The RAS Principle

1 Introduction

1.1 The purpose of this note is to describe the RAS principle - a method of controlling matrix cells to desired row and column totals.

1.2 The (integer-valued) desired row and column totals must be pre-specified and must sum to the same value. Initial cell values must also be assigned, but these need not sum to the desired totals. The initial cell values should be integer valued.

1.3 The RAS principle consists of essentially two steps: the first of which involves multiplying the initial cell values by row and column factors, as described in section 2; and the second step, described in section 3, involves controlling any residual difference that might remain after the first step.

2 Step 1 - Multiplicative Action

2.1 First of all, select either rows or columns first, according to whichever is least likely to be well reflected by the initial cell values. The following description assumes that columns are selected.

2.2 For each column, calculate a **Column factor = (desired column total)/(actual column total)**, and multiply each cell in the column by the column factor to amend the cell values so that they now approximately agree with the desired column total. Cell values should be rounded to the nearest integer after being multiplied by column factors.

2.3 Sum the rows to obtain new row totals and calculate row factors: **Row factor = (desired row total)/(actual row total).**

2.4 Multiply each cell in each row by the appropriate row factor to amend the cell values so that they now approximately agree with the desired row totals. Cell values should be rounded to the nearest integer after being multiplied by row factors.

2.5 On repeating this iterative process, i.e. on multiplying all matrix cells by column factors then by row factors then by column factors etc., a number of times, the matrix will converge toward stability.

2.6 After application of column factors, the row values should be summed to see how close they are to the desired totals. Providing all row totals are either within 1 or 1% of the desired total, ‘row convergence’ is said to have been achieved. Similarly after application of row factors test for ‘column convergence’.

2.7 On achieving row or column convergence, the multiplicative action should cease. In case of non-convergence, the number of iterations is limited to 10.

2.8 Once row or column convergence has been achieved, or ten iterations have been performed, any remaining residual difference should be controlled away as described in the following section.

3 Step 2 - Residual Controlling

3.1 For each row, calculate a row difference (**RD**) by subtracting the sum of the actual cell values in that row from the desired row total. Similarly, for each column, calculate a column difference (**CD**).

Stage (i)

Consider first the negative differences.
Select at random a column **Y** with a negative difference and in a similar way randomly select a row **X** with a negative difference. Locate the cell at their intersection and consider its value **CV(X,Y).**

If **CD(Y)** is the column difference and **RD(X)** is the row difference,

let **ADJ(X,Y) = MAX(CD(Y),RD(X))**,
Or equivalently let **ADJ(X,Y) = -MIN(ABS(CD(X),ABS(RD(Y))).**

Let **ADJ1(X,Y) = MIN(CV(X,Y), ABS(ADJ(X,Y))),**
which ensures that the following adjustment cannot result in a negative cell value.

Adjust cell value	**CV(X,Y)** = **CV(X,Y)** - **ADJ1(X,Y).**
Column difference	**CD(Y)** = **CD(Y)** + **ADJ1(X,Y)**.
Row difference	**RD(X)** = **RD(X)** + **ADJ1(X,Y)**.

Repeat stage (i) unless (a) all **CD(Y) >= 0** or (b) all **RD(X) >= 0** or (c) the values at the intersections of negative rows and columns are all zero.

If the reason for ceasing repetition is (c) then special action is required, see stage (iv). The likelihood of this happening is small but action should be taken before dealing with the positive differences.

Stage (ii)

Now consider the positive differences.
Find the column **Y** with the greatest positive difference and similarly the row **X** with the greatest positive difference. Locate the cell at their intersection and its value **CV(X,Y)**. If **CD(Y)** is the column difference and **RD(X)** is the row difference,

let **ADJ(X,Y) = MIN (CD(Y),RD(X)).**

Adjust cell value	**CV(X,Y)** = **CV(X,Y)** + **ADJ(X,Y)** and
Column difference	**CD(Y)** = **CD(Y)** - **ADJ(X,Y)** and
Row difference	**RD(X)** = **RD(X)** - **ADJ(X,Y)**

Repeat stage (ii) unless (a) all **CD(Y) = 0** or (b) all **RD(X) = 0**.

Stage (iii)

After stage (ii) has been completed some differences may still remain e.g. + N in one column and -N in another, or similarly for rows or even spread about over rows or columns. Now there is no obvious cell to adjust, e.g. if it is columns with non-zero differences we add one to a cell in a + column and subtract one from a cell in the same row of a - column. The row may be chosen at random but checks to ensure that the cell affected by subtraction is non-zero should be made. Adjust the column differences and repeat until all column differences are zero. An exactly analogous procedure can be followed if the original differences were in rows rather than columns.

After this process the matrix should be fully converged.

Stage (iv)

This process is needed to cope with the unlikely event of having negative row and column differences but zero cells at their intersection after stage (i).

Locate a column **Y** with negative difference **CD(Y)**. Choose a row **X** at random and consider its row difference **RD(X**). If **RD(X)** < 0 generate another random row. If **RD(X)>=0** locate the cell value at intersection **CV(X,Y)** and test **CV(X,Y)**. If **CV(X,Y)=0** generate another random row. If **CV(X,Y)>0** subtract one from **CV(X,Y)** and add one to **CD(Y)** and **RD(X**).

Repeat until all **CD(Y)>=0.**

Perform an exactly analogous calculation on rows remaining negative. With rows and columns interchanged, repeating until all **RD(X)>=0.**