ACTA UNIVERSITATIS AGRICULTURAE ET SILVICULTURAE MENDELIANAE BRUNENSIS

Volume LX 11 Number 4, 2012

USING EXCEL TO REDUCE A SQUARE MATRIX

J. Holoubek, P. Zach

Received: January 18, 2012

Abstract

HOLOUBEK, J., ZACH, P.: *Using Excel to reduce a Square Matrix.* Acta univ. agric. et silvic. Mendel. Brun., 2012, LX, No. 4, pp. 109–114

When solving operations research problems, one can use either specialised computer programs such as Lingo, Lindo, Storm or more universal programs such Excel, Matlab, and R. To obtain the input data, one can use either a program's own editor or other programs commonly available such as Excel. While the problem-solving methods, being part of various programs, are the subjects of numerous publications (such as Gros, 2003; Jablonský, 2002; Plevný – Žižka, 2007; Stevenson – Ozgur, 2009), the way the input data are obtained, recorded, and processed receives far less attention although this part of problem-solving requires considerable effort and, if the method for data recording is inadequate, may cause subsequent difficulties in their further processing. A problem known as "the travelling salesman problem" (TSP) may serve as an example. Here, the input data form a "square matrix of distances". This paper is concerned with some Excel tools that can be used to obtain and subsequently modify such a square matrix. Given a square $m \times m$ matrix, an ordinary user might want to reduce it to an $i \times i$ square matrix (where $i < m$) without having to copy data from the matrix, skip some of its rows and/or columns or write a program to implement such a reduction.
In her degree project, Kourková, 2009 was looking for an efficient method of reducing an Excel matrix. She had found no relevant papers on this subject concluding that the authors of the commercial program had not considered this. Therefore, she offered her own solution unconventionally using the contingency table menu option. Although this had resulted in the desired submatrix, some of its parts were superfluous and even baffling for the user.
For this reason, the authors analyse the method of representing an $m \times m$ matrix and the way of its reduction. Finally, a better option is offered to achieve the desired objective as well as other methods of obtaining the required submatrix that even users without sufficient programming skills can use.

operations research, travelling salesman problem, Excel, matrix reduction

Operations research as a collection of mathematically heterogeneous disciplines is used to solve various problems. Although being of diverse nature, such problems have still some features in common. This is, above all, the fact that a mathematical (or graphical) model is used to approach such problems as well as that the quantitative characteristics always play a substantial role in their solution. To collect and record such input data necessary to solve a given problem requires some effort, which increases the likelihood of various errors. One of the typical problems in operations research is the TSP. The present paper is concerned with the collection and further processing of the input data necessary to solve such a problem.

MATERIAL AND METHODS

As the TSP is generally well-known, let us just observe that the problem of a travelling salesman is to find the optimum arrangement of a tour of places and to know in which order he should visit all the places of his business to eventually return to the starting point. There are many variations to this basic problem. The criterion used to judge the quality of a route is usually its length, the travel time, and travel costs. There are numerous methods for solving this problem, but this paper will not deal with them. Generally, it is true that the time required to solve the problem grows very rapidly with the number of places to be visited. Devlin, 2005 characterizes this dependency as a factorial growth.

For all the methods and computer programs used to solve the TSP knowledge of the input data is essential. Such data may be obtained by various methods such as using an on-line route planner or GPS. The data thus collected are then arranged in a square $m \times m$ matrix showing $m - 1$ places to be visited and one (the first or the last one) place where the tour begin and end. The matrix itself containing $m \times m - m$ entries may be symmetric or non-symmetric. When repeatedly planning a distribution or collection route, it may happen that not all the places considered in the $m \times m$ matrix need to be visited in a particular case. Then only the currently relevant entries must be extracted from the original matrix. Thus, of all the m places, we only plan to visit some (say i) so that the $m \times m$ matrix must be reduced to an $i \times i$ square submatrix.

There is only slight attention paid in the literature to the problem of collecting, recording, presenting and, subsequently, processing such entries always assuming that they are available in required arrangement and quantity. (Practical experience shows that this is not always true. The TSP is often solved only in an intuitive manner and that is why the input data have to be collected first.) Various computer programs can be used to obtain the $m \times m$ matrix.

The Excel spreadsheet program seems to be the commonest tool available for such a task. This is also the reason why Kourková used it in her degree project in 2009. She obtained the necessary 10,600 entries of the original 103×103 input matrix relating to all the customers of a company using an on-line route planner. Except that creating such a matrix is tedious and error prone, using the program chosen to record the data seems to be reasonable. The original matrix then had to be used to produce the required submatrices of various sizes. Their sizes and contents had to be modified as required. Since the method for reducing the original matrix had to be available even for an ordinary user not supposed to be versed in programming, the question had to be resolved what possibilities are offered by Excel in this regard. (In the sequel, Excel will always mean MS Excel 2003.) After considering all the tools offered by the spreadsheet program, one could see that this operation had not been foreseen. Therefore, non-traditionally using one of the Excel menus designed to create contingency table and graphs proved a practicable way out of this difficulty. The method chosen is described clearly and in detail on pages 65 plus of the degree project. The result of the reduction is the required square submatrix containing all the entries necessary to solve a particular problem. However, this method has a defect as the resulting submatrix contains a superfluous heading in its first column (as a remnant of the contingency table), which is baffling. Considering the size of the original matrix (103×103), all the subsequent demos will only contain its small sections. Figs. 1 and 2 show the appearance of the original matrix entries and the submatrix created. There was a natural question whether the required modifications of the original matrix could be done using Excel without containing the superfluous heading or what other methods could be used to simply reduce the matrix.

RESULTS AND DISCUSSION

After analyzing the method of matrix reduction designed by Kourková, we arrived at a conclusion that the main difficulty in reducing the matrix was the method chosen for obtaining the input data, that is, the matrix. With this in mind, we could design three different methods of reducing the original matrix, which, in addition to discarding the undesired heading, were more efficient:

1. Use the method designed by Kourková and repair the above defect through minor changes as the standard Excel menus do not allow for any other way of reducing the size of the matrix.
2. Reduce the original matrix to a required submatrix by writing a Submatrix (in Czech Submatice) macro.

	A	J	K	L	M	N	O
1	Město	České Budějovice	Český Těšín	Děčín	Dobrá Voda	Dobruška	Domažlice
10	České Budějovice	0,0	405,8	262,3	4,2	284,9	140,4
11	Český Těšín	405,8	0,0	428,7	403,3	258,3	547,0
12	Děčín	262,3	428,7	0,0	262,0	200,5	263,4
13	Dobrá Voda	4,2	403,3	262,0	0,0	288,1	143,8
14	Dobruška	284,9	258,3	200,5	288,1	0,0	301,3
15	Domažlice	140,4	547,0	263,4	143,8	301,3	0,0

1: *A section of the original* $\boldsymbol{m \times m}$ *matrix of distances*

3. Obtain the input data by a method quite different from that of using a matrix as before. A table used for storing data such as one in a relational database seems to be a tool appropriate to our purposes. To data modified in this way, one can subsequently apply a previously written macro called Create_submatrix (in Czech Tvorba_submatice) to create the original matrix 103×103. This matrix will subsequently serve as input to a Submatrix macro creating in a new worksheet a reduced matrix in a form that can be used for solving the TSP (both macros written to reduce the original matrix along with demos are available at: https://akela.mendelu.cz/~xzach/emm/submatice.xls).

It is clear that, without some programming skills, the possibilities of reducing the matrix size are very limited.

Let us now look at the ways of reducing the matrix in more detail.

When designing and describing the first method on pp. 65 to 68, even the author knew that using a contingency table for these purposes was non-standard as, on page 69, she says: „It is clear that the contingency table was not originally designed for these purposes but rather to visualize the relationship of two statistical attributes. The row and column sums represent the number of occurrences of an event regardless of the other event. For this reason, the rows of a contingency table contain the header *Sum of (*in Czech *Součet z)*." This header in column one of a contingency table (undesired for the purpose of reducing the matrix) (see Fig. 2) can be removed by modifying each cell. However, since no two cells of a table may have an identical town denomination (as it was possible in the original matrix see Fig. 1), to distinguish them, we will use the prefix „from" (in Czech „z") for towns of column one and the prefix „to" (in Czech „do") for towns in row two of the contingency table – see Fig. 3 (macros could also be written to carry out these modifications rather than performing them manually).

The second method of reducing the original square matrix to a submatrix also uses the fact that the original data source is an $m \times m$ matrix. Two conditions must be fulfilled before launching a Submatrix macro: a subset of towns must be defined for the new submatrix and a cell must be marked the first element of the set of towns from which the subset is chosen.

This means that, in the original matrix, we first highlight in boldface those towns in column one that are to be contained in the submatrix. This fulfils the first condition.

4	Data	České Budějovice	Český Těšín	Děčín
13	Součet z České Budějovice	0	405,8	262,3
14	Součet z Český Těšín	405,8	0	428,7
15	Součet z Děčín	262,3	428,7	0

2: *A section of the contingency table with undesired headers*

4	Data	do_České Budějovice	do_Český Těšín	do_Děčín
13	z_České Budějovice	0	405,8	262,3
14	z_Český Těšín	405,8	0	428,7
15	z_Děčín	262,3	428,7	0

3: *The section after text modification*

	A	J	K	L	M	N	O
	Submatice						
1	Město	České Budějovice	Český Těšín	Děčín	Dobrá Voda	Dobruška	Domažlice
10	České Budějovice	0,0	405,8	262,3	4,2	284,9	140,4
11	Český Těšín	405,8	0,0	428,7	403,3	258,3	547,0
12	Děčín	262,3	428,7	0,0	262,0	200,5	263,4
13	**Dobrá Voda**	4,2	403,3	262,0	0,0	288,1	143,8
14	Dobruška	284,9	258,3	200,5	288,1	0,0	301,3
15	Domažlice	140,4	547,0	263,4	143,8	301,3	0,0

4: *The original matrix with a button for launching a Submatrix macro*

Next, the first cell in this column has to be clicked with a town regardless of whether it is highlighted or not. As can be seen in Fig. 4, the list of all towns is contained in column A, the first town is in cell A10 and the first town for us to choose (boldface) in cell A13 (Dobrá Voda), we need to click cell A10. This fulfils the second condition.

All we have to do now is launch the Submatrix macro. This may be done in two different ways. The first, more complex one, is to launch it from the system menu (Tools – Macro – Macros – Submatrix – Launch). The second, more user-friendly one, is to use a button with the same label placed directly in the Distance Matrix Excel demo worksheet, which can be found in the file at the above-mentioned URL. This button may be copied at will between new worksheets in this file. Both ways have the same effect: creating in a new worksheet a new square submatrix containing towns in column A of the original matrix highlighted in boldface before the macro is launched. Thus, using this macro is very useful and, provided that both conditions are fulfilled before the macro is launched, the desired submatrix is obtained in a new worksheet.

The algorithm used by the Submatrix macro can be divided into three phases. In phase one, the list of towns in the original matrix is scanned until the end of the list is reached (an empty cell) while creating a list of the future towns. Here is a simplified pseudo-code:

```
WHILE the cell is non-empty DO
   IF the cell is highlighted THEN
      add a new town to the submatrix;
      remember the new count of towns in
      the submatrix;
   END IF
   go to the next cell of the list;
LOOP
```

In the second phase, a new worksheet is created in the workbook with matrix headers, that is, horizontal and vertical town names as headers of the submatrix. This is described by the following simplified pseudo-code:

```
Create and rename a new worksheet;
go to a new worksheet;
FOR i = 1 TO number of the new towns DO
   insert town into the cell with
   coordinates (i + 1, 1);
   insert town into the cell with
   coordinates (1, i + 1);
NEXT
```

In the third phase, the relevant distances are copied. The following simplified pseudo-code describes this. Note that the macro can create a submatrix from both a symmetric and non-symmetric original matrix of distances:

```
FOR y = 1 TO number of new towns DO
   FOR x = 1 TO number of new towns DO
      To cell (x + header, y + header)
      in the new worksheet assign the
      corresponding value from the
      original matrix;
   NEXT
NEXT
```

When launched, this macro creates the desired submatrix on a new Excel worksheet. If, for instance, we want to create from the original 103×103 matrix a submatrix with 10 towns (Děčín, Havířov, Hlučín, Holešov, Cheb, Kolín, Měřín, Most, Písek, Tábor), we obtain the matrix shown by Fig. 5.

The third method of reducing the original matrix is based on the idea that storing the input data as a square $m \times m$ matrix is not the best way if such data are to be further processed. Therefore, it is assumed that the original data are stored in a table. In this table, each entry of the $m \times m$ matrix will be defined by the following three atributes: START| KM |DESTINATION (in Czech CÍL) placed in a single

	A	B	C
1	Město	Km	Cíl
2	Bernartice	0,0	Bernartice
3	Bernartice	384,5	Bohumín
4	Bernartice	194,0	Brno - Židenice
5	Bernartice	320,8	Bruntál
6	Bernartice	243,1	Břeclav
7	Bernartice	46,1	Březnice
8	Bernartice	216,8	Česká Lípa

6: *Data represented by a table*

	A	B	C	D	E	F	G	H	I	J	K
1		Děčín	Havířov	Hlučín	Holešov	Cheb	Kolín	Měřín	Most	Písek	Tábor
2	Děčín	0	424,3	415,8	393,5	191,1	164,6	252,5	69,7	216,3	201,4
3	Havířov	424,3	0	27,5	112,3	572,7	309,5	259,1	499,6	407,1	359
4	Hlučín	415,8	27,5	0	92,8	553,3	290	239,6	480,2	387,7	339,5
5	Holešov	393,5	112,3	92,8	0	456,7	239,9	143,1	383,7	291,1	243
6	Cheb	191,1	572,7	553,3	456,7	0	250,9	320,3	123,4	192,3	222
7	Kolín	164,6	309,5	290	239,9	250,9	0	99,9	156,9	174,9	126,5
8	Měřín	252,5	259,1	239,6	143,1	320,3	99,9	0	243,1	150,6	102,5
9	Most	69,7	499,6	480,2	383,7	123,4	156,9	243,1	0	189,5	190
10	Písek	216,3	407,1	387,7	291,1	192,3	174,9	150,6	189,5	0	46,1
11	Tábor	201,4	359	339,5	243	222	126,5	102,5	190	46,1	0

5: *The resulting matrix created by the Submatrix macro*

row of the table as shown in Fig. 6. First, a Create_Matrix macro will be applied to this table containing all the data of the original $m \times m$ matrix. This will transform the table into an original matrix sized 103×103 in our case.

We will use a demo example from the Table worksheet in the demo file at https://akela.mendelu.cz/~xzach/emm/submatice.xls to describe and demonstrate the third method. The Table worksheet contains demo data in the form as shown by Fig. 6. Next, a Create_Matrix will be applied to this table, which can be launched in two different ways: using the standard Excel menu (Tools – Macro – Macros – Create_Matrix – Run) or by pressing a button displayed directly in the worksheet labelled with the macro name. Compared with the Submatrix macro, only one condition must be fulfilled before running this macro (Create_Matrix), that is, the first cell must be clicked in the Town column containing a particular value (town). The macro will then create a new worksheet with an original $m \times m$ matrix of distances, which can then be processed by the Submatrix macro launched as described above selecting various submatrices at will.

The algorithm of the Create_Matrix macro can be divided into two phases. The first phase creates a new worksheet as described above. The second phase then creates a distance matrix as described by the simplified pseudo-code below:

```
WHILE the cell is non-empty DO
   remember the current town;
   WHILE the current town is not
   different DO
      copy the matrix entries;
   LOOP
LOOP
```

The data stored in this way in a relational database can be imported to Excel by two different methods. The first method involves the use of a CSV file. This is, however, limited by a maximum number of rows in a worksheet, that is, 65 536. The second method uses a direct link between Excel and a relational database. It is, however, not the purpose of this paper to deal with these issues in detail. For more information, see the MySQL Manual (2010) and a paper available at Wall.cz (2010).

SUMMARY

In this paper, the authors wanted to judge the suitability and efficiency of methods of obtaining, presenting, and further processing the input data of the TSP using Excel as designed by Kourková, improve this method in a suitable way and offer a different approach to processing such data.

In the first part, the problem was how Excel can be used to reduce an original square $m \times m$ matrix to a square $i \times i$ submatrix with $i < m$. The method designed by Kourková made a non-traditional use of an Excel option designed for different purposes – for creating contingency table. A drawback of this method was a superfluous header remaining from the contingency table.

When looking for ways of creating the necessary submatrix without the unwanted header, the authors arrived at the following three solutions:

1. Use the solution designed by Kourková and, in the resulting $i \times i$ submatrix, carry out partial modifications to repair the above defect.
2. To create an $i \times i$ submatrix use a Submatrix macro applying it to the original $m \times m$ matrix.
3. If the distances are stored in a relational database as entities in a table as described above, use macro a Create_Matrix to transform the data into a form to which, subsequently, the Submatrix macro can easily be applied to create $i \times i$ submatrices at will.

There are certainly other methods that could use other tools (both commercial and non-commercial[1]) and the user's knowledge to create the desired submatrix efficiently. However, this paper is concerned with those possibilities that users with only a basic computer background can employ using commonly available tools. The solutions offered fulfil the purpose and, therefore, can find their practical applications.

REFERENCES

DEVLIN, K., 2005: *Problémy pro třetí tisíciletí: Sedm největších nevyřešených otázek matematiky*. 1. vyd. Praha: Dokořán, s. r. o., 269 p. ISBN 80-7363-016-8.

GROS, I., 2003: *Kvantitativní metody v manažerském rozhodování*. 1. vyd. Praha: Grada, 2003. 432 p. ISBN 80-245-0162-7.

JABLONSKÝ, J., 2007: *Operační výzkum*. 3. vyd. Praha: Professional publishing, 323 p. ISBN 978-80-86946-44-3.

1 For example: http://www.or.deis.unibo.it/research_pages/tspsoft.html

KOURKOVÁ, E., 2009: *Využití metod lineárního programování při řešení dopravních úloh*. Brno, 93 p. Diplomová práce. Mendelova zemědělská a lesnická univerzita v Brně. Available at: <https://is.mendelu.cz/lide/clovek.l?zalozka=7;id=14101;studium=30746;download_prace=1>.

PLEVNÝ, M., ŽIŽKA, M., 2007: *Modelování a optimalizace v manažerském rozhodování*. 1. vyd. Plzeň: Západočeská univerzita, 296 p. ISBN 978-80-7043-435-2.

STEVENSON, W. J., OZGUR, C., 2007: *Introduction to management science with spreadsheets*. Boston: McGraw-Hill/Irwin, 812 p. ISBN 978-0-07-299066-9.

MySQL Manual [online]. 2010 [cit. 2010-11-18]. MySQL:: MySQL 5.1 Reference Manual:: 21.1 MySQL Connector/ODBC. Available at: <http://dev.mysql.com/doc/refman/5.1/en/connector-odbc.html>.

WALL.CZ [online]. 2010 [cit. 2010-11-18]. Kontingenční tabulka - externí data MySQL. Available at: <http://wall.cz/kontingencni-tabulka-externi-data-mysql.a136.html>.

Address

doc. Ing. Josef Holoubek, CSc., Bc. Petr Zach, Ústav statistiky a operačního výzkumu, Mendelova univerzita v Brně, Zemědělská 1, 613 00 Brno, Česká republika, e-mail: josef.holoubek@mendelu.cz, petr.zach@mendelu.cz