.2
sc 25 2  25 24  d.jxiIO6sec3.2
sc 26 2  26 24  d.jxiIO6sec4.2
sc 27 2  27 24  d.jxiIO6sec5.2

sc  1 28   1 33  d.jxiIO6sec1.6
sc 25 28  25 33  d.jxiIO6sec3.6
sc 26 28  26 33  d.jxiIO6sec4.6
sc 27 28  27 33  d.jxiIO6sec5.6

# Scale columns in intermediate input
sc  2  1  24  1  d.jxiIO6sec2.1
sc  2 25  24 25  d.jxiIO6sec2.3
sc  2 26  24 26  d.jxiIO6sec2.4
sc  2 27  24 27  d.jxiIO6sec2.5

sc 28  1  33  1  d.jxiIO6sec6.1
sc 28 25  33 25  d.jxiIO6sec6.3
sc 28 26  33 26  d.jxiIO6sec6.4
sc 28 27  33 27  d.jxiIO6sec6.5

# Scale blocks in intermediate input
sc  2  2  24  24  d.jxiIO6sec2.2
sc 28  2  33  24  d.jxiIO6sec6.2
sc  2 28  24  33  d.jxiIO6sec2.6
sc 28 28  33  33  d.jxiIO6sec6.6
```

```
# Single cells in value added
eq 34  1  d.jxiIO6sec8.1
eq 35  1  d.jxiIO6sec9.1
eq 36  1  d.jxiIO6sec10.1
eq 37  1  d.jxiIO6sec11.1

eq 34 25  d.jxiIO6sec8.3
eq 35 25  d.jxiIO6sec9.3
eq 36 25  d.jxiIO6sec10.3
eq 37 25  d.jxiIO6sec11.3

eq 34 26  d.jxiIO6sec8.4
eq 35 26  d.jxiIO6sec9.4
eq 36 26  d.jxiIO6sec10.4
eq 37 26  d.jxiIO6sec11.4

eq 34 27  d.jxiIO6sec8.5
eq 35 27  d.jxiIO6sec9.5
eq 36 27  d.jxiIO6sec10.5
eq 37 27  d.jxiIO6sec11.5

# Scale rows in value added
sc 34  2  34 24  d.jxiIO6sec8.2
sc 35  2  35 24  d.jxiIO6sec9.2
sc 36  2  36 24  d.jxiIO6sec10.2
sc 37  2  37 24  d.jxiIO6sec11.2

sc 34 28  34  33  d.jxiIO6sec8.6
sc 35 28  35  33  d.jxiIO6sec9.6
sc 36 28  36  33  d.jxiIO6sec10.6
sc 37 28  37  33  d.jxiIO6sec11.6
```