Gautam Appa and H. Paul Williams

A formula for the solution of DEA models

Working paper

A FORMULA FOR THE SOLUTION OF DEA MODELS

by

Gautam Appa and H. Paul Williams

Department of Operational Research,
The London School of Economics and Political Science,
Houghton Street, London WC2A 2AE, UK

Abstract

When solving Data Envelopment Analysis (DEA) models it is usual to solve a Linear Programme (LP) many times, with different right-hand-side (RHS) vectors: once for each Decision Making Unit (DMU) in the organisation being evaluated. Besides being tedious and involving repeated computation this *iterative* approach gives little insight into the overall structure of the model for the organisation. Instead, by projecting out all the variables of the LP which are common to all LP runs, we obtain a formula into which we can substitute the inputs and outputs of each DMU in turn in order to obtain its efficiency number and efficient comparators. In addition we also obtain the best weightings which the DMU would choose to put on its inputs and outputs. The method of projection, which we use, is Fourier-Motzkin Elimination. This provides us with a finite number of extreme rays of the elimination cone. These rays give the dual multipliers which can be interpreted as weights which will apply to the inputs and outputs for particular DMUs. As the approach provides all the extreme rays of the cone, multiple sets of weights, when they exist, are explicitly provided. The method also demonstrates that the same weightings will apply to all DMUs having the same comparators. In addition it is possible to construct the *skeleton* of the efficient frontier of efficient DMUs.

Keywords: *Data Envelopment Analysis, Efficiency Evaluation, Performance Evaluation, Linear Programming, Fourier-Motzkin elimination.*

1. INTRODUCTION

Data Envelopment Analysis (DEA) is an established method of comparing the performance of a number of similar units (eg shops, hospitals, schools, garages etc). Many case studies have been published (see eg Seiford (1995), Cooper et al (1999))

The individual units are known as Decision Making Units (DMUs). Each DMU has a number of *inputs* used to produce different *outputs*. In the standard model it is assumed that there are *constant returns to scale* (eg doubling all inputs will result in the doubling of all outputs). This implies *infinite divisibility* (ie fractional amounts of the inputs can be used). In addition *additivity* is also assumed, enabling one to make up fictional DMUs by combining DMUs in any multiples. All these assumptions can be altered to produce non-standard models. A full description of DEA can be found in many texts (eg Cooper et al (1999)). For the purposes of this paper we will use the standard model, known in the literature as the CCR model, first introduced in the pioneering work of Charnes et al (1978).

In section 2 we formulate the standard model as a Linear Programme (LP) and show how it is possible to *project out* all but one of the variables to give us weightings for the inputs and outputs resulting in a *formula* which gives us the efficiency number and comparators for each DMU. It also enables us to construct the *skeleton* of the efficiency frontier. This is illustrated, in section 3, by a numerical example. In section 4 we take a larger case study from the literature and apply the analysis to that problem. Finally, in section 5, we suggest how this approach may be extended to non-standard models.

2. PROJECTION OF THE STANDARD MODEL

We consider a problem with *n* DMUs indexed by j in {1,2,…,n}. Each DMU has *p* inputs and *q* outputs. DMUj has inputs $a_{1j}, a_{2j},\ldots,a_{pj}$ and outputs $c_{1j}, c_{2j},\ldots,c_{qj}$.

In the *Primal* model we introduce the following variables:

x_j = Amount of DMUj used.
w = The proportion of the input bundle of DMUk needed to produce its own output bundle.

In order to evaluate DMUk we have the model:

$$\begin{array}{ll}
\text{Minimise} & w \\
\text{subject to:} & -a_{11}x_1 - a_{12}x_2 - \ldots - a_{1n}x_n + a_{1k}w \geq 0 \\
 & -a_{21}x_1 - a_{22}x_2 - \ldots - a_{2n}x_n + a_{2k}w \geq 0 \\
 & \\
 & -a_{p1}x_1 - a_{p2}x_2 - \ldots - a_{pn}x_n + a_{pk}w \geq 0 \\
 & c_{11}x_1 + c_{12}x_2 + \ldots + c_{1n}x_n \geq c_{1k} \\
 & c_{21}x_1 + c_{22}x_2 + \ldots + c_{2n}x_n \geq c_{2k} \\
 & \quad . \qquad . \qquad . \qquad . \\
 & c_{q1}x_1 + c_{q2}x_2 + \ldots + c_{qn}x_n \geq c_{qk} \\
 & x_j \geq 0 \quad j = 1,2,\ldots,n
\end{array}$$

Here a_{ij} is the amount of input i used by DMUj for i = 1, … , p and

c_{tj} is the amount of output t produced by DMUj for t = 1, … , q.

Its interpretation is as follows:

Choose a mixture of DMUs to produce at least the outputs of DMUk using the smallest possible multiple of the inputs of DMUk. If DMUk is *efficient*, its outputs will be best produced using all of its own inputs. In this case $x_k = 1$, $x_j = 0$ for all $j \neq k$ and $w = 1$ (w is its efficiency number). On the other hand, if DMUk is *inefficient*, its outputs will be best produced by a mixture of *other* DMUs using a fraction w of all its inputs. In this case w, its efficiency number, is such that $0 < w < 1$.

If we *project* out variables $x_1, x_2, \ldots, x_n$ using Fourier- Motzkin elimination (see Williams (1986)) this results in the model taking the form:

$$\left(\sum_{i=1}^{p} \lambda_{il}\ a_{ik}\right) w \geq \sum_{t=1}^{q} \mu_{tl}\ c_{ik} \quad \text{for } l = 1,2, \ldots L$$

The optimal value of w is

$$w^* = \underset{l}{Max}\left[\left(\sum_{t=1}^{q} \mu_{tl}\ c_{ik}\right) / \left(\sum_{i=1}^{p} \lambda_{il}\ a_{ik}\right)\right].$$

Let $\underset{l}{arg\max}\ w^* = m$ (1)

w* gives DMUk's efficiency number and λ_{im}, μ_{tm} are the weightings which it should use, for its inputs and outputs, in order to maximise its ratio of weighted outputs to inputs.

It is convenient to consider the inputs and outputs as a matrix.

$$A = \begin{bmatrix} -a_{11} & -a_{12} \ldots -a_{1n} \\ -a_{21} & -a_{22} \ldots -a_{2n} \\ \ldots & \ldots \\ -a_{p1} & -a_{p2} \ldots -a_{pn} \\ c_{11} & c_{12} \ldots\ c_{1n} \\ c_{21} & c_{22} \ldots\ c_{2n} \\ \ldots & \ldots \\ c_{q1} & c_{q2} \ldots\ c_{qn} \end{bmatrix}$$

In order to eliminate the variables $x_1, x_2, \ldots, x_n$ we consider the cone $\mathbf{By} \leq \mathbf{0}$ defined by the matrix **B** such that $\mathbf{B}A \leq \mathbf{0}$.

It is only necessary to consider the (finite number of) extreme rays of this cone, giving rise to the rows of **B**. These are delivered by restricted Fourier-Motzkin elimination. Each extreme ray consists of a set of multipliers (also called weights) for the inputs and outputs. For a particular DMU the 'best' extreme ray(s) provide the maximum possible ratio of outputs to inputs as well as its efficiency number.

The 'best' extreme ray will also give an efficiency number of 1 for certain (efficient) DMUs. These will be the comparator DMUs which we can associate with each extreme ray. From the Duality theorem of LP (see eg. Dantzig(1963)), the variables for these comparator DMUs , in the corresponding Primal model, will be the only ones which are active.

In order to find the multiples of the comparator DMUs which show the current DMU to be inefficient, we solve the binding constraints as *equations* for the variables representing the comparator DMUs.

A particular set of comparator DMUs represent the vertices of a region of the efficient frontier. For a model with p inputs and q outputs this frontier will normally have dimension p + q - 2. The ray from the origin to the point representing the DMU under consideration must pass through one such region, the vertices of which represent its comparators. This is illustrated in the next section with a numerical example.

3. A NUMERICAL EXAMPLE (A)

We consider an example, taken from Cooper et al (1999), with 12 DMUs, each of which has 2 inputs and 2 outputs. These are given in Table 1 below.

DMU	1	2	3	4	5	6	7	8	9	10	11	12
Inputs	20	19	25	27	22	55	33	31	30	50	53	38
	151	131	160	168	158	255	235	206	244	268	306	284
Outputs	100	150	160	180	94	230	220	152	190	250	260	250
	90	50	55	72	66	90	88	80	100	100	147	120

Table 1

The Primal model is:

Minimise w

$$
\begin{array}{lll}
\text{subject to:} & -20x_1 - 19x_2 - \ldots - 38x_{12} + a_{1k}w & \geq 0 \\
 & -151x_1 - 131x_2 - \ldots - 284x_{12} + a_{2k}w & \geq 0 \\
 & 100x_1 + 150x_2 + \ldots + 250x_{12} & \geq c_{1k} \\
 & 90x_1 + 50x_2 + \ldots + 120x_{12} & \geq c_{2k} \\
 & & \\
 & x_1, x_2, \ldots, x_{12} \geq 0 &
\end{array}
$$

The relevant extreme rays of the cone associated with the corresponding matrix **B**, together with the (efficient) comparators, are given in table 2 below.

Ray	Extreme Rays (Weights for Inputs and Outputs)	Comparator DMUs
1	(0.808, 0.797, 0.465, 1)	1, 2, 4
2	(0, 1.111, 0.637, 1)	2, 4
3	(0, 0.867, 0.409, 1)	1,4
4	(7.727, 0, 0.645, 1)	1, 2
5	(0, 0.596, 0, 1)	1
6	(4.5, 0, 0, 1)	1
7	(0, 1.145, 1, 0)	2
8	(7.899, 0, 1, 0)	2

Table 2

N.B. Since these vectors represent rays, they can be scaled by any positive multiplier and still represent the same ray. For convenience we have scaled the multiplier for the second output to be 1, where non-zero, and otherwise scaled the first output multiplier to be 1.

The Efficiency Number of DMUk with inputs a_{1k}, a_{2k} and outputs c_{1k}, c_{2k} is given by:

$$\text{Max}\ [0.465c_{1k}+c_{2k} / 0.808a_{1k}+0.797a_{2k};\ 0.637c_{1k}+c_{2k} / 1.111a_{2k};\ \ldots ;\ c_{1k} / 7.899a_{1k}] \qquad (2)$$

Clearly the multipliers in (2) are the coefficients from the extreme rays above. (Note that the coefficients in the numerator and denominator of any of the ratios in (2) can be scaled by equal amounts).

Table 3 gives all the ratios in (2) for all 12 DMUs.

RAY DMU	1	2	3	4	5	6	7	8	MAX
1	1	0.9161	1.0000	1.0000	1.0000	1.0000	0.5784	0.6333	1
2	1	1	0.9807	1.0000	0.6404	0.5848	1	1	1
3	0.8761	0.8827	0.8685	0.8193	0.5767	0.4889	0.8733	0.8107	0.8827
4	1	1	1	0.9020	0.7190	0.5926	0.9357	0.8444	1
5	0.7635	0.7170	0.7626	0.7451	0.7008	0.6667	0.5196	0.5412	0.7635
6	0.7953	0.8348	0.8328	0.5611	0.5922	0.3636	0.7877	0.5297	0.8348
7	0.8895	0.8738	0.8738	0.9020	0.6283	0.5926	0.8176	0.8444	0.9020
8	0.7963	0.7726	0.7962	0.7435	0.6516	0.5735	0.6444	0.6211	0.7963
9	0.8612	0.8153	0.8402	0.9604	0.6876	0.7407	0.6801	0.8022	0.9604
10	0.8515	0.8706	0.8706	0.6765	0.6260	0.4444	0.8147	0.6333	0.8706
11	0.9345	0.9195	0.9551	0.7687	0.8060	0.6164	0.7420	0.6214	0.9551
12	0.9191	0.8850	0.9028	0.9582	0.7089	0.7018	0.7688	0.8333	0.9582

Table 3

The argument of (2), giving rise to the maximum, corresponds to one of the extreme rays in Table 3. Then the comparators can be read off from table 2. If the argument of (2) is not unique, (for example it is 2 or 3 for DMU10) multiple sets of weights are applicable and can be read off from Table 2.

Example 1: DMU 3 has inputs 25 and 160 and outputs 160 and 55. Substituting in (2) the maximum ratio comes from the 2nd argument:

$$(0.637*160 + 1*55) / (1.111*160) = 0.883 \qquad (3)$$

showing DMU3 to be inefficient with an efficiency score of 0.883. Its comparators are DMU2 and DMU4. [Note that the maximum value in Table 3 is .8827 rather than .883 simply because four digit accuracy is used.]

The multipliers 0.637 and 1 for the outputs and 0 and 1.111 for the inputs in (3) are the weightings which DMU3 should choose in order to maximise its ratio of weighted outputs to inputs.

In order to find the quantities of the comparators DMU2 and DMU4 with which DMU3 should be compared, we note that input 2 and outputs 1 and 2 have positive weightings. We therefore solve the corresponding inequalities in the Primal model as equations to give:

$$\begin{aligned} 131x_2 + 168x_4 - 160w &= 0 \\ 150x_2 + 180x_4 &= 160 \\ 50x_2 + 72x_4 &= 55 \end{aligned}$$

leading to $x_2 = 0.9$, $x_4 = 0.139$, $w = 0.883$;
ie the artificial DMU made up from these quantities of DMU2 and DMU4 produces the same outputs as DMU3 using no more than 88.3% of all the inputs.

Example 2: DMU1 has inputs 20 and 151 and outputs 100 and 90. Substituting in (2) the maximum ratio comes from arguments 1,3,4,5 and 6, giving eg
$(0.465*100 + 90) / (0.808*20 + 0.797*151) = 1$ if we use argument 1, and showing DMU1 to be efficient (efficiency number 1). Clearly one of its efficient 'comparators' is itself.

DMU1 could also choose the weightings from any of the rays 3,4,5 and 6 to maximise its weighted ratio of outputs to inputs.

Example 3: From table 3 it is clear that DMU10 achieves its maximum ratio from arguments 2 and 3. So its comparators could be DMU2 and DMU4 or DMU1 and DMU4. In fact the primal solution is degenerate with $w = 0.8706$ achieved by letting $x_4 = 25/18 = 1.3889$ and letting all other $x_j = 0$. Using DMU4 at level 1.3889 uses 37.5 of input 1 and 233.33 of input 2 and produces 250 of output 1 and 100 of output 2. So the level of output is the same as DMU10 but only 0.8706 of input 2 is required and even less (the slack is active) of input 1. The dual has multiple optimal solutions – the weights corresponding to ray 2 or ray 3 are applicable.

Figure 1 gives the skeleton of the (two dimensional) efficiency frontier.
The *efficient* DMUs on this frontier, viz., 1,2 and 4 are marked.

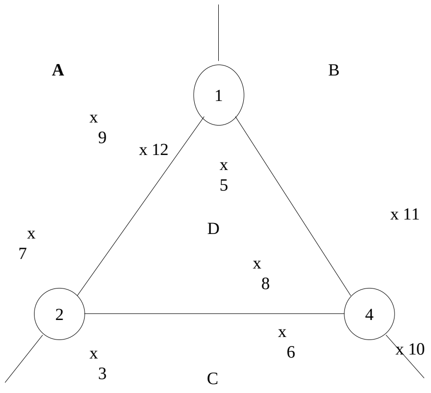

Figure 1

The positions on the skeleton where the rays from the origin to a particular DMU intersect are shown. For example it can be seen that DMU3 intersects the region C showing that the efficient DMUs 2 and 4 are its comparators.

Within each region the same set of multipliers will apply for *all* DMUs whose intersection point lies in this region. The regions are marked and the corresponding input and output multipliers which apply are given below in table 4.

Region	Input Multipliers		Output Multipliers	
A	7.727	0	6.45	1
B	0	0.867	0.409	1
C	0	1.111	0.637	1
D	0.808	0.797	0.465	1

Table 4

In addition DMUs 1 and 2 also each have alternate multipliers arising from rays 5, 6 and 7,8 respectively.

Where a DMU's intersection point is on the boundary of different regions the multipliers for *each* region are alternatives. For example 3 above it was seen that both the multipliers for regions B and C apply. For example 2 the multipliers for regions A, B and D as well as those arising from rays 5 and 6 are applicable. This corresponds with the LP result that a degenerate primal solution has alternate corresponding dual solutions. For this model, with 2 inputs and 2 outputs the primal model will produce at most 3 comparators for any DMU. If there are fewer (as must be the case, for example with an efficient DMU) then the solution is degenerate and there are alternate sets of multipliers (dual values) are possible.

4. A PRACTICAL EXAMPLE (B)

We consider a problem discussed by Harris et al (2000). This concerns measuring the efficiencies of hospitals. There are 4 inputs (service mix, number of beds, number of employees, operational expenses) and 2 outputs (inpatient discharges, outpatients) applying to 20 hospitals (DMUs). The full list of inputs and outputs is given in table 5 below.

DMU	Input	Input	Input	Input	Output	Output
1	55	313	61.15	28.85463	73.0734	41.136
2	42	380	113.3	42.30335	86.6436	28.305
3	25	284	36.4	13.84643	34.2916	14.549
4	71	497	224.85	95.57345	265.55	150.325
5	56	379	119.75	77.52051	173.162	53.741
6	40	230	71.1	33.86819	85.3403	73.162
7	48	215	61.5	22.03391	80.5706	84.347
8	61	264	114	39.21953	148.3293	155.287
9	60	725	359.2	139.6945	414.764	172.088
10	55	358	144.15	55.09739	183.5155	122.371
11	62	509	242.3	87.30443	256.3207	159.644
12	67	357	104.8	33.31825	132.0741	199.955
13	36	120	45.7	10.33024	42.72	23.724
14	62	694	283.2	109.7963	335.9808	326.805
15	24	152	39.4	14.20126	67.7379	45.949
16	75	671	252	85.03626	286.1716	429.329
17	52	399	141.55	50.29971	144.7157	102.422
18	79	822	447.55	146.7919	501.5636	419.729
19	47	325	130.05	42.52691	178.8929	49.649
20	38	284	67.5	30.13273	101.8504	50.704

Table 5

We have scaled the data presented in the original paper in order to give figures of comparable orders of magnitude. This should reduce any potential numerical inaccuracy.

Applying our method to project out the 20 x_j variables from the Primal model we obtain 60 non-trivial extreme rays. These are listed below, in table 6, together with the corresponding comparators.

RAY		INPUT WEIGHTS			OUTPUT WEIGHTS		COMPARATOR DMUs
1	1.000	0.000	0.592	0.236	0.719	0.043	{ 9 14 15 18}
2	1.000	0.000	0.549	0.608	0.698	0.152	{ 14 15 16 18}
3	1.000	0.000	0.506	0.495	0.746	0.010	{ 9 15 18 19}
4	0.112	0.052	0.000	1.000	0.299	0.100	{ 8 12 15 16}
5	0.157	0.061	0.000	1.000	0.337	0.096	{ 8 15 16 18}
6	0.102	0.091	0.000	1.000	0.416	0.050	{ 8 15 18 19}
7	0.000	0.054	0.000	1.000	0.276	0.081	{ 8 12 15}
8	0.000	0.134	0.000	1.000	0.253	0.240	{ 8 12 16}
9	0.000	0.235	0.000	1.000	0.659	0.023	{ 8 18 19}
10	0.000	0.094	0.000	1.000	0.400	0.030	{ 8 15 19}
11	0.188	0.000	0.000	1.000	0.218	0.085	{ 12 15 16}
12	0.369	0.000	0.000	1.000	0.296	0.065	{ 15 16 18}
13	0.404	0.000	0.000	1.000	0.338	0.022	{ 15 18 19}
14	0.000	0.000	0.000	1.000	0.175	0.051	{ 12 15}
15	1.000	0.045	0.755	0.000	0.850	0.066	{ 9 14 15 18}
16	1.000	0.000	0.698	0.000	0.731	0.043	{ 9 14 15}
17	1.000	0.000	0.396	0.000	0.464	0.055	{ 9 14 18}
18	0.859	0.143	0.960	0.000	1.000	0.270	{ 14 15 16 18}
19	1.000	0.000	0.887	0.000	0.727	0.210	{ 14 15 16}
20	1.000	0.000	0.119	0.000	0.134	0.156	{ 14 16 18}
21	0.794	0.131	0.777	0.000	1.000	0.039	{ 9 15 18 19}
22	0.175	0.137	1.000	0.000	0.707	0.361	{ 8 12 15 16}
23	0.576	0.206	0.916	0.000	1.000	0.294	{ 8 15 16 18}
24	0.000	0.802	0.000	0.000	0.381	1.000	{ 8 16}
25	0.333	0.890	0.000	0.000	0.675	1.000	{ 8 16 18}
26	0.260	0.249	0.688	0.000	1.000	0.075	{ 8 15 18 19}
27	0.000	0.140	1.000	0.000	0.672	0.331	{ 8 12 15}
28	0.000	0.265	1.000	0.000	0.636	0.578	{ 8 12 16}
29	0.000	0.381	0.426	0.000	1.000	0.005	{ 8 18 19}
30	0.000	0.928	0.000	0.000	1.000	0.623	{ 8 18}
31	0.000	0.268	0.716	0.000	1.000	0.026	{ 8 15 19}
32	0.373	0.000	1.000	0.000	0.496	0.322	{ 12 15 16}
33	1.000	0.000	0.000	0.000	0.026	0.158	{ 16 18}
34	1.000	0.000	0.000	0.000	0.132	0.030	{ 9 18}
35	0.000	0.000	1.000	0.000	0.410	0.254	{ 12 15}
36	0.000	0.141	0.000	1.000	0.000	0.419	{ 12 16}
37	0.196	0.000	0.000	1.000	0.000	0.232	{ 12 16}
38	0.000	0.000	0.000	1.000	0.000	0.167	{ 12}
39	0.000	0.275	0.972	0.000	0.000	1.000	{ 12 16}
40	1.000	0.000	0.000	0.000	0.000	0.175	{ 16}
41	0.000	0.640	0.000	0.000	0.000	1.000	{ 16}
42	0.392	0.000	1.000	0.000	0.000	0.655	{ 12 16}
43	0.000	0.000	1.000	0.000	0.000	0.524	{ 12}
44	1.000	0.000	0.451	0.559	0.723	0.000	{ 9 18 19}
45	1.000	0.000	0.549	0.421	0.762	0.000	{ 9 15 19}
46	0.000	0.309	0.000	1.000	0.799	0.000	{ 18 19}
47	0.000	0.066	0.000	1.000	0.357	0.000	{ 15 19}
48	0.522	0.000	0.000	1.000	0.375	0.000	{ 18 19}
49	0.307	0.000	0.000	1.000	0.318	0.000	{ 15 19}
50	0.000	0.000	0.000	1.000	0.210	0.000	{ 15}
51	0.931	0.083	0.831	0.000	1.000	0.000	{ 9 15 19}
52	1.000	0.000	0.737	0.000	0.783	0.000	{ 9 15}

53	0.597	0.218	0.615	0.000	1.000	0.000	{ 9 18 19}
54	0.000	0.385	0.414	0.000	1.000	0.000	{ 18 19}
55	0.000	0.610	0.000	0.000	1.000	0.000	{ 18}
56	0.000	0.253	0.744	0.000	1.000	0.000	{ 15 19}
57	1.000	0.118	0.000	0.000	0.350	0.000	{ 9 18}
58	1.000	0.000	0.000	0.291	0.243	0.000	{ 9 18}
59	1.000	0.000	0.000	0.000	0.145	0.000	{ 9}
60	0.000	0.000	1.000	0.000	0.582	0.000	{ 15}

Table 6

Since these rows represent *rays*, they may each be scaled by any positive quantity we wish. We have scaled them all to be between 0 and 1.

The efficiency frontier is 4- dimensional. Therefore its skeleton graph will not, normally, be planar. It is, therefore, not possible to observe visually, for a given DMU, what its comparators are. Its intersection point will lie in one of the 4-dimensional regions. The skeleton graph is presented in figure 2.

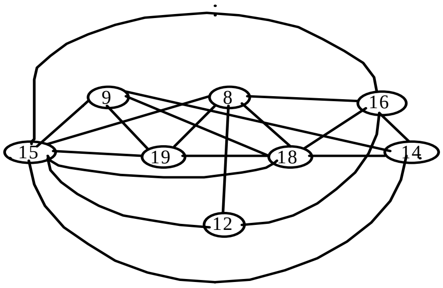

Figure 2

Our solution differs somewhat from that given in the original paper. We believe that the difference is due to numerical accuracy problems for this particular data set involving very large and very small numbers.

Example 4: We deduce the efficiency of hospital (DMU) 1.

Applying the multipliers given in the extreme rays above, to its inputs and outputs, the maximum ratio comes from the multipliers in ray 60 which gives an efficiency score of 0.695 showing hospital 1 to be inefficient with hospital 15 being the comparator.

Example 5: We deduce the efficiency of hospital 5.

Applying the multipliers in the extreme rays above, to its inputs and outputs, the maximum ratio comes from the multipliers in ray 51, which gives an efficiency score of 0.945 showing hospital 5 to be (slightly) inefficient with hospitals 9, 15 and 19 being comparators.

Example 6: We deduce the efficiency of hospital 8.

Applying the multipliers in the extreme rays above, to its inputs and outputs, the maximum ratio comes from ray 22 (among others) giving an efficiency score of 1 showing hospital 8 to be efficient.

All the DMUs with their efficiencies, the ray of their input and output multipliers (there may be alternatives) and their comparators are listed in table 7 below.

DMU	**EFFICIENCY**	**RAY**	COMPARATORS
1	0.695	60	{ 15}
2	0.541	45	{ 9 15 19}
3	0.548	60	{ 15}
4	0.934	29	{ 8 18 19}
5	0.945	51	{ 9 15 19}
6	0.791	22	{ 8 12 15 16}
7	0.895	27	{ 8 12 15}
8	1.000	22	{ 8}
9	1.000	52	{ 9}
10	0.951	26	{ 8 15 18 19}
11	0.867	26	{ 8 15 18 19}
12	1.000	22	{ 12}
13	0.867	50	{ 15}
14	1.000	2	{ 14}
15	1.000	50	{ 15}
16	1.000	11	{ 16}
17	0.734	1	{ 9 14 15 18}
18	1.000	54	{ 18}
19	1.000	21	{ 19}
20	0.909	52	{ 9 15}

Table 7

5. Further Considerations

For this preliminary exposition we have deliberately kept things simple by looking at the original constant return to scale CCR model. We are aware that the standard constant returns to scale model in modern literature minimises w + ε(T) where T is the sum of all the slack and surplus variables to deal with points on the frontier which have $w = 1$ but $T \neq 0$. Other models such as the variable returns to scale model, models with restrictions on weights etc. are also of great interest for practical work. We have been able to solve all these models using the Restricted Fourier – Motzkin Elimination method.

The projection of a polytope into a smaller dimension can lead to an exponential number of extreme rays. We carried out computational experiments to determine if this is likely to be an impediment to solving very large DEA model in this way. The data sets available at http://www.mscmga.ms.ic.ac.uk/jeb/orlib/deainfo.html and http://www.deazone.com/datasets/FILE1/index.asp were utilised. No serious computational problems were encountered. The lessons learnt are:

(i) As the problem size grows, performance measures such as number of rays, maximum no.of inequalities produced at any iteration, sum of inequalities dealt with, maximum memory used and operating time does grow.

(ii) The effect of increasing number of factors (inputs + outputs) is much greater than that of increasing the number of DMUs.

(iii) A problem with only one input or only one output is easier to solve than a comparable one with same number of factors but multiple inputs as well as outputs.

(iv) For problems tried in our experiments (with up to 70 DMUs and 11 factors) the computational growth is easily manageable.

Based on our experience so far we maintain that the method proposed here is a viable method since

(i) DEA models tend to be small in comparison with the large LP models that arise in other applications.

(ii) The DEA based LP models have a special structure with positive coefficients in one of the row partitions and negative coefficients in the other. Experience with Fourier-Motzkin elimination suggests this will reduce the growth in inequalities when projecting out variables.

(iii) In practical DEA problems the number of factors is usually small.

The calculations for the examples in this paper were carried out by means of the PORTA Version 2, Free Software Foundation, Boston, 1991 Program interfaced to our own C program.

REFERENCES

Charnes, A., Cooper, W.W., Rhodes, E. 1978. Measuring the Efficiency of Decision Making Units. *European Journal of Oper. Res.* 2, 429-444.

Cooper, W. W., Seiford, L.W., Tone, K. 1999. *Data Envelopment Analysis*. Kluwer Academic Publications.

Dantzig, G.B. (1963). *Linear Programming and Extensions,* Princeton University Press, Princeton, New Jersey

Harris II, J. Ozgen H. and Y. Oscan (2000), Do Mergers Enhance the Performance of Hospital Efficiency? *JORS* **51,** 801-11

Seiford, L.M. 1995. *A Bibliography of Data Envelopment Analysis (1978-1995).*

Williams, H.P. (1986), Fourier's Method of Linear Programming and its Dual, *Am.Math Monthly* **93,** 681-94

Acknowledgments

The authors would like to thank Yiannis Mourtos and Atsushi Kamiya for help with the calculations and Dr John Beasley for providing the test problems. The work of the second author was partially supported by Leverhulme Research Fellowship RF&G/9/RFG/2000/0174.