Measurement efficiency and productivity in SAS/OR

Ali Emrouznejad

Statistics and Operational Research, School of Mathematical and Information Sciences, Coventry University, Priory Street, Coventry CV1 5FB, UK

Abstract

This paper explores the use of the optimisation procedures in SAS/OR software with application to the measurement of efficiency and productivity of decision-making units (DMUs) using data envelopment analysis (DEA) techniques. DEA was originally introduced by Charnes et al. [J. Oper. Res. 2 (1978) 429] is a linear programming method for assessing the efficiency and productivity of DMUs. Over the last two decades, DEA has gained considerable attention as a managerial tool for measuring performance of organisations and it has widely been used for assessing the efficiency of public and private sectors such as banks, airlines, hospitals, universities and manufactures. As a result, new applications with more variables and more complicated models are being introduced.

Further to successive development of DEA a non-parametric productivity measure, Malmquist index, has been introduced by Fare et al. [J. Prod. Anal. 3 (1992) 85]. Employing Malmquist index, productivity growth can be decomposed into technical change and efficiency change.

On the other hand, the SAS is a powerful software and it is capable of running various optimisation problems such as linear programming with all types of constraints. To facilitate the use of DEA and Malmquist index by SAS users, a SAS/MALM code was implemented in the SAS programming language. The SAS macro developed in this paper selects the chosen variables from a SAS data file and constructs sets of linear-programming models based on the selected DEA. An example is given to illustrate how one could use the code to measure the efficiency and productivity of organisations.

© 2003 Elsevier Ltd. All rights reserved.

Keywords: SAS; SAS/OR; DEA software; Linear programming; Malmquist index; Efficiency and productivity

1. Introduction

Data envelopment analysis (DEA) is a linear programming method for assessing the efficiency and productivity of decision-making units (DMUs) (see [1,2] for details). DEA continues to grow in importance as managerial tools become more reliable and handle performance measurement of
organisations. As a result, new applications with more variables and more complicated models are being introduced (see DEA models in [3]).

On the other hand, SAS is recognised as one of the lead packages for statistical analysis and as a powerful tool for data base systems in many organisations, both in public and private sectors. SAS users come from every major industry (banking to pharmaceuticals, manufacturing to telecommunications, etc.) all with the same basic needs to make better strategic decisions and to gain a competitive edge.

There are many applications in SAS that the users recognised as powerful tools in organisational management. For example, the SAS/OR System has numerous optimisation procedures which handle the standard problems such as linear and non-linear programming with all types of constraints [4]. These capabilities are exposed to the SAS users in a variety of places such as in LP and NLP procedures [5].

In addition to the standard procedures available in SAS System applications such as neural network, simulation [6] and control project management [7] are introduced. A SAS/DEA macro was introduced by Emrouznejad [8] but the macro is not able to handle the productivity measure such as Malmquist index.

This paper aims to introduce a new application in SAS System for measuring efficiency and productivity of DMUs using DEA and Malmquist index techniques.

The paper unfolds as follows. The Sections 2 & 3 describes the DEA and Malmquist models and method of calculations of efficiency and productivity measures. Section 4 illustrates features that have been added to the SAS System, call it SAS/MALM. These include procedures for data handling, model building and reporting the results. Finally, this paper uses a small data set to show some of the features of the SAS/MALM system.

2. About DEA and Malmquist index

DMUs are units of organisations such as banks, universities, and hospitals, which typically perform the same function. A DMU usually uses a set of inputs (resources) to secure a set of outputs (products) (Models 1–3).

Model 1. Output oriented—CRS model

Max  \( h \)

s.t.

\[
\sum_j \lambda_j x_{ij} + S_i^+ = x_{ij0} \quad \forall i,
\]

\[
\sum_j \lambda_j y_{rj} - S_r^- = h y_{rj0} \quad \forall r,
\]

\( S_i^+, S_r^- \geq 0 \) \quad \forall i, \forall r,

\( \lambda_j \geq 0 \) \quad \forall j,
where $x_{ij}$ the amount of the $i$th input at DMU $j$, $y_{rj}$ the amount of the $r$th output from DMU $j$ and $j_0$ the DMU to be assessed.

Model 2. Input oriented—CRS model

Min $\phi$

s.t.

$$\sum_j \lambda_j x_{ij} + S_i^+ = \phi x_{i0} \quad \forall i,$$

$$\sum_j \lambda_j y_{rj} - S_r^- = y_{r0} \quad \forall r,$$

$S_i^+, S_r^- \geq 0$ \quad $\forall i, \forall r$,

$\lambda_j \geq 0$ \quad $\forall j$,

where $x_{ij}$ the amount of the $i$th input at DMU $j$, $y_{rj}$ the amount of the $r$th output from DMU $j$ and $j_0$ the DMU to be assessed.

Model 3. Linear programming models for calculation of the Malmquist index and its components

$$[D_t'(x_t, y_t)]^{-1} = \text{Min} \; \phi$$

s.t.

$$\sum_j \lambda_j x_{ij} \leq \phi x_{i0} \quad \forall i$$

$$\lambda_j \geq 0 \quad \forall j$$

$$[D_{t+1}'(x_{t+1}, y_{t+1})]^{-1} = \text{Min} \; \phi$$

s.t.

$$\sum_j \lambda_j x_{ij} \leq \phi x_{i0} \quad \forall i$$

$$\sum_j \lambda_j y_{rj} \geq y_{r0} \quad \forall r$$

$$\lambda_j \geq 0 \quad \forall j$$

$$[D_{t+1}'(x_t, y_t)]^{-1} = \text{Min} \; \phi$$

s.t.

$$\sum_j \lambda_j x_{ij} \leq \phi x_{i0} \quad \forall i$$

$$\lambda_j \geq 0 \quad \forall j$$

$$[D_t'(x_{t+1}, y_{t+1})]^{-1} = \text{Min} \; \phi$$

s.t.

$$\sum_j \lambda_j x_{ij} \leq \phi x_{i0} \quad \forall i$$

$$\sum_j \lambda_j y_{rj} \geq y_{r0} \quad \forall r$$

$$\lambda_j \geq 0 \quad \forall j$$
DEA is a method for measuring efficiency of DMUs using linear programming techniques to “envelop” observed input–output vectors as tightly as possible [9]. One main advantage of DEA is that it allows several inputs and outputs to be considered at the same time. In this case, efficiency is measured in terms of inputs or outputs along a ray from the origin. For more information about DEA and its recent development see Emrouznejad [3], Emrouznejad [13] and Thanassoulis [10].

Further, Malmquist index [11] is a productivity measure that can be decomposed to technical and efficiency change.

3. DEA Software

4. SAS/MALM

The SAS/MALM introduced in this paper provides a powerful management tool for assessing both efficiency and productivity of organisations in SAS system. The program can handle both input minimisation and output maximisation. Further, it can calculate the input and output Malmquist index and its components. To enhance the model there are several parameters. The user can select the desired parameters according to the particular model that is required. Users familiar with SAS can add their own features to build other DEA models. Users not familiar with SAS need only to run the program with their model specification prior to running the system.

The SAS/MALM requires two initial data sets that contain the name of variable and data file for observed units. The data describing inputs/outputs must be presented in the format in which variables appear in columns and units in rows and saved as SAS data set. SAS has the ability also to read from a text (Tab delimited) or Exel format. The program has the ability to accommodate unlimited number of inputs/outputs with unlimited number of DMUs. The only limitation is the memory of computer used to run the SAS/MALM.

The SAS/MALM software then converts data sets to a selected DEA model. Based on the data and parameters specified in the SAS/MALM, the code first creates the usual linear program, then use “Proc LP” to solve the model. The results will then be transferred to report files.

The SAS/MALM produces a table of efficiencies of DMUs. It also supplies much other valuable information including lambda and slack values in primal and weights in dual DEA models. These information are very useful for analysing the inefficient units, where the source of inefficiency comes from and how to improve an inefficient unit to the desired level.

In the rest of this paper the procedure of implementation of SAS/MALM together with an example are explained.

5. Definition of terms and typographical conventions

In the rest of this paper and particularly in the SAS/MALM code we will see several types of styles used. Style conventions are summarised below:

Courier font: is used to show example of SAS statements. In most cases, this paper uses lowercase type for SAS code. The user can enter own SAS code in lowercase, uppercase or a mixture of the two. Enter any titles and footnotes exactly as you want them to appear on the printout.
Underscore: Variable name that are surround by “_” are specifically used as parameters to the SAS. In all case these variables must be used without any change.

Underscore: Variable name that are started with “_” are specifically used as parameters to the SAS/MALM.

The SAS/MALM runs three sections for data handling (%data1 and %data2), model building (%duality and %dea) and report writing (%report).

6. Illustration of SAS/MALM

This section presents a simple example of two inputs two outputs and 6 DMUs for illustration of SAS/MALM. This example was taken from Sexton [12, p. 19], and will allow us to compare the efficiency scores obtained from SAS/MALM with that of reported by Sexton. He used this example for assessing six nursing homes. A DEA model with 2 inputs and 2 outputs were used.

The inputs are:
- Staff hours per day (StHr), including nurses, physician, therapists, and so on.
- Suppliers per day (Supp) which are measured in thousands of dollars.

The outputs are:
- Total Medicare plus Medicaid-reimbursed patient days (MCPD),
- Total privately paid patient days (PPPD).

The inputs are measured on a daily basis, while the outputs are annual totals. It is possible to express all inputs and outputs in the same units, but this is not necessary, since the basic DEA model assumes constant returns to scale. For more information see Sexton [12, pp. 19–23].

To allow measuring the productivity indices I extended this example with artificial values for another year.

7. Data handling (%data1 and %data2)

This part of SAS/MALM reformats the data to a suitable form that can be used in SAS/OR. The SAS/MALM requires two data sets including the name of variable and the data file. In data file, variables must be presented in columns and units in rows. The unit names must start with a letter and may contain up to 50 characters and must be listed in the first column of the data file. Period should be recorded in the second column of the data file. The other columns are including numeric values of variables. These variables can be entered into the file in any order. An example of data file is as follows:

File: datafile.sd2

<table>
<thead>
<tr>
<th>DMU</th>
<th>Period</th>
<th>StHr</th>
<th>Supp</th>
<th>MCPD</th>
<th>PPPD</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1985</td>
<td>150</td>
<td>0.2</td>
<td>14000</td>
<td>3500</td>
</tr>
<tr>
<td>B</td>
<td>1985</td>
<td>400</td>
<td>0.7</td>
<td>14000</td>
<td>21000</td>
</tr>
<tr>
<td>C</td>
<td>1985</td>
<td>320</td>
<td>1.2</td>
<td>42000</td>
<td>10500</td>
</tr>
<tr>
<td>D</td>
<td>1985</td>
<td>520</td>
<td>2.0</td>
<td>28000</td>
<td>42000</td>
</tr>
<tr>
<td>E</td>
<td>1985</td>
<td>350</td>
<td>1.2</td>
<td>19000</td>
<td>25000</td>
</tr>
</tbody>
</table>
The variable name file should have two columns. The first column contains the name of variables and the second column contains type of variable (input or output). An example of variable name file is as follows:

File: varname.sd2

<table>
<thead>
<tr>
<th>VARIABLE</th>
<th>IOTYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>StHr</td>
<td>INPUT</td>
</tr>
<tr>
<td>Supp</td>
<td>INPUT</td>
</tr>
<tr>
<td>MCPD</td>
<td>OUTPUT</td>
</tr>
<tr>
<td>PPPD</td>
<td>OUTPUT</td>
</tr>
</tbody>
</table>

There are six parameters prior to calling the data macro:
- _LibName: name of directory where all files were saved,
- _DataF: name of data file,
- _VarF: name of variable file,
- _Orienta: to select the orientation of the model, it can be “OUTPUTMAX” or “INPUTMIN”,
- _Period1: first period for Malmquist index and
- _Period2: second period for Malmquist index.

SAS procedure for data handling is presented in Appendix A.1.

8. Model building (%Duality and %DEA)

This part of SAS/MALM builds the requested DEA model in the format suitable for SAS/OR and for purpose of calling “Proc LP”. There is one parameter prior calling the procedures:
- _Orienta

For example, for solving a standard input minimisation model the user should set the parameters to:
- _Orienta=‘INPUTMIN’;

and for solving output maximisation model the user should set the parameters to:
- _Orienta=‘OUTPUTMAX’;

SAS procedure for model building is presented in Appendix A.2.

9. SAS/MALM Efficiency report writing (%report)

The SAS/MALM results are including table of efficiencies of DMUs in two formats sorted by the most efficient DMU and sorted alphabetically by the name of DMUs. All other information like
slacks and lambdas are saved in file called “Report1”. All information on the dual to each model including weights are saved in file called “Report2”.
SAS procedure for report writing is presented in Appendix A.3.

10. SAS/MALM macro

To make the system as easy as possible the “%sasmalm macro” put all the above code together.

```
%macro sasmalm;
   libname sasmalm &libname;
   %data1;
   proc datasets nolist library=sasmalm;
      delete MRP MRD;
      run;
   %do j=1 %to &nDmu;
     %do M=1 %to 8;
       %let Uj0=DMU&j;
       %data2(&Uj0);
       %DEA(&Uj0, &M);
     %end;
   %end;
   %report;
%mend sasmalm;
```

In the above code the “%sasmalm macro” is used to manage all previously explained codes including data handling, model building and report writing. To get the result, user needs to set up the parameters and run only one statement:

```
%sasmalm;
```

11. Results of SAS/MALM

This section uses an example for assessing six nursing homes over two years as described earlier. The data and the variable name are saved in “datafile.sd2” and “varname.sd2” respectively.
A user needs to set the parameters as required and run the following code for input orientation model.

```
* Parameter definition;
%let _libname=‘c:\sasmalm’; * Name of directory;
%let _dataF=sasmalm.DataFile; * Name of data file;
%let _varF=sasmalm.VarName; * Name of variable file;
%let _Orienta=‘INPUTMIN’; * Setting orientation of the model;
```
12. Sample DEA results from SAS/MALM

The results of running the above code is presented below. The third and fourth columns show the efficiency score in each year.

<table>
<thead>
<tr>
<th>OBS</th>
<th>DMU</th>
<th>EFF1985</th>
<th>EFF1986</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>1.0000</td>
<td>0.7720</td>
</tr>
<tr>
<td>4</td>
<td>D</td>
<td>1.0000</td>
<td>0.7682</td>
</tr>
<tr>
<td>5</td>
<td>E</td>
<td>0.9775</td>
<td>1.0000</td>
</tr>
<tr>
<td>6</td>
<td>F</td>
<td>0.8674</td>
<td>0.5557</td>
</tr>
</tbody>
</table>

This table shows, in 1985, units E and F are inefficient units, all other units are efficient. This is exactly matched with the results reported by Sexton [12, Table 5, p. 22]. Unit F is less efficient than unit E. The best that unit F can do is 0.8674, while the maximum of unit E is 0.9775.

SAS/MALM also creates a wealth of information and saved them in two files, report1 and report2. By using these in SAS system we can specify a management strategy indicating by how much each unit should reduce each input and/or increase each output in order to become perfectly efficient. To see this we need the value of slacks and lambdas in primal models (1 or 2) and value of weights in dual to each. These all are saved in files report1 and report2 respectively. It is important to use these reports effectively to derive the measure required for managerial purpose.

For example the lambdas’ value for unit E can be obtained from report1 by using fields .var_ and .value_. It can be seen that the value of lambdas for unit E, in 1985, are as follows:

<table>
<thead>
<tr>
<th>OBS</th>
<th>.ROW_ID_</th>
<th>.VALUE_</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LA A</td>
<td>0.20000</td>
</tr>
<tr>
<td>2</td>
<td>LA B</td>
<td>0.08048</td>
</tr>
<tr>
<td>3</td>
<td>LA C</td>
<td>0.00000</td>
</tr>
<tr>
<td>4</td>
<td>LA D</td>
<td>0.53833</td>
</tr>
<tr>
<td>5</td>
<td>LA E</td>
<td>0.00000</td>
</tr>
<tr>
<td>6</td>
<td>LA F</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

What these values suggest is that the target for unit E has an input output levels that are 0.2 times those for unit A plus 0.8048 times those for unit B plus 0.53833 those for unit D. Therefore, the
optimum value of input output variables for unit E are:

<table>
<thead>
<tr>
<th>StHr</th>
<th>Supp</th>
<th>MCPD</th>
<th>PPPD</th>
</tr>
</thead>
<tbody>
<tr>
<td>342</td>
<td>1.17</td>
<td>19000</td>
<td>25000</td>
</tr>
</tbody>
</table>

Thus unit E should try to reduce its input levels by 2.3% without reducing either output levels. (Note that, 342 equals 97.7% of 350 and 1.17 equals 97.7% of 1.2).

13. Sample Malmquist index results from SAS/MALM

The SAS/MALM code also gives the details of Malmquist index and its components. For example the results obtained from the above example are as follows:

The SAS System

<table>
<thead>
<tr>
<th>OBS</th>
<th>DMU</th>
<th>EFF85</th>
<th>Eff86</th>
<th>Tech85</th>
<th>Tech85</th>
<th>Tech85</th>
<th>Tech85</th>
<th>Tech85</th>
<th>Tech85</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>In86</td>
<td>In86</td>
<td>Out86</td>
<td>Out86</td>
<td>Out86</td>
<td>Out86</td>
<td>Out86</td>
<td>Out86</td>
</tr>
<tr>
<td>1</td>
<td>A</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>1.00000</td>
<td>0.77200</td>
<td>1.00000</td>
<td>1.00000</td>
<td>0.84437</td>
<td>0.81060</td>
<td>0.88659</td>
<td>1.00000</td>
</tr>
<tr>
<td>4</td>
<td>D</td>
<td>1.00000</td>
<td>0.76821</td>
<td>1.00000</td>
<td>0.85854</td>
<td>0.79228</td>
<td>1.00000</td>
<td>1.00000</td>
<td>0.89289</td>
</tr>
<tr>
<td>5</td>
<td>E</td>
<td>0.97750</td>
<td>1.00000</td>
<td>1.00000</td>
<td>0.97750</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>6</td>
<td>F</td>
<td>0.86745</td>
<td>0.55571</td>
<td>0.70119</td>
<td>0.72312</td>
<td>0.68072</td>
<td>0.67751</td>
<td>0.82993</td>
<td>0.58452</td>
</tr>
</tbody>
</table>

The third and fourth columns in these tables reported the efficiencies of each DMU respectively for two consecutive years 1985 and 1986. The last 6 columns are showing the component of Malmquist index with different disaggregation. For example column “Tech85 In86 0ut86” indicates the component of Malmquist index where we use technology in 1985; i.e. inputs and outputs of all DMUs come from 1985, but input and output of DMU under assessment come from 1986. Using this component various type of Malmquist indexes can easily be calculated. For example the following report shows the disaggregation of Malmquist index to efficiency change (EFFCH) and technical change (TECHCH).

The SAS System

<table>
<thead>
<tr>
<th>OBS</th>
<th>DMU</th>
<th>EFFCH</th>
<th>TECHCH</th>
<th>MALM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>0.77200</td>
<td>1.20874</td>
<td>0.93314</td>
</tr>
<tr>
<td>4</td>
<td>D</td>
<td>0.76821</td>
<td>1.07810</td>
<td>0.82821</td>
</tr>
<tr>
<td>5</td>
<td>E</td>
<td>1.02302</td>
<td>0.98869</td>
<td>1.01144</td>
</tr>
<tr>
<td>6</td>
<td>F</td>
<td>0.64062</td>
<td>1.04853</td>
<td>0.67171</td>
</tr>
</tbody>
</table>

As it can be seen Malmquist index = Technical Change × Efficiency Change.
14. Conclusion

Today many organisations recognise SAS as one of the lead packages for data base system and statistical analysis. In particular, optimisation procedures in SAS/OR are exposed to the user in a variety of places such as “Proc LP” and “Proc NLP”. Therefore, many applications such as neural network and control project management are introduced. SAS/MALM as introduced in this paper is a new application in SAS/OR that is a powerful managerial tool for measuring the efficiency and productivity of decision making units.

The SAS/MALM application implemented in this paper has no limitation on the input and output variables or the number of DMUs. The only limitation is the memory and disk space of the computer uses. It is flexible enough to add a other DEA model.

A SAS/MALM for end user is as easy as to run one statement:

```
%SASmalm.
```

The SAS/MALM report’s files can directly feed to other SAS routines for further analysis. Fig. 1

Appendix A

A.1. SAS procedure for data handling

```
* Macro for transfering data files to suitable format for DEA models;
%Macro data1;
  data _TypeF;
  format _type_ $3.;
  format IOTYPE $7.;
  _type_='LE'; IOTYPE='OUTPUT'; output;
```
_type_='LE'; IOType='INPUT'; output;
IOType='OBJ'; output;
run;
data &_VarF;
set &_varF;
Variable=upcase(variable);
run;
proc sort
data=&_VarF(where=(UPCASE(IOType)='INPUT'))
out=_VarFi;
by Variable;
run;
data _null_; set _VarFi;
call symput('_nInput','_'); run;
data _VarFi;
do i= 1 to &_nInput;
   Link Read;
   _VarName=compress('X'||i); output;
end;
Read: set _VarFi;return;
drop i;
run;
proc sort
data=&_VarF(where=(UPCASE(IOType)='OUTPUT'))
out=_VarFo;
by Variable;
run;
data _null_; set _VarFo;
call symput('_nOutput','_'); run;
data _VarFo;
do r= 1 to &_nOutput;
Link Read;
_VarName=compress('Y'||r); output;
end;
Read: set _VarFo;return;
drop r;
run;
proc sort data=&_DataF
out=&_DataF;
by DMU;
run;
proc summary data=&_DataF;
output out=_Dmu1F(keep=DMU);
by DMU;
run;
data _null_; 
set _Dmu1F;
call symput('nDmu',_n_);
run;
data _DmuF;
do j=1 to &nDmu;
   Link Read;
   _VarName=compress('DMU'||j);output;
end;
Read: set _Dmu1F;return;
drop j;
run;
data _DataF1;
merge &_DataF _DmuF;
by DMU;
run;
data _VarF1;
set _VarFi _VarFo;
run;
proc sort data=_VarF1;
by variable;
run;
data _ObjF; _VarName='TETA';
Variable='Objective';
IOType='OBJ';
if &Orienta='INPUTMIN' 
   then do;
      _type_='MIN';
   end;
if &Orienta='OUTPUTMAX' 
   then do;
      _type_='MAX';
   end;
output;
RUN;
data _D&Per1.;
set _DataF1(where=(Period=&Per1));
run;
data _D&Per2.;
set _DataF1(where=(Period=&Per2));
run;
data _TypeF;
set _TypeF;
IOTYPE=UPCASE(IOTYPE);
run;
%do t=&Per1 %to &Per2;
proc transpose data=_D&t.(drop=DMU)
out=_D&t.t(rename=Name=Variable);
  id _VarName;run;
  proc sort data=_D&t.t;
by variable;
run;
data _VarF2;
merge _VarF1(in=IO) _D&t.t (in=Dt);
by Variable; if Dt and IO;
run;
  proc sort data=_VarF2;
by IOType;
run;
proc sort data=_TypeF;
by IOType;
run;
data _F4&t.;
  merge _VarF2(in=Dt) _TypeF;
  by IOType;
  Period=&t.;
  if Dt;
  run;
data _F4&t.;
  set _F4&t.;
  if UpCase(IOTYPE)='OUTPUT' then do;
    %do j0=1 %to &_nDmu;
    DMU&j0=- DMU&j0;
    %end;
  end;
end;
r
%end;
data Model01;
set _F4&Per1.;
_Kind='T1xt1yt1';
run;
data Model02;
set _F4&Per2.;
_Kind='T2xt2yt2';
run;
%mend data1;
* Macro for transferring data files to suitable format for Malmquist index;
%macro data2(Uj0);
  data _Model0x(keep=_varname &Uj0
  rename=(&Uj0=TempUj0));
  set _F4&_Per2.;
  run;
  data _Model0y; merge _Model0x _F4&_Per1.;
  by _VarName;
  run;
  data _Model03(drop=TEMPUj0);
  set _Model0y;
  if UPCASE(IOTYPE)='INPUT' then &Uj0=TEMPUj0;
  _Kind='T1xt2yt1';
  run;
  data _Model0x(keep=_varname &Uj0
  rename=(&Uj0=TempUj0));
  set _F4&_Per2.;
  run;
  data _Model0y; merge _Model0x _F4&_Per1.;
  by _VarName;
  run;
  data _Model04(drop=TEMPUj0);
  set _Model0y;
  if UPCASE(IOTYPE)='OUTPUT' then &Uj0=TEMPUj0;
  _Kind='T2xt1yt2';
  run;
  data _Model0x(keep=_varname &Uj0
  rename=(&Uj0=TempUj0));
  set _F4&_Per1.;
  run;
  data _Model0y; merge _Model0x _F4&_Per2.;
  by _VarName;
  run;
  data _Model05(drop=TEMPUj0);
  set _Model0y;
  if UPCASE(IOTYPE)='INPUT' then &Uj0=TEMPUj0;
  _Kind='T2xt1yt2';
  run;
  data _Model0x(keep=_varname &Uj0
  rename=(&Uj0=TempUj0));
  set _F4&_Per1.;
  run;
  data _Model0y; merge _Model0x _F4&_Per2.;
  by _VarName;
run;
data _Model06(drop=TEMPUj0);
set _Model0y;
if UPCASE(IOTYPE)='OUTPUT' then &Uj0=TEMPUj0;
_KIND='T2xt2yt1';
run;
data _Model0x(keep=_varname &Uj0 rename=(&Uj0=TempUj0));
set F4&Per1.;
run;
data _Model0y;
merge _Model0x _F4&Per2.;
by VarName;
run;
data _Model07(drop=TEMPUj0);
set _Model0y;
&Uj0=TEMPUj0;
_KIND='T2xt1yt1';
run;
data _Model0x(keep=_varname &Uj0 rename=(&Uj0=TempUj0));
set _F4&Per2.;
run;
data _Model0y;
merge _Model0x _F4&Per1.;
by VarName;
run;
data _Model08(drop=TEMPUj0);
set _Model0y;
&Uj0=TEMPUj0;
_KIND='T1xt2yt2';
run;
%mend data2;

A.2. SAS procedure for model building

* Macro for preparation dual model;
%macro Duality(PrimF, DualF, Ind);
proc sort data=&PrimF out=Temp1;
by _row_;
run;
proc transpose data =Temp1 out=Temp2;
id _row_;
run;
data &DualF(rename=(Teta=_RHS_ _name_=._row_));
set Temp2;
format _type_ $3.;
if _name_ = ‘_RHS_’ then _name_ = ‘TETA’;
if Teta=. then Teta=0; else Teta= - Teta;
if substr(_name_,1,2) = ‘FI’
   then _type_ = ‘EQ’;
   else _type_ = ‘GE’;
if upcase(_name_) = ‘TETA’ then _type_ = ‘MIN’;
run;
%Mend Duality;

* Macro for preparation DEA models;
%macro DEA(DMUj0, M);
data _ModelP1(rename=(_VarName=row_) drop=IOTYPE PERIOD variable);
   set _Model0&M;
   if &_orienta = ‘INPUTMIN’ then do;
      if UPCASE(IOType) = ‘INPUT’
         then do;
            FI = - &DMUj0;
            _RHS_ = 0;
         end;
      if UPCASE(IOType) = ‘OUTPUT’
         then do;
            FI = 0;
            _RHS_ = &DMUj0;
         end;
   end;
   if &_orienta = ‘OUTPUTMAX’ then do;
      if UPCASE(IOType) = ‘OUTPUT’
         then do;
            FI = - &DMUj0;
            _RHS_ = 0;
         end;
      if UPCASE(IOType) = ‘INPUT’
         then do;
            FI = 0;
            _RHS_ = &DMUj0;
         end;
   end;
run;
%Mend Duality(_ModelP1, _ModelW1,’LE’);
data _ModelR1;
set _ModelW1;
if _row_ = ‘FI’
   then do;
if &_Orienta='INPUTMIN' then _rhs=- 1;
if &_Orienta='OUTPUTMAX' then _rhs=1;
end;
run;
proc LP data= _Modelr1
primalout=RP dualout=RD MAXIT2=100000;
run;
data RP;
set RP;
_DMUj0='&DMUj0';
_KIND=&M.;
run;
data RD;
set RD;
_DMUj0='&DMUj0';
_KIND=&M.;
run;
proc datasets nolist force;
append base=sasmalm.MRP data=RP;
run;
proc datasets nolist force;
append base=sasmalm.MRD
data=RD;
run;
%mend dea;

A.3. SAS procedure for report writing

* Macro for reporting the results;
%macro report;
data _KindF;
kind=1; _tKind='T1xt1yt1'; output;
kind=2; _tKind='T2xt2yt2'; output;
kind=3; _tKind='T1xt2yt1'; output;
kind=4; _tKind='T1xt1yt2'; output;
kind=5; _tKind='T2xt1yt2'; output;
kind=6; _tKind='T2xt2yt1'; output;
kind=7; _tKind='T2xt1yt1'; output;
kind=8; _tKind='T1xt2yt2'; output;
run;
proc sort
data=sasmlm.mrp(rename=(Dmuj0=_VarName));
by _VarName;
run;
proc sort data=_DmuF;
by _VarName;
run;
data report1(keep=DMU _value_ _kind);
merge _DmuF sasmalm.mrp(where=( _var_='TETA'));
by _VarName;
run;
data report1(drop=_value_);
set report1;
if &_Orienta='INPUTMIN' then eff=- _value_;
if &_Orienta='OUTPUTMAX' then eff=1/(_value_);
run;
proc sort data=report1;
by _Kind;
run;
data report1(drop=_kind);
merge report1 _KindF;
by _kind;
run;
proc sort data=report1;
by DMU;
run;
proc transpose
data=report1 out=sasmalm.report(drop=name_);
by DMU;
id _tkind;
run;
%mend report;

A.4. SAS/MALM macro

* SAS/MALM macro for calculating and reporting DEA and Malmquist index;
%macro sasmalm;
libname sasmalm &_libname;
%data1;
proc datasets nolist library=sasmalm;
delete MRP MRD;
run;
%do j_Note=&nDmu %to 1 %by - 1;
%do M=1 %to 8;
%let Uj0=DMU&j_Note;
%data2(&Uj0);
%DEA(&Uj0, &M);
%end;
%end;
%report;
%mend sasmalm;

* Parameter definitions and running SAS/MALM;
%let _libname='c:\sasmalm'; * Name of directory;
References