The Stata Journal (2010)
10, Number 2, pp. 267–280

Data envelopment analysis

Yong-bae Ji
Korea National Defense University
Seoul, Republic of Korea
jyb77@hanmail.net

Choonjoo Lee
Korea National Defense University
Seoul, Republic of Korea
sarang90@kndu.ac.kr

Abstract. In this article, we introduce a user-written data envelopment analysis command for Stata. Data envelopment analysis is a linear programming method for assessing the efficiency and productivity of units called decision-making units. Over the last decades, data envelopment analysis has gained considerable attention as a managerial tool for measuring performance of organizations, and it has been used widely for assessing the efficiency of public and private sectors such as banks, airlines, hospitals, universities, defense firms, and manufacturers. The `dea` command in Stata will allow users to conduct the standard optimization procedure and extended managerial analysis. The `dea` command developed in this article selects the chosen variables from a Stata data file and constructs a linear programming model based on the selected `dea` options. Examples are given to illustrate how one could use the code to measure the efficiency of decision-making units.

Keywords: st0193, dea, data envelopment analysis, linear programming, nonparametric, efficiency, decision-making units

1 Introduction

In this article, we introduce a new application in Stata for performance measurement of decision-making units (DMUs) using data envelopment analysis (DEA) techniques. DEA is a nonparametric linear programming method for assessing the efficiency and productivity of DMUs. DEA application areas have grown since it was first introduced as a managerial and performance measurement tool in the late 1970s. Since then, new applications with more variables and complicated models have been and are being introduced.

Stata equipped with the `dea` command will provide the user with a new nonparametric tool to analyze productivity data. From within Stata, users will be able to produce DEA scores and analyze them. Because second-stage DEA analysis and DEA efficiency estimates involve statistical inference, DEA users need a software package that can analyze the whole process in one system. The alternative is juggling between a DEA command and a statistical software that uses the DEA scores as dependent variables to find the influential variables in second-stage DEA analysis.

The main purpose of this article is to implement the `dea` command in Stata. The article unfolds as follows. The next section describes the DEA models and calculations in DEA. The remainder of this article illustrates the features and options of the `dea` command.

2 The basics of DEA

DEA is a method for measuring efficiency of DMUs using linear programming techniques to envelop observed input–output vectors as tightly as possible (Boussofiane, Dyson, and Thanassoulis 1991). DEA allows multiple inputs–outputs to be considered at the same time without any assumption on data distribution. In each case, efficiency is measured in terms of a proportional change in inputs or outputs. A DEA model can be subdivided into an input-oriented model, which minimizes inputs while satisfying at least the given output levels, and an output-oriented model, which maximizes outputs without requiring more of any observed input values.

DEA models can also be subdivided in terms of returns to scale by adding weight constraints. Charnes, Cooper, and Rhodes (1978) originally proposed the efficiency measurement of the DMUs for constant returns to scale (CRS), where all DMUs are operating at their optimal scale. Later Banker, Charnes, and Cooper (1984) introduced the variable returns to scale (VRS) efficiency measurement model, allowing the breakdown of efficiency into technical and scale efficiencies in DEA.

In figure 1, frontiers determined by economies of scale are presented considering one input and one output for 5 DMUs labeled A through E. The CRS, VRS, and nonincreasing returns to scale frontiers are displayed in the figure. If CRS is assumed, then only DMU C would be efficient; DMUs A, C, and E are efficient if VRS is assumed. Where the nonincreasing returns to scale and VRS frontiers are equal, decreasing returns to scale exist for those DMUs on the efficient frontier (such as E). Where the two frontiers are unequal, then increasing returns to scale exist for those DMUs (such as DMU B). The remaining DMUs, which are inefficient, can be classified as increasing returns to scale if the sum of the reference weights is less than unity for the CRS frontier or as decreasing returns to scale otherwise.

The efficiency of observation B is defined as $\theta_{B,\text{input,CRS}} = \overline{B_0B_1}/\overline{B_0B}$ for the input-oriented CRS DEA model and represents that one can obtain the same output by reducing the input by the ratio of $1 - \theta_{B,\text{input,CRS}}$. The efficiency for the output-oriented CRS DEA model is defined as $\theta_{B,\text{output,CRS}} = \overline{B_3B}/\overline{B_3C}$ and represents that one can obtain the same input by increasing the output by the ratio of $1 - \theta_{B,\text{output,CRS}}$. Accordingly, the input-oriented efficiency relative to the VRS frontier is defined as $\theta_{B,\text{input,VRS}} = \overline{B_0B_2}/\overline{B_0B}$. All efficiency measures of DMU C are the same regardless of orientation because the frontiers meet at point C.

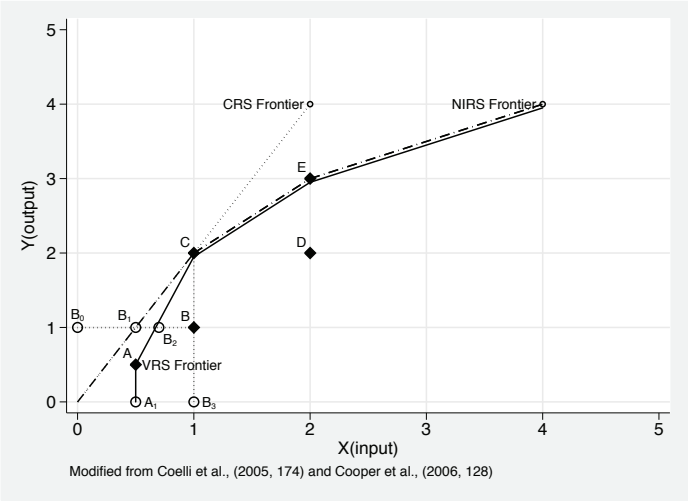

Figure 1. Concepts of efficiency and returns to scale

It is possible to decompose the CRS technical inefficiency into scale efficiency and "pure" technical efficiency. In figure 1, $\overline{B_2B}$ contributes to the technical efficiency of point B regarding the VRS model, and $\overline{B_1B}$ contributes to the technical efficiency of point B regarding the CRS model. Then $\overline{B_1B_2}$ contributes to scale efficiency.

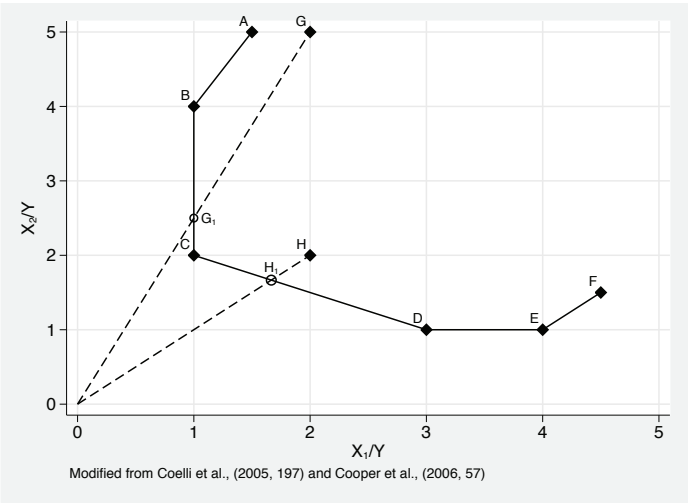

Figure 2. CRS input-oriented DEA example

Figure 2 illustrates the concepts of efficiency, slacks, and references or peers in an intuitive manner using two inputs and one output. The concept of frontier is especially important for the analysis of efficiency, because we measure efficiency as the relative

distance to the frontier. For example, firms that are technically inefficient operate at points in the interior of the frontier, while those that are technically efficient operate somewhere along the technology defined by the frontier. The DMU is called efficient when the DEA score equals 1 and all slacks are 0 (Cooper, Seiford, and Tone 2006). If only the first condition is satisfied, the DMU is called efficient in terms of "radial", "technical", and "weak" efficiency. If both conditions are satisfied, the DMU is called efficient in terms of "Pareto–Koopmans" or "strong" efficiency. The technical efficiencies of DMUs G and H are defined as $\overline{OG_1}/\overline{OG}$ and $\overline{OH_1}/\overline{OH}$, respectively.

Inefficiency can be seen as how much the inputs must contract along a ray from the origin until the ray crosses the frontier. For example, for firm G, the measure of technical efficiency is $\overline{OG_1}/\overline{OG}$. Point G_1 is the Farrell efficient point; however, input X_2 could be further reduced and still produce the same output. For this case, firm G has input slack $\overline{CG_1}$. If we disregard the slack and calculate it residually, the DEA model becomes the single-stage DEA model. The way to reduce the slack and find the Pareto optimal reference set can be further discussed; there are two-stage and multistage DEA models available in the literature (Cooper, Seiford, and Tone 2006; Coelli et al. 2005).

Cherchye and Puyenbroeck (2001) showed that "most representative efficient points" can be found using a direct approach and may differ from those obtained by multistage DEA. The DEA model in this article provides stage options for single-stage and two-stage which are still the most prevalent approaches in DEA literature. Reference or peer is a point that an inefficient DMU, such as point G, targets to move from the Farrell efficient point, such as point G_1, to the Pareto–Koopman efficient point, such as point C, in figure 2 (based on the two-stage DEA solution or Pareto optimal solution). However, the slack issues in DEA models disappear as the number of DMUs increases because the DEA piecewise linear frontier becomes smoother and has fewer chances to run the Farrell point to the input or output axes.

Free disposability means that one can produce the same output by wasting resources or increase the output without increasing resources. Strong disposability assumes that it is costless for firms to dispose of inputs or outputs or the isoquant does not bend backwards. In figure 2, the line that links A and F represents the frontier imposed by weak disposability. The line that links B and E represents the frontier imposed by strong disposability.

Assuming the economic production activities, convexity, strong disposability, and CRS, we can develop the linear program as a type of piecewise linear frontier. Input-oriented CRS efficiency is defined as (1) by applying the piecewise linear frontier to the input requirement set (Cooper, Seiford, and Tone 2006). This enables us to evaluate the efficiency relative to the frontier.

$$\max_{\nu,u} z = uy_j \tag{1}$$

subject to $\nu x_j = 1$, $-\nu X + uY \leq 0$, $\nu \geq 0$, $u \geq 0$, and u_j being free in sign, where a set of observed DMUs is $\text{DMU}_j, j = 1, \ldots, n$; x_j and y_j are input and output vectors; u is the row vector; ν are the output and input multipliers; and X and Y are the input and output matrices. The goal of the input-oriented DEA model is to minimize

the virtual input, relative to a given virtual output, subject to the constraint that no DMU can operate beyond the production possibility set and the constraint relating to nonnegative weights. In practice, most of the available DEA programs use the dual forms as expressed in (2), which lower the calculation burden and are virtually the same as (1).

$$\min_{\theta,\lambda} \theta \tag{2}$$

subject to $\theta x_j - X\lambda \geq 0$, $Y\lambda \geq y_j$, and $\lambda \geq 0$, where λ is a semipositive vector in R^k and θ is a real variable. The computational procedure for (2) can be expressed as

$$\min_{\theta} \theta \tag{3}$$

$$\min_{\lambda,s^+,s^-} \sum -s^+ - s^- \tag{4}$$

subject to $\theta x_j - X\lambda - s^- = 0$, $Y\lambda + s^+ = y_j$, and $\lambda \geq 0$, where s^+, s^-, and λ are semipositive vectors in R^k and θ is a real variable. The single-stage DEA model solves (3), and the two-stage DEA model solves (3) followed by (4), consecutively. The input-oriented CRS model was introduced in this section; however, other variations are easily extendable and available in most DEA literature, including Coelli et al. (2005) and Cooper, Seiford, and Tone (2000, 2006).

3 The dea command

3.1 Syntax

The syntax of the `dea` command is

```
dea ivars = ovars [if] [in] [, rts(crs|vrs|drs|nirs) ort(in|out)
   stage(1|2) trace saving(filename) ]
```

where *ivars* and *ovars* are input and output variable lists, respectively.

3.2 Description

`dea` requires the user to select the input and output variables from the user-designated data file or in the dataset currently in memory and solves DEA models with the specifications set in the options specified. There are several options to enhance the models. The user can select the desired options according to the particular model that is required.

The `dea` command requires an initial dataset that contains the input and output variables for observed DMU. Variable names must be identified by *ivars* for input variables and by *ovars* for output variables so that the `dea` command can identify and handle the multiple input–output dataset. In the output of the `dea` command, the prefix `dmu:` precedes DMU names.

The command has the ability to accommodate an unlimited number of inputs and outputs with an unlimited number of DMUs. The only limitation is the available computer memory. The resulting file reports information including reference points and slacks in the DEA model. This information can be used to analyze the inefficient DMU, for example, the source of the inefficiency and how an inefficient unit could be improved to the desired level.

`saving(`*filename*`)` creates *filename*`.dta`, which contains the results of `dea`, including information about the DMUs, inputs and outputs the data used, ranks of DMUs, efficiency scores, reference sets, and slacks. The log file `dea.log` will be created in the working directory.

Based on the data and the options specified, the `dea` command conducts matrix operations and linear programming to produce a results dataset that is available to print or can be used for further analysis.

3.3 Options

`rts(crs | vrs | drs | nirs)` specifies the returns to scale. The default, `rts(crs)`, specifies constant returns to scale. `rts(vrs)`, `rts(drs)`, and `rts(nirs)` specify variable returns to scale, decreasing returns to scale, and nonincreasing returns to scale, respectively.

`ort(in | out)` specifies the orientation. The default is `ort(in)`, meaning input-oriented DEA. `ort(out)` is output-oriented DEA.

`stage(1 | 2)` specifies the way to identify all efficiency slacks. The default is `stage(2)`, meaning two-stage DEA. `stage(1)` is single-stage DEA.

`trace` specifies to save all the sequences displayed in the Results window in the `dea.log` file. The default is to save the final results in the `dea.log` file.

`saving(`*filename*`)` specifies that the results be saved in *filename*`.dta`. If *filename*`.dta` already exists, the existing data will be moved to the file *filename*`_bak_`*DMYhms*`.dta` before the new data are saved in *filename*`.dta`.

3.4 Saved results

`dea` saves the following in `r()`:

Matrices

`r(dearslt)`	$n \times m$ matrix of the results of `dea`, where n is the number of DMUs and m is dependent on the model specified. Rows correspond to DMUs and columns correspond to variables, including inputs, outputs, rank (of DMU scores), theta (efficiency scores), ref. (reference DMUs), input and output slacks, and more, depending on the model specified.

4 Applications of dea

4.1 Data

This section provides examples using data from Cooper, Seiford, and Tone (2006, 75, table 3.7) and Coelli et al. (2005, 175, table 6.4) for illustration of the `dea` command. The data of Cooper, Seiford, and Tone (2006) consist of five stores that use two inputs—`i_employees` (number of employees as an input variable) and `i_area` (the area of floor as an input variable)—to produce two outputs: `o_sales` (the volume of sales as an output variable) and `o_profits` (the volume of profits as an output variable). The data of Coelli et al. (2005) consist of five firms that use one input, `i_1`, to produce one output, `o_1`.

4.2 CRS input-oriented two-stage DEA model

The `dea` default specifies a CRS input-oriented two-stage DEA model. If you want to use this specification for your analysis, just use the `dea` command as we have below using `cooper_table3.7.dta`. Then you will have the following results. Store E is the only efficient DMU and is the referent for all other stores, which is an equivalent result to Cooper, Seiford, and Tone (2006, 75–76). The two-stage DEA model provides the optimal solution, as shown in the results table.

For example, the optimal solution of efficiency score (theta), reference weights (λ_A, λ_B, λ_C, λ_D, λ_E), and slack (`i_area`, `i_employee`, `o_sales`, `o_profits`) for Store A are 0.933333, $(0, 0, 0, 0, 0.777778)$, and $(11.6667, 0, 0, 0, 0.222222)$, respectively. Thus the performance of Store A can be improved by subtracting 11.6667 units from input (area) and 0.777778 unit from output (profits) even after they have reduced all inputs by 6.67% without worsening any other input and output. Because Store A has an efficient score of 93.33%, all inputs (employee and area) could be reduced by 6.67%. In addition, because Store A has an input slack for area of 11.67, 11.67 units of area could be reduced even after Store A has reduced all inputs by 6.67%.

(*Continued on next page*)

```
. use cooper_table3.7.dta
. dea i_employee i_area = o_sales o_profits
```

```
options: RTS(CRS) ORT(IN) STAGE(2)
                                                  ref:
                    rank       theta     store_A
dmu:store_A            2     .933333           .
dmu:store_B            3     .888889           .
dmu:store_C            5     .533333           .
dmu:store_D            4     .666667           .
dmu:store_E            1           1           .

                    ref:        ref:        ref:
                 store_B     store_C     store_D
dmu:store_A            .           .           .
dmu:store_B            .           .           .
dmu:store_C            .           .           .
dmu:store_D            .           .           .
dmu:store_E            .           .           .

                    ref:     islack:     islack:
                 store_E  i_employee      i_area
dmu:store_A     .777778           .     11.6667
dmu:store_B     1.11111           .     3.33333
dmu:store_C     .888889           .           8
dmu:store_D     1.11111     3.33333           .
dmu:store_E           1           .           0

                 oslack:     oslack:
                o_sales   o_profits
dmu:store_A    2.98e-07     .222222
dmu:store_B    7.45e-07     5.88889
dmu:store_C    4.17e-06     2.11111
dmu:store_D    2.98e-06     6.88889
dmu:store_E           .           .
```

4.3 CRS output-oriented single-stage DEA model

If you want to use the CRS output-oriented single-stage DEA model, you can use the `dea` command as we have below using `coelli_table6.4.dta`.

```
. use coelli_table6.4.dta
. dea i_x = o_q, rts(crs) ort(o) stage(1)
```

```
options: RTS(CRS) ORT(OUT) STAGE(1)
CRS-OUTPUT Oriented DEA Efficiency Results:
                              ref:     ref:     ref:     ref:     ref:
          rank    theta         A        B        C        D        E
dmu:A        4       .5         .        .  .333333        .        .
dmu:B        4       .5         .        .  .666667        .        .
dmu:C        1        1         .        .        1        .        .
dmu:D        3       .8         .        .  1.33333        .        .
dmu:E        2  .833333         .        .  1.66667        .        .
         islack:  oslack:
            i_x      o_q
dmu:A        .        .
dmu:B        .        .
dmu:C        .        .
dmu:D        .        .
dmu:E        .        .
```

The rank of DMUs and efficiency score (`theta`), as well as the residually given reference set (`ref:`) and the slacks (`islack:` or `oslack:`), are listed in the above results. Store C is the only efficient DMU and seems to be the referent for all other stores. The Results window will display the above result, and a `dea.log` file that contains the above result will be created in your working directory. The "`.`" in the results table represent small numbers less than 10 to the minus 12 power, which mostly can be ignored. However, sometimes when you want to analyze financial data, the distinction between zero and "`.`" values may be required to maintain accuracy.

4.4 VRS input-oriented single-stage DEA model

Now we illustrate the VRS input-oriented single-stage DEA analysis using the data of `coelli_table6.4.dta`:

(*Continued on next page*)

```
. dea i_x = o_q, rts(vrs) stage(1)
```

```
options: RTS(VRS) ORT(IN) STAGE(1)
VRS-INPUT Oriented DEA Efficiency Results:
                              ref:     ref:     ref:     ref:     ref:
          rank    theta         A        B        C        D        E
dmu:A        1        1         1        .        .        .        .
dmu:B        5     .625        .5        .       .5        .        .
dmu:C        1        1         0        .        1        .        .
dmu:D        4       .9         .        .       .5        .       .5
dmu:E        1        1         .        .        0        .        1

         islack:  oslack:
            i_x      o_q
dmu:A         .        0
dmu:B         .        .
dmu:C         .        .
dmu:D         .        .
dmu:E         .        .

VRS Frontier(-1:drs, 0:crs, 1:irs)
           CRS_TE     VRS_TE    NIRS_TE      SCALE        RTS
dmu:A    0.500000   1.000000   0.500000   0.500000   1.000000
dmu:B    0.500000   0.625000   0.500000   0.800000   1.000000
dmu:C    1.000000   1.000000   1.000000   1.000000   0.000000
dmu:D    0.800000   0.900000   0.900000   0.888889  -1.000000
dmu:E    0.833333   1.000000   1.000000   0.833333  -1.000000

VRS Frontier:
```

	dmu	o_q	i_x	CRS_TE	VRS_TE	SCALE	RTS
1.	A	1	2	0.500000	1.000000	0.500000	irs
2.	B	2	4	0.500000	0.625000	0.800000	irs
3.	C	3	3	1.000000	1.000000	1.000000	-
4.	D	4	5	0.800000	0.900000	0.888889	drs
5.	E	5	6	0.833333	1.000000	0.833333	drs

If `rts(vrs)` is specified, the results show some additional information, as shown above. Stores D and E are on the decreasing returns to scale (`drs`) portion of the VRS frontier. On the other hand, Stores A and B are on the increasing returns to scale (`irs`) portion of the VRS frontier.

4.5 VRS input-oriented two-stage DEA model

Here we illustrate the VRS input-oriented two-stage DEA model, again using data from `coelli_table6.4.dta`:

```
. dea i_x = o_q, rts(vrs) ort(i)
```

```
options: RTS(VRS) ORT(IN) STAGE(2)
VRS-INPUT Oriented DEA Efficiency Results:
                              ref:     ref:     ref:     ref:     ref:
          rank    theta         A        B        C        D        E
dmu:A        1        1         1        .        0        .        .
dmu:B        5     .625        .5        .       .5        .        .
dmu:C        1        1         .        .        1        .        .
dmu:D        4       .9         .        .       .5        .       .5
dmu:E        1        1         .        .        0        .        1

         islack:  oslack:
            i_x      o_q
dmu:A         .        .
dmu:B         .        .
dmu:C         .        .
dmu:D         .        .
dmu:E         0        .

VRS Frontier(-1:drs, 0:crs, 1:irs)
           CRS_TE     VRS_TE    NIRS_TE      SCALE        RTS
dmu:A    0.500000   1.000000   0.500000   0.500000   1.000000
dmu:B    0.500000   0.625000   0.500000   0.800000   1.000000
dmu:C    1.000000   1.000000   1.000000   1.000000   0.000000
dmu:D    0.800000   0.900000   0.900000   0.888889  -1.000000
dmu:E    0.833333   1.000000   1.000000   0.833333  -1.000000

VRS Frontier:
```

	dmu	o_q	i_x	CRS_TE	VRS_TE	SCALE	RTS
1.	A	1	2	0.500000	1.000000	0.500000	irs
2.	B	2	4	0.500000	0.625000	0.800000	irs
3.	C	3	3	1.000000	1.000000	1.000000	-
4.	D	4	5	0.800000	0.900000	0.888889	drs
5.	E	5	6	0.833333	1.000000	0.833333	drs

Note that the efficiency score (theta) of DMU B is 0.625, and DMUs A and C are the reference DMUs for DMU B. The sum of the reference weights should equal 1 because `rts(vrs)` specifies that $\sum_{j=1}^{n} \lambda_j = 1$. The sum of the reference weights for DMU B equals 1 from $(\lambda_A, \lambda_B, \lambda_C, \lambda_D, \lambda_E) = (0.5, 0, 0.5, 0, 0)$. And note that the efficiency scores of all the DMUs are not changed from the single-stage model shown in the previous section to the current two-stage analysis because there is no slack in this case. This means that the slack level of profits has no effect on the efficiency evaluation. Stores A, C, and E are the efficient points that inefficient DMUs (B and D) can target to move in input-oriented DEA calculation.

Here we analyze data from Coelli et al. (2005, 175) using the `saving()` option:

```
. dea i_x = o_q, rts(vrs) ort(i) saving(coelli_6.4_results)
```

(*output omitted*)

Specifying `saving(coelli_6.4_results)` will save the VRS frontier results, shown above, as `coelli_6.4_results.dta`. The results match the Pareto–Koopman solution of Coelli et al. (2005, 176, table 6.5) because slack has no role in this case.

4.6 Second-stage regression analysis using the efficiency scores

The prevalent method in the literature to find the determinants of efficiency gaps among DMUs is by using tobit regression analysis because the efficiency scores are censored at the maximum value of the efficiency scores. The tobit regression analysis uses the efficiency scores as the dependent variables for the possible candidates of influential variables (see, for example, Lee, Lee, and Kim [2009]).

Here we illustrate the second-stage regression analysis using `seco_stage_regression.dta`.

```
. dea i_x = o_q, saving(seco_stage_regre_results)
  (output omitted)
. use seco_stage_regre_results
. tobit theta rnd, ul(1)
Tobit regression                                  Number of obs   =         20
                                                  LR chi2(1)      =      46.44
                                                  Prob > chi2     =     0.0000
Log likelihood =  19.060498                       Pseudo R2       =     5.5845

------------------------------------------------------------------------------
       theta |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         rnd |   .1464713   .0118676    12.34   0.000     .1216322    .1713104
       _cons |   .3003908   .0360827     8.33   0.000     .2248688    .3759127
-------------+----------------------------------------------------------------
      /sigma |   .0649661    .011565                      .0407603     .089172
------------------------------------------------------------------------------
  Obs. summary:          0  left-censored observations
                        16     uncensored observations
                         4 right-censored observations at theta>=1
```

The results show that the `rnd` (R&D) level is positively related with the CRS efficiency scores of DMUs at the 1% level of significance.

5 Conclusion

Today, many academic researchers recognize Stata as one of the leading packages for statistical analysis; however, there are still uncovered areas that managerial organizations are interested in. In particular, optimization procedures in Stata can be further developed to fill in the gaps between parametric and nonparametric analysis. The `dea` command introduced in this article is a new application in Stata and is a powerful managerial tool for measuring the efficiency and productivity of DMUs.

The `dea` command application has several advantages, including the following:

- It can be used by Stata users with no extra cost for DEA software.
- It is flexible to add other DEA models.
- It provides Stata with managerial tools for reports and statistical analysis, as well as optimization procedures.
- The `dea` command report files can directly feed to other Stata routines for further analysis.

6 Acknowledgments

We thank H. Joseph Newton and an anonymous reviewer for comments.

7 References

Banker, R. D., A. Charnes, and W. W. Cooper. 1984. Some models for estimating technical and scale inefficiencies in data envelopment analysis. *Management Science* 30: 1078–1092.

Boussofiane, A., R. G. Dyson, and E. Thanassoulis. 1991. Applied data envelopment analysis. *European Journal of Operational Research* 52: 1–15.

Charnes, A., W. W. Cooper, and E. Rhodes. 1978. Measuring the efficiency of decision making units. *European Journal of Operational Research* 2: 429–444.

Cherchye, L., and T. V. Puyenbroeck. 2001. A comment on multi-stage DEA methodology. *Operations Research Letters* 28: 93–98.

Coelli, T. J., D. S. P. Rao, C. J. O'Donnell, and G. E. Battese. 2005. *An Introduction to Efficiency and Productivity Analysis*. 2nd ed. New York: Springer.

Cooper, W. W., L. M. Seiford, and K. Tone. 2000. *Data Envelopment Analysis: A Comprehensive Text with Models, Applications, References and DEA-Solver Software*. 2nd ed. New York: Springer.

———. 2006. *Introduction to Data Envelopment Analysis and Its Uses*. New York: Springer.

Lee, C., J. Lee, and T. Kim. 2009. Innovation policy for defense acquisition and dynamics of productive efficiency: A DEA application to the Korean defense industry. *Asian Journal of Technology Innovation* 17: 151–171.

About the authors

Yong-bae Ji is a graduate student in the Defense Science and Technology Department at Korea National Defense University in Seoul, Republic of Korea.

Choonjoo Lee (corresponding author) is an assistant professor in the Defense Science and Technology Department at Korea National Defense University in Seoul, Republic of Korea.