

Data Envelopment Analysis in the Presence of Partial Input-to-Output Impacts

Wade D. Cook * and Raha Imanirad

Schulich School of Business, York University, Toronto, Ontario, Canada

Abstract

Data envelopment analysis (DEA) is a methodology used to evaluate the relative efficiencies of peer decision-making units (DMUs) in multiple input, multiple output situations. In the original formulation, and in the vast literature that followed, the assumption was that all members of the input bundle affected the output bundle. However, many potential applications of efficiency measurement exist wherein some inputs do not influence certain outputs. For example, in a manufacturing setting from which multiple products (outputs) emerge, resources (e.g., packaging labor) will not affect products that do not pass through that department. For this paper, extension of the conventional DEA methodology allows for the measurement of technical efficiency in situations where only partial input-to-output impacts are evident. Evaluating the efficiencies of a set of steel fabrication plants using the methodology was the focus of the research.

Keywords: DEA, partial input to output impacts

JEL Classification code: C02

Data envelopment analysis (DEA), first introduced by Charnes, Cooper, and Rhodes (1978), is a methodology useful in evaluating the relative efficiencies of peer decision-making units (DMUs) in a multiple input, multiple output setting. In the original formulation, and in the vast literature that followed, the assumption was that all members of the input bundle affected the output bundle. However, in many settings measured for efficiency, some inputs may not influence certain outputs. For example, in a manufacturing setting producing multiple products (outputs), various inputs may not influence some outputs at all. Painting or packaging resources only affect products that require painting or packaging. Similarly, in hospitals, certain staff may not influence particular activities, so evaluation of those activities in terms of those staff inputs should not occur.

The current paper involves an examination of the measurement of efficiency within the context of a set of steel fabrication plants in which the phenomenon of partial impacts of inputs on outputs is present. A brief description of the problem setting follows. The next section illustrates the development of a general methodology, in the spirit of DEA, designed to address such partial interactions between inputs and outputs. The methodology is based on the view of a DMU as a business unit comprising a set of separate subunits, with the proposition being to define efficiency of the DMU as a weighted average of the efficiencies of the subunits. An application of the new methodology using data related to a set of 20 fabrication plants and the conclusions of the paper follow.

Efficiency Measurement in Steel Fabrication Plants

The focus of the current paper is to examine the efficiencies of a set of steel fabrication plants. The product lines were generally consistent across the plants and reflected four primary categories:

1. sheet steel products (ladders, guards, bumpers, and conveyors),
2. flat bar products used mainly in building construction (brackets, base plates, headers, and posts),
3. pipes and cylinders (storm drains, plumbing products, etc.), and
4. cylindrical bearings (automotive and nonautomotive).

These four product groupings constituted the outputs for analysis purposes. For the inputs, plant resources included four categories: (a) plant labor, (b) shearing machines, (c) presses, and (d) lathes. A number of smaller machines and tools, such as grinders and welders, were included in the lathes category.

In this setting, not all inputs affected all outputs. Table 1 shows the input-to-output connections. Pipes and cylinders, for example, do not require the usage of shearing machines and presses. Shearing machines and presses are necessary only in the fabrication of sheet steel and flat bar products.

Table 1
Input-to-Output Connections

Inputs	Outputs			
	Sheet steel	Flat bar	Pipes/cylinders	Cylindrical bearings
Labor	X	X	X	X
Shears	X	X		
Presses	X	X		
Lathes		X	X	X

In the conventional DEA context, the assumption is that all members of the stated input set for a DMU influence all members of the output set, a concept prevalent in DEA literature. For efficiency measurement, in situations in which the usual assumptions do not hold, the conventional DEA models, such as the radial Charnes, Cooper, Rhodes (CCR) model, are not appropriate.

One approach to the problem is to view the DMU as a business unit consisting of a set of K subunits, within which the conventional assumptions are valid. Specifically, each input bundle $k = 1, \dots, K$ will be represented by its own input-output bundle (I_k, R_k) , where each input in I_k affects every output in R_k . Given this, treating any business subunit as a DMU in the conventional sense and carrying out the standard DEA analysis (on that subunit across the various DMUs) would be appropriate.

Consider the input-output profile described above. The output group $R = (1, 2, 3, 4)$ consists of three subgroups, namely $R_1 = (1)$, $R_2 = (2)$, $R_3 = (3, 4)$, where the outputs have been numbered 1, 2, 3, and 4. After numbering the inputs 1, 2, 3, and 4, the input sets corresponding to the three output sets are $I_1 = (1, 2, 3)$, $I_2 = (1, 2, 3, 4)$, $I_3 = (1, 4)$. Thus, the DMU includes three subunits, (I_1, R_1) , (I_2, R_2) , (I_3, R_3) . A formal algorithm for generating these bundles is presented later in the paper.

Viewing the subunits independently of one another is necessary and involves assigning a portion α of each input to each subunit of which it is a member. For example, for each DMU j , Input 1 (labor) is a member of each of the three subunits, so apportioning that input across the three subunits $k = 1, 2, 3$ in the amounts $\alpha_{1k} x_{1j}$ is required. Input 4, however, is a member of only two of the subunits, so splitting the total of that resource across only those two subunits is prudent. Figure 1 illustrates the splitting of resources across subunits according to which resources have membership.

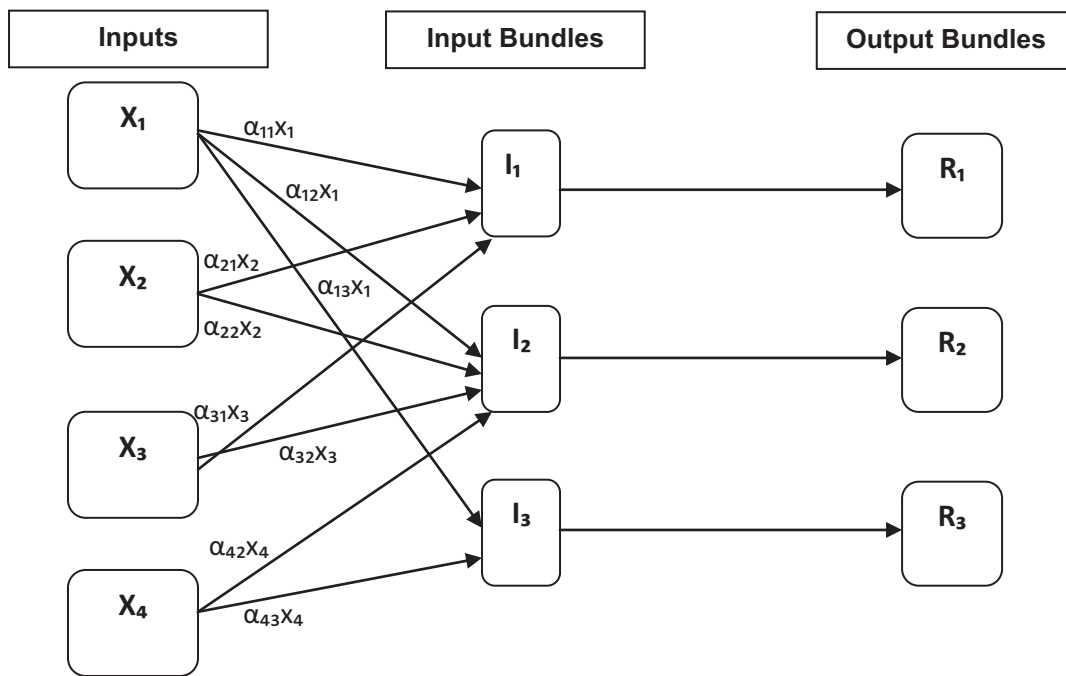


Figure 1. Resource splitting across business subunits.

The following section describes the development of a DEA-based methodology for measuring efficiency in settings where such partial input-to-output interactions exist.

Modeling Efficiency in the Presence of Partial Input-to-Output Interactions

Formalizing the ideas presented so far involves progressing through three stages to arrive at an efficiency score for each DMU:

1. Derive an appropriate split α_{ik} of each input i to each bundle k of which it is a member.
2. Using the split of inputs from Stage 1, apply the conventional DEA analysis to each of the K subunits.
3. Derive an overall efficiency score by combining the subunit scores from Stage 2.

Stage 1: Deriving the Split of Inputs across Subunits

Representing the overall efficiency of a DMU j_o as some weighted average of the K subgroup efficiencies is reasonable, assuming that the DMU may be represented as the sum of its parts, meaning that there are no economies or diseconomies of scope. Where such economies or diseconomies of scope are present, the suggested approach may not accurately capture efficiency at the aggregate level. See Panzar and Willig (1981) and Pulley and Braunstein (1992) for a discussion of economies of scope.

Given that the goal is to derive the aggregate efficiency of the DMU and that this aggregate will be represented as a convex combination of the K subgroup efficiencies, determining the α split of inputs (with the objective of maximizing this aggregate efficiency) is necessary. Consider the following input-oriented radial projection model for a DMU j_o :

$$e_o = \text{Max} \sum_{k=1}^K W_k \left[\sum_{r \in R_k} u_r y_{rj_o} / \sum_{i \in I_k} v_i \alpha_{ik} x_{ij_o} \right] \quad (1)$$

s.t.

$$\sum_{k=1}^K W_k \left[\sum_{r \in R_k} u_r y_{rj} / \sum_{i \in I_k} v_i \alpha_{ik} x_{ij} \right] \leq 1, \quad \forall j, \quad (2)$$

$$\sum_{r \in R_k} u_r y_{rj} / \sum_{i \in I_k} v_i \alpha_{ik} x_{ij} \leq 1, \quad \forall k, j, \quad (3)$$

$$\sum_{k \in L_i} \alpha_{ik} = 1, \quad \forall i, \quad (4)$$

$$a_{ik} \leq \alpha_{ik} \leq b_{ik}, \quad \forall i, k, \quad (5)$$

$$u_r, v_i, \alpha_{ik} \geq \varepsilon, \quad \forall i, k. \quad (6)$$

While the weights W_k can be any set of values that represent the importance of the relevant subunits, it is fitting in many situations, to allow the proportion of inputs assigned to subunits to dictate their importance. If, for example, 40% of the inputs are assigned to a particular subunit and 60% to another, assigning a weight of 40% to the efficiency ratio of the first subunit and 60% to the other is reasonable. Thus, we define herein

$$W_k = \sum_{i \in I_k} v_i \alpha_{ik} x_{i_o} / \sum_{k=1}^K \left[\sum_{i \in I_k} v_i \alpha_{ik} x_{i_o} \right] \quad (7)$$

At the same time, and in anticipation of the second stage, it is essential to impose the restriction that the ratio of weighted outputs to weighted inputs at the subgroup level not exceed unity. Specifically, constraints (3) specify that the α_{ik} be selected in a manner that ensures that the efficiency ratio corresponding to each DMU j in each of its subunits does not exceed unity for some values of the multipliers u_r, v_i . In the presence of constraints (3), constraints (2) are redundant, so imposing the latter set is unnecessary. In constraints (4), the set L_i denotes those subunits k that have i as a member. These constraints specify that those α values sum to unity for each i . Finally, constraints (5) limit the sizes of the α variables.

The current structure of model (1-6) is nonlinear. To convert the formulation to one that is more tractable, first note that by virtue of the definition of W_k , as given by Equation 7, the objective function (1) becomes

$$e_o = \text{Max} \sum_{k=1}^K \sum_{r \in R_k} u_r y_{rj_o} / \sum_i v_i x_{ij_o} \quad (8)$$

Next, implement the change of variables $z_k = v_i \alpha_k$ meaning that

$$\sum_{k \in L_i} \alpha_k = 1 \Rightarrow v_i \sum_{k \in L_i} \alpha_k = v_i \Rightarrow \sum_{k \in L_i} z_k = v_i$$

Using the Charnes and Cooper (1962) transformation, $t = 1 / \sum_i v_i x_{ij_o}$, and defining

$\mu_r = t u_r, v_i = t v_i, \gamma_{ik} = t z_{ik}$, model 1-6 becomes the following:

$$e_o = \text{Max} \sum_{k=1}^K \sum_{r \in R_k} \mu_r y_{rj_o} \quad (9)$$

s.t.

$$\sum_i v_i x_{ij_o} = 1, \quad (10)$$

$$\sum_{r \in R_k} \mu_r y_{rj} - \sum_i \gamma_{ik} x_{ik} \leq 0, \quad \forall j, k, \quad (11)$$

$$\sum_{k \in L_i} \gamma_{ik} = v_i, \quad \forall i, \quad (12)$$

$$v_i a_{ik} \leq \gamma_{ik} \leq v_i b_{ik}, \quad \forall i, k, \quad (13)$$

$$\mu_r, v_i, \gamma_{ik} \geq \varepsilon, \quad \forall r, i, k. \quad (14)$$

Stage 2: Deriving Subunit Efficiencies

The result of applying the model (9-14) to data, such as that in Table 1, is a split of the inputs across the relevant subunits, as illustrated in Figure 1. Specifically, the γ_{ik} and v_i generated from the solution of model (9-14) are needed to compute the α_{ik} , that is $\alpha_{ik} = \gamma_{ik} / v_i$. The α_{ik} are then applied to the relevant inputs i to generate the input data needed for the efficiency evaluation of the k th subunit. The outcome of this stage is a set of K subunit efficiency scores.

Stage 3: Deriving the Overall Efficiency Scores for the DMUs

Stage 3 involves combining the K subunit scores, arising from Stage 2, by calculating their weighted average, using W_k as defined in Equation 7. The idea is to partition the input-output set into K bundles (I_k, R_k) . A discussion of the formation of these bundles is given in the following section.

Generating the Input-Output Bundles

In a multiple input-output setting, for each $k = 1, \dots, K$, let I_k and R_k denote a set of inputs and outputs respectively. The first objective is to generate input-output bundles $(I_1, R_1), (I_2, R_2), \dots, (I_k, R_k)$ in a way that the $R_{k=1}^K$ form a mutually exclusive set and that for each k , (I_k, R_k) is maximal.

Definition

An input-output bundle (I_k, R_k) is *maximal* if it possesses the following two properties:

1. Every output r in R_k is influenced by every input i in I_k , and no other input outside of I_k influences any output r in R_k .
2. No output outside of R_k whose input bundle is identical to that of R_k exists.

However, there can be an input in i_0 a given bundle I_k that influences an output r_0 outside of R_k , but at least one i in I_k does not influence r_0 .

For a given multiple input-output setting, the algorithm for generating the maximal bundles includes the following three steps:

1. Define S to be an empty set.
2. For each output r , derive $I(r)$, the set of all inputs i that influence r . Add $I(r)$ to S.
3. Compare each $I(r)$ in S with every other $I(r')$ in S, and identify all $I(r')$ that have the same elements as $I(r)$. If no such r' is evident, create bundle (I_k, R_k) using $I(r)$ and r so that $(I_k, R_k) = (I(r), r)$. Remove $I(r)$ from S. Otherwise, group outputs r and all r' together to derive R_k , and create bundle (I_k, R_k) , using $I(r)$ and R_k so that $(I_k, R_k) = (I(r), R_k)$. Remove $I(r)$ and all $I(r')$ from S.

Theorem. The generated set of maximal input-output bundles is unique.

Proof. If one were to assume that the set of maximal bundles was not unique, at least two different sets of maximal bundles, S_1 and S_2 , must exist. Accordingly, at least one input-output bundle B_k in S_1 that is different from every bundle B'_k in S_2 must be present. B_k and B'_k may differ in terms of their respective input and/or output sets. If the input sets I_k and I'_k are different, there must exist at least one input i_k such that $i_k \in I_k$ and $i_k \notin I'_k$. If i_k influences any $r \in R_k$ then bundle B'_k violates the first requirement of a maximal bundle because input i_k outside of I'_k influences $r \in R_k$ is evident. Otherwise, bundle B_k violates the first requirement of a maximal bundle because $i_k \in I_k$ does not influence any $r \in R_k$. In either case, only one maximal bundle exists.

In the case of a difference between output sets R_k and R'_k , there must be at least one output r_k such that $r_k \in R_k$ and $r_k \notin R'_k$. If the input bundle of r_k is not I_k then bundle B_k violates the first requirement of a maximal bundle because an input $i \in I_k$ that does not influence r_k exists or an input outside of I_k that influences r_k exists. Otherwise, if the input bundle of r_k is equal to I_k , bundle R'_k violates the second requirement of a maximal bundle because there exists an output r outside of R'_k with an input bundle identical to that of R'_k . In either case, only one maximal bundle can exist. This completes the proof.

The VBA code for this algorithm is presented in Appendix B.

Application

This section discusses an examination of the efficiency measurement of a set of 20 steel fabrication plants.

Table A1 displays data on 20 steel fabrication plants, described by four inputs and four outputs. Given the input-output interaction in Table 1, model (9-14) was run to determine the α_{ik} . The ranges (a, b) for the α_{ik} as required in the model were initially set at $(.1, .6)$ meaning that the largest share of any input assigned to any bundle of which it is a member is 60%, and the smallest is 10%. Table A2 illustrates the resulting α_{ik} .

The resulting efficiency scores are displayed in Table A3. Specifically, Table A3 shows the aggregate score from model (9-14), the three subunit scores, and the overall score (weighted average of the subunit scores) for each DMU. None of the DMUs has an efficiency score of 100%. (This would require that all K subunits be 100% efficient as well). In the case of DMUs 7, 9, 10, 12, and 15-20, at least one subunit is efficient.

One might hypothesize that the various scores would be overly sensitive to the particular (a, b) ranges chosen. Thus, an alternative set of ranges was selected and the analysis repeated. The second analysis included two levels of ranges. For Input 1, split across all three input bundles, the (a, b) range was set at $(.1, .6)$. For Inputs 2, 3, and 4, each of which is split across only two bundles, the (a, b) range was set at $(.3, .7)$. The resulting efficiency scores appear in Table A4. While some movement was apparent in the aggregate scores from model (9-14), the subunit scores and their weighted averages (the overall scores) reflected virtually no change, attesting to the relative stability of the methodology. Arguably, if one were to impose fixed proportional splits of the resources, one might anticipate that the efficiency scores would undergo shifts that are more substantial.

A conventional DEA analysis on the DMUs, without distinguishing among the subunits, followed to ensure a comprehensive examination. Table A5 illustrates the results. Half of the DMUs (7, 9, 10, 12, and 15-20) are shown to be efficient. The earlier analysis indicated that at least one subunit was efficient for the same DMUs. In the conventional analysis, a minimal or zero weight is assigned to certain inputs and outputs, meaning that significant nonzero weights are attached only to certain subsets of inputs and outputs. In most cases, these subsets correspond precisely to the previously identified subunits. Hence, the approach proposed in this paper provides significantly more information about the inner workings of the DMU in terms of which parts of the business unit are operating at an efficient level and which are not. A related type of analysis designed to examine the internal structure of the DMU is network DEA, as discussed in Färe and Grosskopf (1996); Cook, Liang, and Zhu (2010); and Cook, Zhu, Bi, and Yang (2010).

Conclusions

The purpose of this paper was to present a DEA methodology useful in addressing efficiency measurement situations in which not all inputs affect all outputs. Such an incomplete input-to-output interaction occurs in many different environments, rendering the conventional DEA models inappropriate. The basis of the model is a separation of the DMU into subunits in each of which the full input-to-output interaction is present, making the conventional DEA models applicable (at that subunit level). Defining a unique set of (input, output) bundles that collectively represents the DMU is necessary to implement the methodology. The paper includes a statement of an algorithm for developing the bundles, as well as an application of the new methodology to the problem of measuring the efficiency of a set of steel fabrication plants.

The incompleteness concept has widespread applicability and many important extensions. One area currently under investigation is that in which some of the inputs are nondiscretionary and nonseparable. Another research direction pertains to the application of assurance regions in an incomplete setting.

References

- Charnes, A., & Cooper, W.W. (1962). Programming with linear fractional functionals. *Naval Research Logistics Quarterly*, 9, 67-88.
- Charnes, A., Cooper, W. W., & Rhodes, E. (1978). Measuring the efficiency of decision making units. *European Journal of Operational Research*, 2(6), 429-444.
- Cook, W. D., Liang, L., & Zhu, J. (2010). Measuring performance of two-stage network structures by DEA: A review and future perspective. *OMEGA*, 38(6), 423-430.
- Cook, W. D., Zhu, J., Bi, G., & Yang, F. (2010). Network DEA: Additive efficiency decomposition. *European Journal of Operational Research*, 207(2), 1122-1129.
- Färe, R., & Grosskopf, S. (1996). Productivity and intermediate products: A frontier approach. *Economics Letters*, 50(1), 65-70.
- Panzar, J. C., & Willig, R. D. (1981). Economies of scope. *American Economic Review*, 71(2), 268-272.
- Pulley, L. B., & Braunstein, Y. M. (1992). A composite cost function for multiproduct firms with an application to economies of scope in banking. *Review of Economics and Statistics*, 74(2), 221-230.

Authors Note

Wade D. Cook and Raha Imanirad, *Schulich School of Business, York University, 4700 Keele Street, Toronto, Ontario, Canada M3J 1P3.*

Correspondence concerning this article should be addressed to Wade D. Cook, Email: wcook@schulich.yorku.ca

Appendix A

Table A1
Data on 20 Plants

DMU	OUTPUT				INPUT			
	Sheet steel	Flat bar	Pipes/ cylinders	Bearings	Labor	Shears	Presses	Lathes
	Y1	Y2	Y3	Y4	X1	X2	X3	X4
1	70	103	100	80	30	5.0	5.0	15.0
2	60	125	90	90	40	4.0	4.0	18.0
3	50	110	105	85	35	5.2	4.2	10.0
4	80	80	110	90	38	7.0	4.6	8.5
5	56	40	60	55	28	9.0	5.5	12.5
6	40	95	120	110	37	4.2	3.8	14.0
7	100	180	200	210	31	6.0	4.1	11.0
8	25	55	180	160	35	5.0	5.0	15.0
9	65	150	125	145	25	6.2	4.8	19.0
10	40	110	70	115	30	3.0	3.2	21.0
11	70	117	122	115	25	4.0	4.0	12.0
12	92	135	89	64	45	5.0	3.3	23.0
13	88	47	57	109	35	4.1	6.0	20.5
14	48	68	146	99	32	5.3	3.4	11.2
15	79	123	220	122	26	7.7	4.3	15.6
16	99	114	89	49	19	5.3	4.2	12.4
17	97	101	88	55	25	8.0	3.0	8.8
18	55	55	132	116	32	6.0	2.8	6.8
19	80	97	142	168	33	2.8	3.9	13.4
20	97	68	209	122	27	3.3	4.3	21.6

Table A2
 α_{ik} Values

DMU	X_1	X_1	X_1	X_2	X_2	X_3	X_3	X_4	X_4
	K = 1	K = 2	K = 3	K = 1	K = 2	K = 1	K = 2	K = 2	K = 3
1	0.6	0.3	0.1	0.4	0.6	0.6	0.4	0.5	0.5
2	0.32136	0.32136	0.35727	0.4	0.6	0.4	0.6	0.6	0.4
3	0.1	0.3	0.6	0.4	0.6	0.6	0.4	0.6	0.4
4	0.2	0.2	0.6	0.6	0.4	0.6	0.4	0.4	0.6
5	0.6	0.1	0.3	0.5	0.5	0.6	0.4	0.6	0.4
6	0.31856	0.31856	0.36287	0.4	0.6	0.4	0.6	0.6	0.4
7	0.1	0.3	0.6	0.6	0.4	0.6	0.4	0.6	0.4
8	0.1	0.3	0.6	0.5	0.5	0.5	0.5	0.4	0.6
9	0.1	0.6	0.3	0.4	0.6	0.4	0.6	0.5	0.5
10	0.1	0.6	0.3	0.4	0.6	0.4	0.6	0.6	0.4
11	0.28027	0.6	0.11973	0.4	0.6	0.5	0.5	0.5	0.5
12	0.28027	0.6	0.11973	0.4	0.6	0.5	0.5	0.5	0.5
13	0.6	0.1	0.3	0.6	0.4	0.6	0.4	0.5	0.5
14	0.1	0.3	0.6	0.5	0.5	0.6	0.4	0.4	0.6
15	0.1	0.3	0.6	0.5	0.5	0.4	0.6	0.4	0.6
16	0.3	0.6	0.1	0.6	0.4	0.6	0.4	0.6	0.4
17	0.6	0.1	0.3	0.6	0.4	0.6	0.4	0.5	0.5
18	0.26354	0.13646	0.6	0.5	0.5	0.6	0.4	0.4	0.6
19	0.33333	0.33333	0.33333	0.4	0.6	0.5	0.5	0.6	0.4
20	0.59320	0.1	0.30680	0.6	0.4	0.6	0.4	0.5	0.5

Table A3
Subunit and Overall Efficiency Scores

DMU	Aggregate score [Problem 3.3]	Score K1	Score K2	Score K3	Overall score
1	0.60666	0.59829	0.66125	0.45471	0.57142
2	0.74626	0.61939	0.92871	0.34035	0.62948
3	0.61945	0.47002	0.70166	0.56188	0.57786
4	0.62756	0.65130	0.57491	0.67109	0.63243
5	0.34732	0.40929	0.24349	0.31090	0.32123
6	0.55897	0.42169	0.70402	0.48723	0.53764
7	0.99880	0.92883	1.00000	1.00000	0.97628
8	0.53526	0.20955	0.34737	0.73556	0.43083
9	0.90440	0.56048	1.00000	0.85445	0.80498
10	0.79336	0.52589	1.00000	0.56169	0.69586
11	0.81295	0.74255	0.92870	0.71739	0.79621
12	0.92158	1.00000	0.93097	0.25792	0.72963
13	0.57798	0.73011	0.33541	0.45486	0.50679
14	0.57139	0.52012	0.45542	0.71059	0.56204
15	0.90276	0.68224	0.79987	1.00000	0.82737
16	0.95521	1.00000	1.00000	0.55353	0.85118
17	0.89435	1.00000	0.76669	0.54583	0.77084
18	0.79483	0.64810	0.49413	1.00000	0.71408
19	0.96908	0.97168	1.00000	0.74966	0.90711
20	0.90357	1.00000	0.60306	0.93133	0.84480

Table A4
Sensitivity Analysis—Subunit and Overall Efficiency Scores

DMU	Aggregate score [Problem 3.3]	Score K1	Score K2	Score K3	Overall score
1	0.61729	0.59829	0.66126	0.45471	0.57142
2	0.78472	0.61945	0.92874	0.34036	0.62952
3	0.63151	0.47002	0.70167	0.56188	0.57786
4	0.63694	0.65130	0.57494	0.67109	0.63244
5	0.34962	0.40929	0.24349	0.31089	0.32123
6	0.59293	0.42169	0.70404	0.48723	0.53765
7	0.99880	0.92883	1.00000	1.00000	0.97628
8	0.55111	0.20955	0.34737	0.73556	0.43082
9	0.90467	0.56048	1.00000	0.85462	0.80503
10	0.83809	0.52589	1.00000	0.56174	0.69588
11	0.83851	0.74255	0.92870	0.71739	0.79621
12	0.92847	1.00000	0.93097	0.25792	0.72963
13	0.59750	0.73010	0.33542	0.45486	0.50679
14	0.60198	0.52012	0.45543	0.71059	0.56205
15	0.89839	0.68224	0.79987	1.00000	0.82737
16	0.95824	1.00000	1.00000	0.55354	0.85118
17	0.91512	1.00000	0.76672	0.54583	0.77085
18	0.84434	0.64810	0.49414	1.00000	0.71408
19	0.97427	0.97182	1.00000	0.74968	0.90717
20	0.91529	1.00000	0.60307	0.93133	0.84480

Table A5
Efficiency Scores – Conventional DEA Model

DMU	Efficiency score
1	0.71104
2	0.91729
3	0.69367
4	0.92629
5	0.48891
6	0.72387
7	1.00000
8	0.87044
9	1.00000
10	1.00000
11	0.94440
12	1.00000
13	0.78425
14	0.85418
15	1.00000
16	1.00000
17	1.00000
18	1.00000
19	1.00000
20	1.00000

Appendix B

VBA Code for Generating Maximal Bundles

Module1

Option Base 1

```
Private Type ArrayUDT
    arrElements() As String
End Type
```

```
Public ArrayOfArrays() As ArrayUDT
Public NumberOfOutputs, OutputInt As Integer
Public InputArray() As String
Public OutputArray() As String
Public Impact As Boolean
```

```
'It creates an array for each output to store all inputs affecting that output
'It then compares each array with all other arrays
'It stores all outputs with identical input arrays in the OutputBundle array
'and displays the maximal bundle.
```

```
Sub main()
```

```
Dim i, j, k, z, m, q, r, count As Integer
Dim Matched As Boolean
Dim OutputString As String
Dim InputString As String
Dim OutputBundle() As String
```

```
Impact = False
UserForm1.Show
```

```
If NumberOfOutputs = 0 Then
    End
End If
```

```
'Creates an array of arrays whose size is equal to the number of outputs
ReDim Preserve ArrayOfArrays(NumberOfOutputs)
```

```
'Creates an array of size 1 for each element of ArrayOfArrays
For count = 1 To UBound(ArrayOfArrays)
    ReDim Preserve ArrayOfArrays(count).arrElements(1)
Next count
```

```
UserForm2.Show
UserForm3.Show
```

```
'Checks whether any interactions were recorded
If Impact = False Then
    End
End If
```

```
'Compares every element of ArrayOfArrays with all other elements of
'ArrayOfArrays in order to find a match
For i = 1 To UBound(ArrayOfArrays)
    k = i + 1
    j = 1
    z = 1
    Matched = False
    m = 0
```

```
If ArrayOfArrays(i).arrElements(1) = "0" Then
    GoTo Break
End If
```

```

'Creates the OutputBundle array that stores all outputs with identical input
bundles
ReDim Preserve OutputBundle(m + 1)
m = m + 1
OutputBundle(m) = OutputArray(i)

'Compares each input array with all other input arrays until a match is found
Do While k <= UBound(ArrayOfArrays)
  If UBound(ArrayOfArrays(k).arrElements) = UBound(ArrayOfArrays(i).
arrElements) And ArrayOfArrays(k).arrElements(1) <> "0" Then
    Do While z <= UBound(ArrayOfArrays(i).arrElements) And j <=
UBound(ArrayOfArrays(k).arrElements)
      If ArrayOfArrays(i).arrElements(z) = ArrayOfArrays(k).arrElements(j) Then
        z = z + 1
        j = 1
        Matched = True
      Else
        j = j + 1
        Matched = False
      End If
    Loop
    If Matched = True Then
      ReDim Preserve OutputBundle(m + 1)
      m = m + 1
      OutputBundle(m) = OutputArray(k)
      ArrayOfArrays(k).arrElements(1) = "0"
    End If
  End If
  j = 1
  z = 1
  k = k + 1
Loop

'Stores the output bundle of the maximal bundle in OutputString
OutputString = "("
For q = 1 To UBound(OutputBundle)
  OutputString = OutputString & " " & OutputBundle(q)
Next q
OutputString = OutputString & ")"

'Stores the input bundle of the maximal bundle in InputString
InputString = "("
For r = 1 To UBound(ArrayOfArrays(i).arrElements)
  InputString = InputString & " " & ArrayOfArrays(i).arrElements(r)
Next r
InputString = InputString + ")"

'Displays the maximal bundle
MsgBox "(" & InputString & " , " & OutputString & ")"
Erase OutputBundle

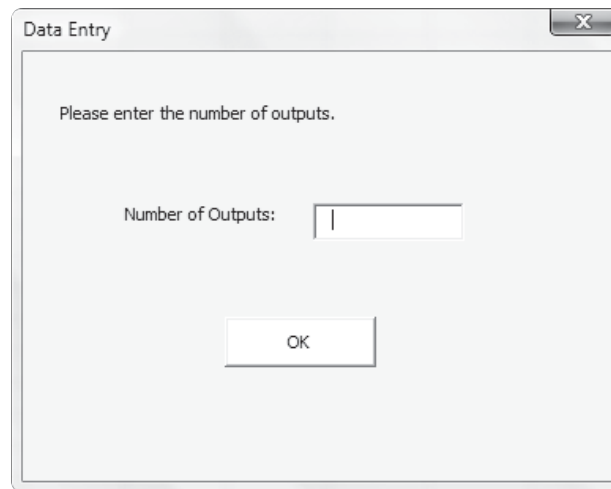
Break:      Next i

Erase ArrayOfArrays
End Sub

```

UserForm1

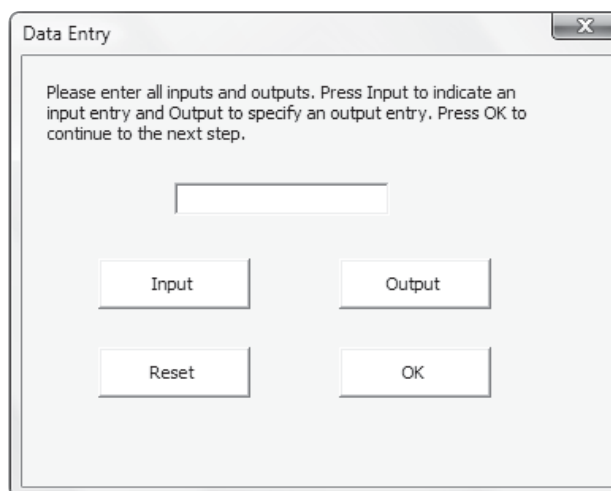
Figure 2.1: UserForm 1

A screenshot of a Windows-style dialog box titled "Data Entry". The dialog box has a close button (X) in the top right corner. The main text inside the dialog box reads "Please enter the number of outputs." Below this text, there is a label "Number of Outputs:" followed by a text input field containing a vertical cursor. At the bottom center of the dialog box, there is an "OK" button.

```
'Prompts the user to enter the number of outputs  
Private Sub OKButton_Click()  
  
If IsNumeric(TextBox1.Value) = False Or TextBox1.Value <= 0 Then  
    MsgBox "Please enter a valid number."  
    GoTo Finish  
End If  
  
NumberOfOutputs = TextBox1.Value  
Unload Me  
  
Finish: End Sub
```

UserForm2

Figure 2.2 : UserForm 2

A screenshot of a Windows-style dialog box titled "Data Entry". The dialog box has a close button (X) in the top right corner. The main text inside the dialog box reads "Please enter all inputs and outputs. Press Input to indicate an input entry and Output to specify an output entry. Press OK to continue to the next step." Below this text, there is a text input field. At the bottom of the dialog box, there are four buttons arranged in a 2x2 grid: "Input", "Output", "Reset", and "OK".

```

Option Base 1
Private CountInput, CountOutput As Integer

'Stores input data received by the user in InputArray

Private Sub InputButton_Click()
    CountInput = CountInput + 1
    ReDim Preserve InputArray(CountInput)
    InputArray(CountInput) = TextBox1.Value
End Sub

'Stores output data received by the user in OutputArray
Private Sub OutputButton_Click()
    CountOutput = CountOutput + 1
    ReDim Preserve OutputArray(CountOutput)
    OutputArray(CountOutput) = TextBox1.Value
End Sub

'Allows the user to clear the entered data
Private Sub ResetButton_Click()
    CountInput = 0
    CountOutput = 0
    Erase InputArray
    Erase OutputArray

End Sub

'Unloads the UserForm
Private Sub OKButton_Click()
    If CountOutput <> NumberOfOutputs Then
        MsgBox "Incorrect Number of Outputs"
        GoTo Finish
    End If

Unload Me
Finish: End Sub

'Initializes the UserForm
Private Sub UserForm_Initialize()
    CountInput = 0
    CountOutput = 0
End Sub

```

UserForm3

Figure 2.3: UserForm 3

The screenshot shows a Windows-style dialog box titled "Input/Output Interactions". Inside the dialog, there is a text area with the following instructions: "Please enter all existing interactions among inputs and outputs. Press Impact after entering each interaction. Press Generate Bundles after all interactions are recorded to display the maximal bundles." Below the text area, there are two text input fields. The first is labeled "Input" and the second is labeled "Output". Below these two fields is a button labeled "Impact". At the bottom of the dialog is a button labeled "Generate Bundles". The dialog has a standard Windows window border with a close button (X) in the top right corner.

Option Base 1

*'Locates the entered output in OutputArray
'and stores the entered input in the proper input array*

Private Sub ImpactButton_Click()

Dim Output As String
Dim i As Integer
Dim Found As Boolean
Dim ErrorFound As Boolean

Aw2Output = TextBox2.Value
Found = False
ValidInput = False

'Checks whether the entered input is valid
For i = 1 To UBound (InputArray)
If InputArray(i) = TextBox1.Value Then
ValidInput = True
Exit For
End If

Next i

If ValidInput = False Then
MsgBox "Please enter a valid input."
GoTo Finish
End If

'Checks whether the entered output is valid
For i = 1 To UBound(OutputArray)
If Output = OutputArray(i) Then
OutputInt = i
Found = True
GoTo Fill
End If
Next i
If Found = False Then
MsgBox "Please enter a valid output."
GoTo Finish
End If

Fill:

Impact = True
If UBound(ArrayOfArrays(OutputInt).arrElements) = 1 And
ArrayOfArrays(OutputInt).arrElements(1) = "" Then
ArrayOfArrays(OutputInt).arrElements(1) = TextBox1.Value
Else
ReDim Preserve ArrayOfArrays(OutputInt).arrElements(UBound(ArrayOfArrays(OutputInt).arrElements) + 1)
ArrayOfArrays(OutputInt).arrElements(UBound(ArrayOfArrays(OutputInt).arrElements)) = TextBox1.Value
End If
Finish: End Sub

'Unloads the UserForm
Private Sub GenerateButton_Click()
Unload Me
End Sub