Mathematical Programming in Advanced Analytics

Project 1: Performance Analytics with DEA

Group 2 :
B063241
B060210
B128088
B082983

UNIVERSITY OF EDINBURGH
Business School

0.1 Task 1

0.1.1 Matrix Representation of CCR - IO and CCR - OO Models in Envelopment form

CCR - IO Matrix Format

$$\text{Minimize z} = \begin{bmatrix} 0_{1\times n} & -\varepsilon_{1\times(m+s)} & 1_{1\times 1} \end{bmatrix} \begin{bmatrix} \lambda_{n\times 1} \\ s^-_{m\times 1} \\ s^+_{s\times 1} \\ \theta_k \end{bmatrix}$$

$$\text{s.t.} \quad \begin{bmatrix} X_{m\times n} & I_{m\times m} & 0_{m\times s} & -X_{m\times 1} \\ Y_{s\times n} & 0_{s\times m} & -Is\times s & 0_{s\times 1} \end{bmatrix} \begin{bmatrix} \lambda_{n\times 1} \\ s^-_{m\times 1} \\ s^+_{s\times 1} \\ \theta_k \end{bmatrix} = \begin{bmatrix} 0_{m\times 1} \\ J_{s\times 1} \end{bmatrix} \tag{1}$$

where θ_k is the input oriented efficiency ratio of DMU_k, λ is vector of weights assigned to DMU_j to construct the benchmark for DMU_k, s^- (and s^+) are slacks associated with each input (and output).

Comments on the structure of the constraints matrix

X denotes a matrix of input values with the dimensions of $m \times n$. I is an identity matrix of $m \times m$ dimensions assigning the slack variables s^- to the DMU inputs. 0 $m \times s$ is a matrix of zeros of these dimensions, while, - X is a vector of negative input values of $m \times 1$ dimensions. Y is a matrix of input values of $s \times n$ dimensions and 0 $s \times m$ is a matrix of zeros of these dimensions. -I is a negative identity matrix assigning a s^- slacks to each output of the DMU, and 0 $s \times 1$ is a vector of zeros.

CCR - OO Matrix Format

$$\text{Maximize z} = \begin{bmatrix} 0_{1\times n} & \varepsilon_{1\times(m+s)} & 1_{1\times 1} \end{bmatrix} \begin{bmatrix} \lambda_{n\times 1} \\ s^-_{m\times 1} \\ s^+_{s\times 1} \\ \phi_k \end{bmatrix}$$

$$\text{s.t.} \quad \begin{bmatrix} X_{m\times n} & I_{m\times m} & 0_{m\times(s+1)} & \\ Y_{s\times n} & 0_{s\times m} & -Is\times s & -Y_{s\times 1} \end{bmatrix} \begin{bmatrix} \lambda_{n\times 1} \\ s^-_{m\times 1} \\ s^+_{s\times 1} \\ \phi_k \end{bmatrix} = \begin{bmatrix} X_{m\times 1} \\ 0_{s\times 1} \end{bmatrix} \tag{2}$$

where ϕ_k is the input oriented efficiency ratio of DMU_k, λ is vector of weights assigned to DMU_j to construct the benchmark for DMU_k, s^- (and s^+) are slacks associated with each input (and output).

Comments on the structure of the constraints matrix

X denotes a matrix of input values with the dimensions of $m \times n$. I is an identity matrix of $m \times m$ dimensions assigning the slack variables s^- to the DMU inputs. 0 $m \times (s+1)$ is a matrix of zeros of these dimensions, while. Y is a matrix of input values of $s \times n$ dimensions. 0 $s \times m$ is a matrix of zeros of these dimensions, -I is a negative identity matrix assigning a s^- slacks to each output of the DMU, while -Y is a $s \times 1$ vector of negative input values.

0.1.2 Pseudo Code

Algorithm 1 Executable (RUN.m)

```
procedure EXECUTABLE (RUN.M)

    Read content of provided data file into variable A

    Prepare 3 types of linprog algorithms to use - "dual simplex", "interior point", "interior point legacy"

    Prepare variables to hold sheet names

    Prepare the column headers that are common for several files in order to make the code as DRY(don"t repeat yourself) as possible

    Extract inputs into variable X and output into variable Y

    Extract data dimensions where "n" is the amount of DMUs, "m" is the number of inputs, and "s" is the number of outputs

    Set the value of allowable error to a small number

    Execute each of the files that solve and print out results for each of the required analysis
```

Algorithm 2 Analysis 1 - CCR Input Oriented in Envelopment Form

procedure ANALYSIS 1 - CCR INPUT ORIENTED IN ENVELOPMENT FORM

Declare an empty matrix Z that will hold the results

Declare the base column titles that will be used for the output files in a cell array variables declared as "column_headers" and "column_headers_2"

Declare the objective function "f"
Declare a vector of 0s that serves as a lower bound for the solution vector
For each DMU
Find its Aeq
Find its beq
Solve the optimization problem for the objective function
Append the achieved result to the solution matrix Z
Use the custom function "calculate_benchmarks" to save tuples of the type "benchmark group member 1 (benchmark value)" benchmark group member 2 (benchmark value) ..." into a matrix declared as "benchmarks"

Use the custom function "calculate_frontier_io" to save frontier related information in a matrix variable declared as "eff_frontier"

Use the custom function "classify" to save the classification of each DMU into a vector matrix variable declared as "classification"

Loop from 1 to n using a variable "i" and horizontally append a string entry to the cell array variable "column_headers_2" in the form "λ^i"

Loop from 1 to m using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "s+i" which stands for slack values

Loop from 1 to s using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "s-i" which stands for slack values

Append needed elements ”slacks sum” and ”efficiency score” to the ”column_headers” vector variable

Horizontally concatenate the matrix variable ”benchmarks” with the λ values from the matrix Z

Assign the return value of the custom function ”sum_slacks” to the variable Z

Save the current file name as a string into the variable ”filename”

Execute the custom function ”export” once for each of the result variables - ”Z”, ”benchmarks”, ”eff_frontier”, and ”classification”

0.1.3 Matlab Implementation

Please find the MATLAB code used in Project 1 Task 1 in the appropriate folder on the memory stick, files titled ”RUN.m”, ”env_CCR_IO.m” and ”env_CCR_OO.m”.

In addition, open functions ”calculate_benchmarks.m”, ”calculate_frontier_io.m”, ”calculate_frontier_barnum_io.m”, ”calculate_frontier_oo.m” , ”classify.m”, ”export.m”, ”optimize.m” for all analyses in project 1.

Opening all matlab files and executing ”RUN.m” returns analysis output excel results for all tasks.

0.1.4 Analysis

Score Ranking

The ranking is the relative efficiency of the DMUs (Charnes, Cooper and Rhodes, 1978) The DMU are ranked as follows:

	CCR - IO Envelopment DMU Ranking		CCR - OO Envelopment DMU Ranking		Comparative Analysis
DMU	Efficiency score	Rank	Efficiency score	Rank	Rank difference
1	1	1	1	1	0
4	1	2	1	2	0
15	1	3	1	3	0
21	1	4	1	4	0
28	1	5	1	5	0
36	1	6	1	6	0
44	1	7	1	7	0
22	0,9816	8	1,0188	8	0
40	0,8997	9	1,1115	9	0
27	0,8822	10	1,1336	10	0
7	0,8671	11	1,1533	11	0
3	0,8203	12	1,219	12	0
49	0,8158	13	1,2258	13	0
29	0,8	14	1,25	14	0
31	0,7471	15	1,3385	15	0
30	0,7437	16	1,3447	16	0
18	0,6892	17	1,4509	17	0
42	0,6792	18	1,4723	18	0
26	0,6721	19	1,4878	19	0
24	0,6653	20	1,5032	20	0
8	0,6592	21	1,5171	21	0
25	0,6472	22	1,5452	22	0
2	0,6448	23	1,5508	23	0
50	0,6341	24	1,577	24	0
38	0,6265	25	1,5962	25	0
48	0,615	26	1,6261	26	0
47	0,6134	27	1,6302	27	0
16	0,6127	28	1,6322	28	0
9	0,6066	29	1,6485	29	0
14	0,6022	30	1,6607	30	0
43	0,5717	31	1,7491	31	0
41	0,5691	32	1,7572	32	0
11	0,5616	33	1,7806	33	0
13	0,5591	34	1,7887	34	0
23	0,5095	35	1,9627	35	0
5	0,5039	36	1,9845	36	0
10	0,4904	37	2,0393	37	0
45	0,4839	38	2,0666	38	0
12	0,4784	39	2,0902	39	0
19	0,4755	40	2,1028	40	0
20	0,4672	41	2,1405	41	0
37	0,4621	42	2,1638	42	0
33	0,4576	43	2,1854	43	0
35	0,456	44	2,1928	44	0
32	0,4414	45	2,2655	45	0
34	0,4376	46	2,2854	46	0
46	0,4213	47	2,3735	47	0
39	0,4174	48	2,3957	48	0
6	0,3858	49	2,5918	49	0
17	0,3196	50	3,1286	50	0

Figure 1: CCR-IO Envelopment form DEA Score Ranking

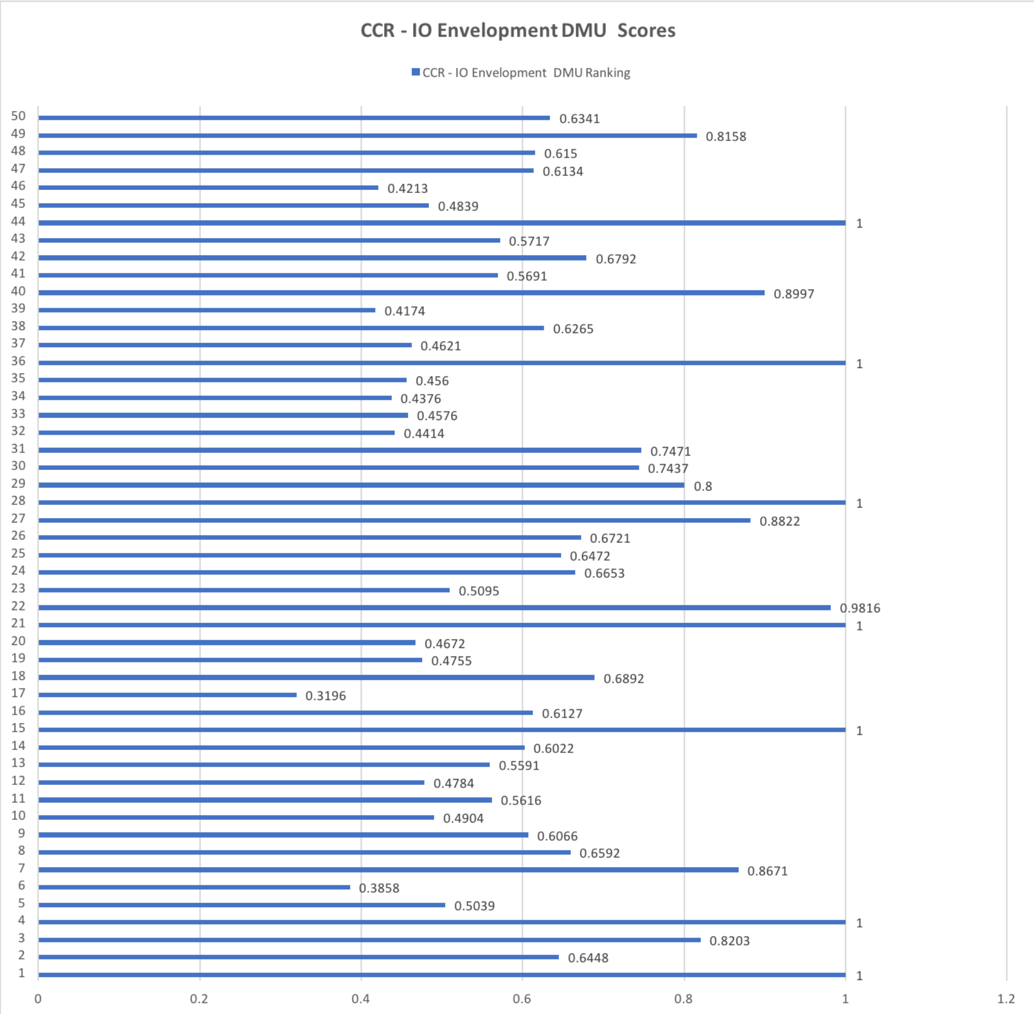

Figure 2: CCR-IO Envelopment form DEA Scores

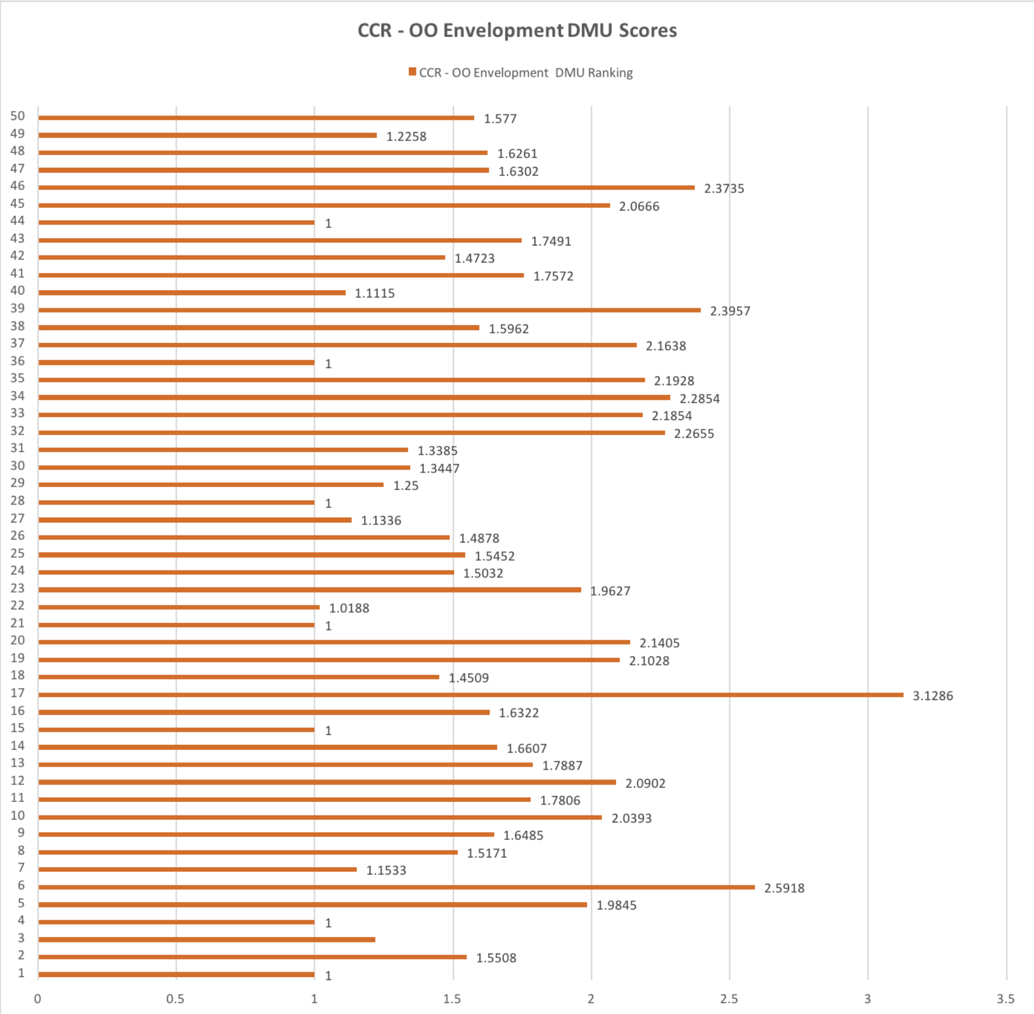

Figure 3: CCR-OO Envelopment form DEA Scores

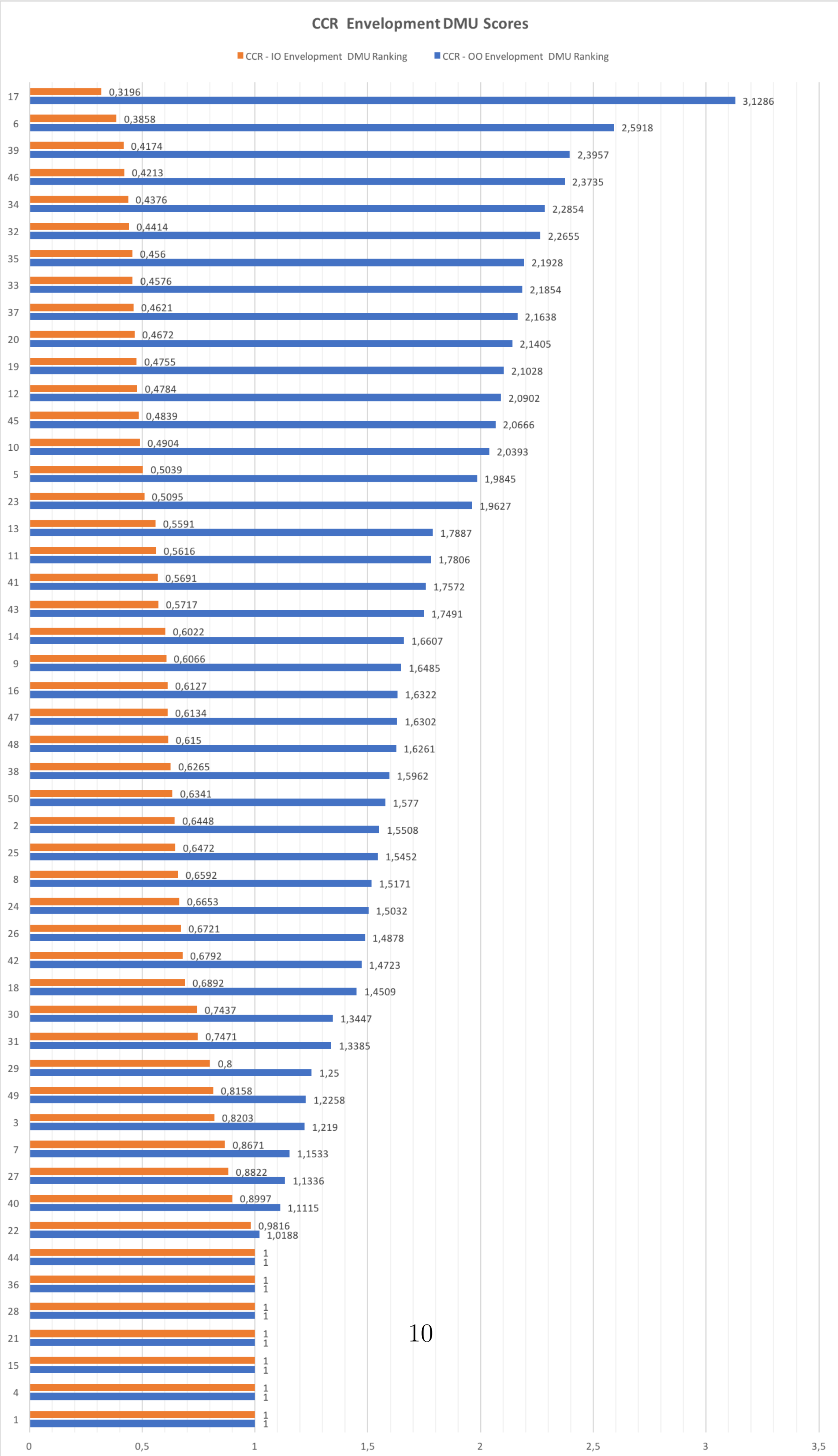

Figure 4: CCR Envelopment form DEA Score Graph

Figure 2 shows the graphical representation of the CCR-IO efficiency scores. For the technically efficienct DMUs the bar reaches 1. The shortfall in efficiency scores for each of the other DMUs is therefore equal to the distance from their current value to the benchmark score of 1. As made obvious by the graphical interpretation, DMU 17 has the lowest efficiency score of the ranked firms. For Investment purposes, the graphical representation can be used to gain an immediate overview of the relative efficiency performance of the potential investments. By using vertical lines as the ones shown, firms that are above a certain efficiency threshold can easily be selected.

Similarly, Figure 3 shows the CCR-OO efficiency scores. Inversely to the IO model, the OO scores are efficient at 1 and inefficient at greater scores. Large inefficiencies, such as in DMU 17, can therefore be plainly observed as having a value significantly greater than 1. For investors selecting firms this is a useful property to observe; DMUs that have values close to 1, such as DMU 27 at 1.1336, do not require much efficiency improvements to achieve technical efficiency. If the firms development trends are favourable, this might therefore be a worthwhile investment. On the other hand, if the DMU has a value significantly greater than 1, the improvements from favourable developments in a DMU´s efficiency will likely still place it far off technical efficiency in the short to medium run.

This means DMU 1, 4, 15, 21, 28, 36 and 44 are the most efficiently run firms and most likely to prove the best investments based on the CCR models.

Comparative analysis (Figure 4) show rankings in the Input and Output oriented models are inversely related. For Technically efficient DMUs, like 21, the bars for IO and OO converge at 1, while for DMUs of inefficient scores, like 17, values for IO and OO models show significant differences.

Slacks

For CCR-IO and CCR- OO s_1^+ are the slacks associated with Input 1: the ratio CurrentLiabilities To Total Assets, s_2^+ are the slacks associated with Input 2: Numbers of Credit intervals, s_1^- are the slacks associated with Output 1: Profit before Tax to Current Liabilities, and s_2^- are the slacks associated with Output 2: Current assets to total liabilities.

If s_1^+ or s_2^+ have positive values it is possible to increase the corresponding inputs with these values without changing the intensity vector or violating any of the program constraints. The inverse applies to s_1^- and s_2^- and the corresponding outputs. Even when efficiency scores are 1, the DMU being evaluated has not achieved "(relative) efficiecy" if slacks are non-zero. In the DMUs with slack values, the input associated with the slack can be changed by the value of the slack in order to eliminate mix-inefficiencies in the DMU. (Charnes. et. al.1978)

CCR-IO

None of the DMUs have slacks associated with Input 1: Ratio of Current Liabilities to Total Assets, meaning no changes can be made in this input for any DMU of without violating the intensity vectors and constraints. DMU 3, 7, 8 , 25 , 27 , 40 and 50 have positive slacks associated with Input 2: Credit intervals, meaning the credit intervals of these DMUs can be increased by the slack to eliminate mix-inefficiencies . DMU 22 and 31 have slack values for s_1^-. Output 1 : Profit before tax to current liabilities can therefore be increased by 0.0189 and 0.2044 respectively to eliminate mix-inefficiencies. DMU 17 has a negative slack of value of 0.078 for s_2^- , which means the Output 2: Current assets to total liabilities can be decreased by 0.078 to eliminate the associated mix-inefficiency.

CCR-OO

As with CCR-IO, none of the DMUs have slacks associated with Input 1: Ratio of Current Liabilities to Total Assets, meaning no changes can be made in this input for any DMU of without violating the intensity vectors and constraints. For CCR-OO DMU 3, 7, 8 , 25 , 27 , 40 and 50 also have positive slacks associated with Input 2: Credit intervals, meaning the credit intervals of these DMUs can be increased with intensities and constraints still holding. The value by which these can be increased is the mix-inefficiency that can be eliminated in the DMU.

DMU 22 and 31 also have slack values for s_1^-. Output 1 : Profit before tax to current liabilities can therefore be increased by 0.0189 and 0.2736 respectively without violating the constraints or intensity vectors. DMU 17 has a negative slack of value of 0.078 for s_2^- , which means the Output 2: Current assets to total liabilities can be decreased by 0.078 with the intensities and constraints still holding.

All slacks for CCR-OO have higher absolute values than for the CCR-IO.

	CCR- IO Slacks Statistics						CCR- OO Slacks Statistics					
DMU	efficiency score	s+_1	s+_2	s-_1	s-_2	Sum of Slacks	efficiency score	s+_1	s+_2	s-_1	s-_2	Sum of Slacks
1	1	0	0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	0	0	1	0	0	0	0	0
15	1	0	0	0	0	0	1	0	0	0	0	0
21	1	0	0	0	0	0	1	0	0	0	0	0
28	1	0	0	0	0	0	1	0	0	0	0	0
36	1	0	0	0	0	0	1	0	0	0	0	0
44	1	0	0	0	0	0	1	0	0	0	0	0
22	0,9816	0	0	0,0182	0	0,0182	1,0188	0	0	0,0186	0	0,0186
40	0,8997	0	27,2683	0	0	27,2683	1,1115	0	30,3079	0	0	30,3079
27	0,8822	0	49,5021	0	0	49,5021	1,1336	0	56,1142	0	0	56,1142
7	0,8671	0	37,0166	0	0	37,0166	1,1533	0	42,6911	0	0	42,6911
3	0,8203	0	0,7277	0	0	0,7277	1,219	0	0,887	0	0	0,887
49	0,8158	0	0	0	0	0	1,2258	0	0	0	0	0
29	0,8	0	0	0	0	0	1,25	0	0	0	0	0
31	0,7471	0	0	0,2044	0	0,2044	1,3385	0	0	0,2735	0	0,2735
30	0,7437	0	0	0	0	0	1,3447	0	0	0	0	0
18	0,6892	0	0	0	0	0	1,4509	0	0	0	0	0
42	0,6792	0	0	0	0	0	1,4723	0	0	0	0	0
26	0,6721	0	0	0	0	0	1,4878	0	0	0	0	0
24	0,6653	0	0	0	0	0	1,5032	0	0	0	0	0
8	0,6592	0	50,1952	0	0	50,1952	1,5171	0	76,1501	0	0	76,1501
25	0,6472	0	1,3029	0	0	1,3029	1,5452	0	2,0132	0	0	2,0132
2	0,6448	0	0	0	0	0	1,5508	0	0	0	0	0
50	0,6341	0	18,7643	0	0	18,7643	1,577	0	29,5922	0	0	29,5922
38	0,6265	0	0	0	0	0	1,5962	0	0	0	0	0
48	0,615	0	0	0	0	0	1,6261	0	0	0	0	0
47	0,6134	0	0	0	0	0	1,6302	0	0	0	0	0
16	0,6127	0	0	0	0	0	1,6322	0	0	0	0	0
9	0,6066	0	0	0	0	0	1,6485	0	0	0	0	0
14	0,6022	0	0	0	0	0	1,6607	0	0	0	0	0
43	0,5717	0	0	0	0	0	1,7491	0	0	0	0	0
41	0,5691	0	0	0	0	0	1,7572	0	0	0	0	0
11	0,5616	0	0	0	0	0	1,7806	0	0	0	0	0
13	0,5591	0	0	0	0	0	1,7887	0	0	0	0	0
23	0,5095	0	0	0	0	0	1,9627	0	0	0	0	0
5	0,5039	0	0	0	0	0	1,9845	0	0	0	0	0
10	0,4904	0	0	0	0	0	2,0393	0	0	0	0	0
45	0,4839	0	0	0	0	0	2,0666	0	0	0	0	0
12	0,4784	0	0	0	0	0	2,0902	0	0	0	0	0
19	0,4755	0	0	0	0	0	2,1028	0	0	0	0	0
20	0,4672	0	0	0	0	0	2,1405	0	0	0	0	0
37	0,4621	0	0	0	0	0	2,1638	0	0	0	0	0
33	0,4576	0	0	0	0	0	2,1854	0	0	0	0	0
35	0,456	0	0	0	0	0	2,1928	0	0	0	0	0
32	0,4414	0	0	0	0	0	2,2655	0	0	0	0	0
34	0,4376	0	0	0	0	0	2,2854	0	0	0	0	0
46	0,4213	0	0	0	0	0	2,3735	0	0	0	0	0
39	0,4174	0	0	0	0	0	2,3957	0	0	0	0	0
6	0,3858	0	0	0	0	0	2,5918	0	0	0	0	0
17	0,3196	0	0	0	0,0752	0,0752	3,1286	0	0	0	0,2353	0,2353
Stats	Efficiency score	s+_1	s+_2	s-_1	s-_2	Sum of Slacks	Efficiency score	s+_1	s+_2	s-_1	s-_2	Sum of Slacks
Min:	0,3227	0	0	0	0	0	0,3227	0	0	0	0	0
Max:	1	0	50,1952	0,2044	0,0782	50,1952	1	0	76,1501	0,2736	0,2425	76,1501
Average:	0,660544	0	3,69554	0,00446	0,001564	3,701566	0,660544	0	4,75511	0,00585	0,00485	4,765814
Standard Dev	0,198278154	0	11,7415	0,02897	0,01105915	11,73960645	0,198278154	0	15,2789	0,03873	0,03429	15,27553927
1st quartile	0,4871	0	0	0	0	0	0,4871	0	0	0	0	0
2nd quartile	0,6239	0	0	0	0	0	0,6166	0	0	0	0	0
3rd quartile	0,8122	0	0	0	0	0	0,8162	0	0	0	0	0
4th quartile	1	0	50,1952	0,2044	0,0782	50,1952	1	0	76,1501	0,2736	0,2425	76,1501
1st decile	0,44269	0	0	0	0	0	0,44228	0	0	0	0	0
2nd decile	0,47512	0	0	0	0	0	0,47324	0	0	0	0	0
3rd decile	0,50837	0	0	0	0	0	0,50654	0	0	0	0	0
4th decile	0,59028	0	0	0	0	0	0,57836	0	0	0	0	0
5th decile	0,6239	0	0	0	0	0	0,6166	0	0	0	0	0
6th decile	0,662	0	0	0	0	0	0,6648	0	0	0	0	0
7th decile	0,74472	0	0	0	0	0	0,74574	0	0	0	0	0
8th decile	0,87012	0	0	0	0	0,00372	0,87314	0	0	0	0	0
9th decile	1	0	3,04904	0	0	3,04904	1	0	1,11224	0	0	1,11224
10th decile	1	0	50,1952	0,2044	0,0782	50,1952	1	0	76,1501	0,2736	0,2425	76,1501

Figure 5: CCR Envelopment form Slacks and Statistics

Intensity Vectors and Benchmarks

The intensity vectors indicate which peer group a DMU is part of, i.e, which other DMUs it is benchmarked against(Cooper et.al, 2007 p.47). The intensity vectors show that all DMUs are benchmarked against a combination of the Technically efficient DMUs 1, 4, 15, 21, 28, 36 and 44, and indicate what weight each of these has been given in the benchmarking.

CCR - IO Envelopment Form Intensity Vectors & Benchmarks

DMU	Benchmarks	λ1	λ2	λ3	λ4	λ5	λ6	λ7	λ8	λ9	λ10	λ11	λ12	λ13	λ14	λ15	λ16	λ17	λ18	λ19	λ20	λ21	λ22	λ23	λ24	λ25	λ26	λ27	λ28	λ29	λ30	λ31	λ32	λ33	λ34	λ35	λ36	λ37	λ38	λ39	λ40	λ41	λ42	λ43	λ44	λ45	λ46	λ47	λ48	λ49	λ50
1	1(1)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	4(0.1212) 21(0.0533) 44(0.525)	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1(0.4147) 4(0.5853)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	4(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4(0.2673) 21(0.024) 44(0.2784)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	4(0.263) 21(0.1595) 44(0.0748)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	28(0.5754) 44(0.3549)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
8	28(0.4724) 44(0.2404)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0287	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1,2572	0	0	0	0	0	0
9	4(0.198) 21(0.14) 44(0.4842)	0	0	0	0,1567	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6566	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	4(0.2131) 21(0.3089) 44(0.1941)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2158	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5919	0	0	0	0	0	0
11	4(0.1813) 28(0.2142) 44(0.2236)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5754	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3549	0	0	0	0	0	0
12	4(0.2677) 21(0.186) 44(0.212)	0,4147	0	0	0,5853	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	4(0.4207) 21(0.0228) 44(0.2298)	0	0	0	0,2952	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,062	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,554	0	0	0	0	0	0
14	4(0.3686) 21(0.0009) 44(0.2114)	0	0	0	0,3129	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0369	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,496	0	0	0	0	0	0
15	15(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0057	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1,0348	0	0	0	0	0	0
16	4(0.1196) 21(0.1315) 44(0.4301)	0	0	0	0,1241	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0009	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,646	0	0	0	0	0	0
17	15(0.273) 21(0.1899)	0	0	0	0,0097	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3966	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3639	0	0	0	0	0	0
18	4(0.0097) 28(0.3966) 44(0.3639)	0	0	0	0,1675	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1801	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5301	0	0	0	0	0	0
19	4(0.1999) 21(0.2369) 44(0.2428)	0	0	0	0,0738	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3512	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,281	0	0	0	0	0	0
20	4(0.197) 21(0.0619) 44(0.27)	0	0	0	0,2009	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,135	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4541	0	0	0	0	0	0
21	21(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4724	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2404	0	0	0	0	0	0
22	21(0.0279) 44(1.2577)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2892	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3991	0	0	0	0	0	0
23	4(0.2465) 21(0.1092) 44(0.3117)	0	0	0	0,121	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0549	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5241	0	0	0	0	0	0
24	4(0.2013) 21(0.131) 44(0.4564)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3384	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1437	0	0	0	0	0	0
25	28(0.2892) 44(0.3991)	0	0	0	0,1411	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6172	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0402	0	0	0	0	0	0
26	4(0.0738) 28(0.3512) 44(0.281)	0	0	0	0,1418	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2709	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4611	0	0	0	0	0	0
27	28(0.2158) 44(0.5919)	0	0	0	0,0732	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2069	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5461	0	0	0	0	0	0
28	28(1)	0	0	0	0,1191	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1355	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4277	0	0	0	0	0	0
29	4(0.3131) 21(0.0359) 44(0.4966)	0	0	0	0,1975	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1442	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4818	0	0	0	0	0	0
30	4(0.1241) 21(0.0008) 44(0.646)	0	0	0	0,3686	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2114	0	0	0	0	0	0
31	21(0.0056) 44(1.0349)	0	0	0	0,2216	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0954	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3596	0	0	0	0	0	0
32	4(0.1905) 21(0.289) 44(0.1995)	0	0	0	0,1607	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1948	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4659	0	0	0	0	0	0
33	4(0.1999) 21(0.1829) 44(0.2231)	0	0	0	0,1813	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2142	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2236	0	0	0	0	0	0
34	4(0.2375) 21(0.265) 44(0.182)	0	0	0	0,4207	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0235	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2294	0	0	0	0	0	0
35	4(0.2784) 21(0.1918) 44(0.2826)	0	0	0	0,2461	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1125	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3097	0	0	0	0	0	0
36	36(1)	0	0	0	0,2672	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0247	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,278	0	0	0	0	0	0
37	4(0.21) 21(0.0051) 44(0.2204)	0	0	0	0,2118	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3191	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1882	0	0	0	0	0	0
38	4(0.1438) 21(0.5954) 44(0.0528)	0	0	0	0,2801	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,222	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0598	0	0	0	0	0	0
39	4(0.2684) 21(0.2656) 44(0.1631)	0	0	0	0,267	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1918	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2086	0	0	0	0	0	0
40	4(0.1567) 28(0.6566)	0	0	0	0,199	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2445	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2385	0	0	0	0	0	0
41	4(0.1614) 21(0.1889) 44(0.4693)	0	0	0	0,1967	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0638	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2689	0	0	0	0	0	0
42	4(0.1682) 21(0.1747) 44(0.5332)	0	0	0	0,21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0053	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2203	0	0	0	0	0	0
43	4(0.222) 21(0.0926) 44(0.3612)	0	0	0	0,1992	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1887	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2197	0	0	0	0	0	0
44	44(1)	0	0	0	0,2777	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1978	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2792	0	0	0	0	0	0
45	4(0.291) 21(0.215) 44(0.0638)	0	0	0	0,1893	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2985	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,194	0	0	0	0	0	0
46	4(0.2567) 21(0.1037) 44(0.2462)	0	0	0	0,2364	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2736	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,177	0	0	0	0	0	0
47	4(0.0739) 21(0.2007) 44(0.5497)	0	0	0	0,2563	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1069	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2444	0	0	0	0	0	0
48	4(0.1429) 21(0.2626) 44(0.4659)	0	0	0	0,2673	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2742	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1581	0	0	0	0	0	0
49	4(0.2954) 21(0.0603) 44(0.5551)	0	0	0	0,2624	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1646	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0719	0	0	0	0	0	0
50	28(0.3384) 44(0.1437)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2702	0	0	0	0	0	0,1984	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Min:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Max:	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1,257	0	0	0	0	0	0
	Average:	0,028294	0	0	0,172984	0	0	0	0	0	0	0	0	0	0	0,025404	0	0	0	0	0	0,1214	0	0	0	0	0	0	0,0901960	0	0	0	0	0	0	0	0,02	0	0	0	0	0	0	0	0,308686	0	0	0	0	0	0
	Standard Deviation:	0,15199010	0	0	0,17776	0	0	0	0	0	0	0	0	0	0	0,14573808	0	0	0	0	0	0,17849126	0	0	0	0	0	0	########	0	0	0	0	0	0	0	0,141420	0	0	0	0	0	0	0	########	0	0	0	0	0	0
	1st quartile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1471	0	0	0	0	0	0
	2nd quartile	0	0	0	0,1744	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,05845	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2424	0	0	0	0	0	0
Statistics	**3rd quartile**	0	0	0	0,25375	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,19405	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4593	0	0	0	0	0	0
	4th quartile	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1,257	0	0	0	0	0	0
	1st decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2nd decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0558	0	0	0	0	0	0
	3rd decile	0	0	0	0,05415	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1848	0	0	0	0	0	0
	4th decile	0	0	0	0,1343	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,01638	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2206	0	0	0	0	0	0
	5th decile	0	0	0	0,1744	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,05845	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2424	0	0	0	0	0	0
	6th decile	0	0	0	0,19908	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1215	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2924	0	0	0	0	0	0
	7th decile	0	0	0	0,22604	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,18268	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4076	0	0	0	0	0	0
	8th decile	0	0	0	0,26704	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2003	0	0	0	0	0	0	0,04284	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4846	0	0	0	0	0	0
	9th decile	0	0	0	0,29697	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,27366	0	0	0	0	0	0	0,35514	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5577	0	0	0	0	0	0
	10th decile	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1,257	0	0	0	0	0	0

Figure 6: CCR-IO Envelopment Intensity Vector and Benchmark Table

CCR - OO Envelopment Form Intensity Vectors

DMU	Benchmarks	λ1	λ2	λ3	λ4	λ5	λ6	λ7	λ8	λ9	λ10	λ11	λ12	λ13	λ14	λ15	λ16	λ17	λ18	λ19	λ20	λ21	λ22	λ23	λ24	λ25	λ26	λ27	λ28	λ29	λ30	λ31	λ32	λ33	λ34	λ35	λ36	λ37	λ38	λ39	λ40	λ41	λ42	λ43	λ44	λ45	λ46	λ47	λ48	λ49	λ50
1	1(1)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	4(0.1212) 21(0.0533) 44(0.525)	0	0	0	0,1879	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0826	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8142	0	0	0	0	0	0
3	1(0.4147) 4(0.5853)	0,5056	0	0	0,7134	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	4(1)	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4(0.2673) 21(0.024) 44(0.2784)	0	0	0	0,5305	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0476	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5525	0	0	0	0	0	0
6	4(0.263) 21(0.1595) 44(0.0748)	0	0	0	0,6817	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4134	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1939	0	0	0	0	0	0
7	28(0.5754) 44(0.3549)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6636	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4093	0	0	0	0	0	0
8	28(0.4724) 44(0.2404)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7167	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3647	0	0	0	0	0	0
9	4(0.198) 21(0.14) 44(0.4842)	0	0	0	0,3264	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2307	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7982	0	0	0	0	0	0
10	4(0.2131) 21(0.3089) 44(0.1941)	0	0	0	0,4345	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3958	0	0	0	0	0	0
11	4(0.1813) 28(0.2142) 44(0.2236)	0	0	0	0,3228	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3814	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3982	0	0	0	0	0	0
12	4(0.2677) 21(0.186) 44(0.212)	0	0	0	0,5597	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3888	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4431	0	0	0	0	0	0
13	4(0.4207) 21(0.0228) 44(0.2298)	0	0	0	0,7526	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0408	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,411	0	0	0	0	0	0
14	4(0.3686) 21(0.0009) 44(0.2114)	0	0	0	0,6121	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0015	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3511	0	0	0	0	0	0
15	15(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	4(0.1196) 21(0.1315) 44(0.4301)	0	0	0	0,1953	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2146	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7019	0	0	0	0	0	0
17	15(0.273) 21(0.1899)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8542	0	0	0	0	0	0,5943	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	4(0.0097) 28(0.3966) 44(0.3639)	0	0	0	0,0141	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5754	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5279	0	0	0	0	0	0
19	4(0.1999) 21(0.2369) 44(0.2428)	0	0	0	0,4204	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4982	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5107	0	0	0	0	0	0
20	4(0.197) 21(0.0619) 44(0.27)	0	0	0	0,4216	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1325	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5779	0	0	0	0	0	0
21	21(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	21(0.0279) 44(1.2577)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0284	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1,2813	0	0	0	0	0	0
23	4(0.2465) 21(0.1092) 44(0.3117)	0	0	0	0,4839	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2143	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6117	0	0	0	0	0	0
24	4(0.2013) 21(0.131) 44(0.4564)	0	0	0	0,3027	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1969	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,686	0	0	0	0	0	0
25	28(0.2892) 44(0.3991)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4469	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6166	0	0	0	0	0	0
26	4(0.0738) 28(0.3512) 44(0.281)	0	0	0	0,1098	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5225	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4181	0	0	0	0	0	0
27	28(0.2158) 44(0.5919)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2446	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6709	0	0	0	0	0	0
28	28(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	4(0.3131) 21(0.0359) 44(0.4966)	0	0	0	0,3913	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0448	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6208	0	0	0	0	0	0
30	4(0.1241) 21(0.0008) 44(0.646)	0	0	0	0,1669	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0011	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8687	0	0	0	0	0	0
31	21(0.0056) 44(1.0349)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0075	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1,3853	0	0	0	0	0	0
32	4(0.1905) 21(0.289) 44(0.1995)	0	0	0	0,4315	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6546	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,452	0	0	0	0	0	0
33	4(0.1999) 21(0.1829) 44(0.2231)	0	0	0	0,4368	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3997	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4875	0	0	0	0	0	0
34	4(0.2375) 21(0.265) 44(0.182)	0	0	0	0,5427	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6057	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4159	0	0	0	0	0	0
35	4(0.2784) 21(0.1918) 44(0.2826)	0	0	0	0,6105	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4207	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6198	0	0	0	0	0	0
36	36(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	4(0.21) 21(0.0051) 44(0.2204)	0	0	0	0,4544	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,011	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4768	0	0	0	0	0	0
38	4(0.1438) 21(0.5954) 44(0.0528)	0	0	0	0,2296	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,9504	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0843	0	0	0	0	0	0
39	4(0.2684) 21(0.2656) 44(0.1631)	0	0	0	0,6429	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6364	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3907	0	0	0	0	0	0
40	4(0.1567) 28(0.6566)	0	0	0	0,1742	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7298	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	4(0.1614) 21(0.1889) 44(0.4693)	0	0	0	0,2836	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,332	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8246	0	0	0	0	0	0
42	4(0.1682) 21(0.1747) 44(0.5332)	0	0	0	0,2476	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2573	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7851	0	0	0	0	0	0
43	4(0.222) 21(0.0926) 44(0.3612)	0	0	0	0,3883	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1619	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6318	0	0	0	0	0	0
44	44(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
45	4(0.291) 21(0.215) 44(0.0638)	0	0	0	0,6014	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4444	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1319	0	0	0	0	0	0
46	4(0.2567) 21(0.1037) 44(0.2462)	0	0	0	0,6094	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2463	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5844	0	0	0	0	0	0
47	4(0.0739) 21(0.2007) 44(0.5497)	0	0	0	0,1205	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3271	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8962	0	0	0	0	0	0
48	4(0.1429) 21(0.2626) 44(0.4659)	0	0	0	0,2323	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,427	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7576	0	0	0	0	0	0
49	4(0.2954) 21(0.0603) 44(0.5551)	0	0	0	0,3621	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0739	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6804	0	0	0	0	0	0
50	28(0.3384) 44(0.1437)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5337	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2267	0	0	0	0	0	0
Stats	Min:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Max:	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1,3853	0	0	0	0	0	0
	Average:	0,0301	0	0	0,2999	0,0000	0	0	0	0	0	0	0	0	0	0,0371	0	0	0	0	0	0,2143	0	0	0	0	0	0	0,1161	0	0	0	0	0	0	0	0,0200	0	0	0	0	0	0	0	0,4811	0	0	0	0	0	0
	Standard Deviation:	0,1572	0	0	0,2607	0,0000	0	0	0	0	0	0	0	0	0	0,1841	0	0	0	0	0	0,2648	0	0	0	0	0	0	0,2517	0	0	0	0	0	0	0	0,1414	0	0	0	0	0	0	0	0,3327	0	0	0	0	0	0
	1st quartile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2578	0	0	0	0	0	0
	2nd quartile	0	0	0	0,3027	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0826	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4875	0	0	0	0	0	0
	3rd quartile	0	0	0	0,4839	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3997	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6804	0	0	0	0	0	0
	4th quartile	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1,3853	0	0	0	0	0	0
	1st decile	0,0000	0	0	0,0000	0,0000	0	0	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0,0000	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0
	2nd decile	0,0000	0	0	0,0000	0,0000	0	0	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0,0000	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,1129	0	0	0	0	0	0
	3rd decile	0,0000	0	0	0,1141	0,0000	0	0	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0,0004	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,3927	0	0	0	0	0	0
	4th decile	0,0000	0	0	0,2022	0,0000	0	0	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0,0309	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,4163	0	0	0	0	0	0
	5th decile	0,0000	0	0	0,3027	0,0000	0	0	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0,0826	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,4875	0	0	0	0	0	0
	6th decile	0,0000	0	0	0,3907	0,0000	0	0	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0,2145	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,5831	0	0	0	0	0	0
	7th decile	0,0000	0	0	0,4359	0,0000	0	0	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0,3300	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,6274	0	0	0	0	0	0
	8th decile	0,0000	0	0	0,5495	0,0000	0	0	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0,4232	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,0000	0	0	0	0	0	0	0	0,7242	0	0	0	0	0	0
	9th decile	0	0	0	0,61826	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,61056	0	0	0	0	0	0	0,53308	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,83342	0	0	0	0	0	0
	10th decile	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1,3853	0	0	0	0	0	0

Figure 7: CCR-OO Envelopment Intensity Vector and Benchmarks Table

Correlation Analysis

CCR IO and OO Envelopment Form Correlation Analysis					
s+_1	s+_2	s-_1	s-_2	slacks sum	efficiency score
#DIV/0!	0,988207369	0,99978021	1	0,988201497	-0,951819415

Figure 8: CCR Envelopment form IO and OO Correlation Analysis

The correlation analysis of slacks of CCR - IO and OO reveals an very close relationship between the scores of the two models, with correlation coefficients very close to 1. The efficiency scores has a correlation coefficient close to -1, which means the efficiency scores of the IO and OO models are inversely related.

Classification

The DMUs are classified into (A) Fully efficient, (B) Technically Efficient but mix inefficient, (C) Technically inefficient but mix efficient and (D) Technically inefficient and mix inefficient. Classification analysis reveals most DMUs (firms) are Technically inefficient but mix efficient, where their projections onto the efficiency frontier can be reached by scaling their current input / output mix.

CCR - IO Envelopment DMU Classification							
DMU	**Classification**	**s+_1**	**s+_2**	**s-_1**	**s-_2**	**slacks sum**	**efficiency score**
1	(A) Fully efficient	0	0	0	0	0	1
4	(A) Fully efficient	0	0	0	0	0	1
15	(A) Fully efficient	0	0	0	0	0	1
21	(A) Fully efficient	0	0	0	0	0	1
28	(A) Fully efficient	0	0	0	0	0	1
36	(A) Fully efficient	0	0	0	0	0	1
44	(A) Fully efficient	0	0	0	0	0	1
2	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6448
5	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,5039
6	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,3858
9	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6066
10	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4904
11	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,5616
12	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4784
13	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,5591
14	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6022
16	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6127
18	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6892
19	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4755
20	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4672
23	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,5095
24	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6653
26	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6721
29	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,8
30	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,7437
32	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4414
33	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4576
34	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4376
35	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,456
37	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4621
38	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6265
39	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4174
41	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,5691
42	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6792
43	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,5717
45	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4839
46	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,4213
47	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,6134
48	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,615
49	(C) Technically inefficient but mix-efficient	0	0	0	0	0	0,8158
3	(D) Technically inefficient and mix-inefficient	0	0,7277	0	0	0,7277	0,8203
7	(D) Technically inefficient and mix-inefficient	0	37,017	0	0	37,0166	0,8671
8	(D) Technically inefficient and mix-inefficient	0	50,195	0	0	50,1952	0,6592
17	(D) Technically inefficient and mix-inefficient	0	0	0	0,0752	0,0752	0,3196
22	(D) Technically inefficient and mix-inefficient	0	0	0,0182	0	0,0182	0,9816
25	(D) Technically inefficient and mix-inefficient	0	1,3029	0	0	1,3029	0,6472
27	(D) Technically inefficient and mix-inefficient	0	49,502	0	0	49,5021	0,8822
31	(D) Technically inefficient and mix-inefficient	0	0	0,2044	0	0,2044	0,7471
40	(D) Technically inefficient and mix-inefficient	0	27,268	0	0	27,2683	0,8997
50	(D) Technically inefficient and mix-inefficient	0	18,764	0	0	18,7643	0,6341

Figure 9: CCR-IO Envelopment form IO Classification Table

CCR - OO Envelopment DMU Classification							
DMU	Classification	s-_1	s-_2	s+_1	s+_2	slacks sum	efficiency scor
1	(A) Fully efficient	0	0	0	0	0	1
4	(A) Fully efficient	0	0	0	0	0	1
15	(A) Fully efficient	0	0	0	0	0	1
21	(A) Fully efficient	0	0	0	0	0	1
28	(A) Fully efficient	0	0	0	0	0	1
36	(A) Fully efficient	0	0	0	0	0	1
44	(A) Fully efficient	0	0	0	0	0	1
49	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,2258
29	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,25
30	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,3447
18	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,4509
42	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,4723
26	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,4878
24	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,5032
2	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,5508
38	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,5962
48	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,6261
47	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,6302
16	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,6322
9	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,6485
14	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,6607
43	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,7491
41	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,7572
11	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,7806
13	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,7887
23	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,9627
5	(C) Technically inefficient but mix-efficient	0	0	0	0	0	1,9845
10	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,0393
45	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,0666
12	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,0902
19	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,1028
20	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,1405
37	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,1638
33	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,1854
35	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,1928
32	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,2655
34	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,2854
46	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,3735
39	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,3957
6	(C) Technically inefficient but mix-efficient	0	0	0	0	0	2,5918
22	(D) Technically inefficient and mix-inefficient	0	0	0,0186	0	0,0186	1,0188
40	(D) Technically inefficient and mix-inefficient	0	30,308	0	0	30,3079	1,1115
27	(D) Technically inefficient and mix-inefficient	0	56,114	0	0	56,1142	1,1336
7	(D) Technically inefficient and mix-inefficient	0	42,691	0	0	42,6911	1,1533
3	(D) Technically inefficient and mix-inefficient	0	0,887	0	0	0,887	1,219
31	(D) Technically inefficient and mix-inefficient	0	0	0,2735	0	0,2735	1,3385
8	(D) Technically inefficient and mix-inefficient	0	76,15	0	0	76,1501	1,5171
25	(D) Technically inefficient and mix-inefficient	0	2,0132	0	0	2,0132	1,5452
50	(D) Technically inefficient and mix-inefficient	0	29,592	0	0	29,5922	1,577
17	(D) Technically inefficient and mix-inefficient	0	0	0	0,2353	0,2353	3,1286

Figure 10: CCR-OO Envelopment form IO Classification Table

The classification tables show the classification of DMU into (A) Fully efficient, (B) Technically efficient but Mix-inefficient, (C) Technically inefficienct but mix-efficient and (D) Technically inefficient and mix-inefficient categories. It also reveals the relationship between this classification, the efficiency scores and the slacks. First, the classifications are identical for both IO and OO models. This is directly related to the observation that slacks appear in similar positions with similar values for both models, as well as the observation that two models have only very small differences in efficiency scores. The DMUs with efficiency score 1 are all Fully efficient.

Projection of DMUs on efficiency frontier

The projections of the DMUs onto the efficiency frontier reveals the position at the efficiency frontier a DMU can reach by adjusting its inputs and outputs as proposed by the projection. For each input and output, the Δ indicates the value by which the input or output should be improven in order to reach efficiency. The percent Δ indicate by which percent the current value needs to change. For DMUs that are already efficient, such as DMU 1, the projection corresponds to the current values of each of the inputs and outputs. However, for DMUs that are not technically efficient, such as DMU 2, a change needs to take place in order to bring the DMU to the efficiency frontier. For managers this offers a powerful tool for efficiency improvements. Say the firm corresponding to DMU 3 wishes to undertake efficiency improvements. In order to arrive at the efficient frontier the firm must reduce input 1 by 0.1182, a change of 17.97 percent from its current value. It must also undertake similar changes to input 2, and output 2.

CCR - IO Efficiency Frontier Projections

DMU	efficiency score	P(x1)	Δx1	% Δx1	P(x2)	Δx2	% Δx2	P(y1)	Δy1	%Δy1	P(y2)	Δy2	%Δy2
1	1	0,4917	0	0	233,3411	0	0	4,6188	0	0	0,1494	0	0
2	0,6452	0,3546	-0,1953	-35,52	115,9135	-63,8531	-35,52	1,5434	0	0	0,3553	0	0
3	0,8203	0,5397	-0,1182	-17,97	210,8036	-46,1799	-17,97	4,6188	0	0	0,9984	0,7277	268,7918
4	1	0,5738	0	0	193,6117	0	0	4,6188	0	0	0,3567	0	0
5	0,5041	0,3024	-0,2977	-49,61	100,7822	-99,2222	-49,61	1,7495	0	0	0,2593	0	0
6	0,3868	0,2666	-0,4244	-61,42	64,1145	-102,0714	-61,42	1,5929	0	0	0,194	0	0
7	0,8671	0,4591	-0,0704	-13,29	223,0585	-34,1881	-13,29	2,6665	0	0	37,4235	37,0166	9097,657
8	0,6592	0,3518	-0,1819	-34,08	193,974	-100,2827	-34,08	2,1019	0	0	50,5009	50,1952	16422,166
9	0,6075	0,4214	-0,2733	-39,34	123,6165	-80,1694	-39,34	1,9641	0	0	0,3918	0	0
10	0,4922	0,3706	-0,3851	-50,96	75,4681	-78,4228	-50,96	1,8006	0	0	0,2974	0	0
11	0,5616	0,32	-0,2498	-43,84	120,4718	-94,0436	-43,84	1,987	0	0	0,2671	0	0
12	0,4795	0,35	-0,3816	-52,16	89,181	-97,2342	-52,16	1,8913	0	0	0,2818	0	0
13	0,5592	0,366	-0,2886	-44,09	121,9295	-96,1523	-44,09	2,373	0	0	0,2865	0	0
14	0,6022	0,3161	-0,2088	-39,78	108,5944	-71,7351	-39,78	2,0663	0	0	0,2496	0	0
15	1	0,5991	0	0	178,5676	0	0	4,6188	0	0	0,0206	0	0
16	0,6136	0,3455	-0,2184	-38,73	98,9077	-62,5216	-38,73	1,4958	0	0	0,3305	0	0
17	0,3227	0,2574	-0,548	-68,04	48,6931	-103,8235	-68,04	1,5584	0	0	0	0	
18	0,6892	0,3808	-0,1717	-31,08	151,1066	-68,1427	-31,08	2,0875	0	0	0,3504	0	0
19	0,477	0,3514	-0,3876	-52,45	81,485	-89,882	-52,45	1,7106	0	0	0,2935	0	0
20	0,4676	0,2767	-0,3155	-53,28	85,6845	-97,7156	-53,28	1,4696	0	0	0,2434	0	0
21	1	0,4941	0	0	0,1	0	0	1,5654	0	0	0,3664	0	0
22	0,9818	0,6153	-0,0301	-4,66	221,4732	-4,1697	-1,8481	2,1821	0	0	0,7111	0	0
23	0,5102	0,349	-0,336	-49,05	102,6216	-98,7947	-49,05	1,844	0	0	0,3016	0	0
24	0,6662	0,4052	-0,2038	-33,47	119,3713	-60,0535	-33,47	1,9177	0	0	0,3742	0	0
25	0,6472	0,3395	-0,1851	-35,28	133,6805	-72,8716	-35,28	1,7188	0	0	1,6304	1,3029	397,8477
26	0,6721	0,3543	-0,1728	-32,79	139,1694	-67,8971	-32,79	2,0787	0	0	0,3105	0	0
27	0,8822	0,3982	-0,0532	-11,78	200,0597	-26,7139	-11,78	1,7867	0	0	49,9103	49,5021	12125,917
28	1	0,4939	0	0	214,7026	0	0	3,5766	0	0	0,3634	0	0
29	0,8002	0,4421	-0,1105	-20	148,07	-37,0175	-20	2,3537	0	0	0,4016	0	0
30	0,7437	0,39	-0,1344	-25,63	137,8047	-47,4914	-25,63	1,6825	0	0	0,4046	0	0
31	0,7471	0,3083	-0,378	-55,0739	182,2411	-61,8946	-25,3738	1,579	0	0	0,5788	0	0
32	0,4431	0,3504	-0,4434	-55,86	72,0414	-91,1698	-55,86	1,6742	0	0	0,285	0	0
33	0,4587	0,315	-0,3734	-54,24	78,0012	-92,456	-54,24	1,592	0	0	0,2626	0	0
34	0,439	0,3569	-0,4587	-56,24	78,054	-100,3143	-56,24	1,8237	0	0	0,2832	0	0
35	0,457	0,3938	-0,4698	-54,4	103,6842	-123,6935	-54,4	2,0709	0	0	0,3271	0	0
36	1	0,5422	0	0	192,1987	0	0	4,6188	0	0	0,2233	0	0
37	0,4622	0,2316	-0,2696	-53,79	79,4514	-92,4842	-53,79	1,3557	0	0	0,1996	0	0
38	0,6312	0,4027	-0,2401	-37,35	37,1992	-22,177	-37,35	1,6867	0	0	0,2989	0	0
39	0,4188	0,3656	-0,5103	-58,26	80,7017	-112,6421	-58,26	1,935	0	0	0,284	0	0
40	0,8997	0,4143	-0,0462	-10,03	198,5897	-22,1391	-10,03	3,0724	0	0	27,5628	27,2683	9258,382
41	0,5702	0,4172	-0,3159	-43,09	113,9017	-86,2419	-43,09	1,8459	0	0	0,3883	0	0
42	0,6803	0,4456	-0,2105	-32,08	126,4785	-59,7384	-32,08	1,9647	0	0	0,4212	0	0
43	0,5724	0,3511	-0,263	-42,83	106,5947	-79,8575	-42,83	1,7897	0	0	0,3144	0	0
44	1	0,4927	0	0	176,0903	0	0	1,7148	0	0	0,5573	0	0
45	0,4854	0,3047	-0,325	-51,61	67,602	-72,1005	-51,61	1,7901	0	0	0,2182	0	0
46	0,4219	0,3199	-0,4394	-57,87	93,0683	-127,8392	-57,87	1,7704	0	0	0,2668	0	0
47	0,6147	0,4124	-0,2599	-38,66	111,1327	-70,0422	-38,66	1,5982	0	0	0,4062	0	0
48	0,6166	0,4413	-0,2763	-38,5	109,7268	-68,6908	-38,5	1,8698	0	0	0,4068	0	0
49	0,8162	0,4728	-0,1068	-18,42	154,9436	-34,9848	-18,42	2,4105	0	0	0,4368	0	0
50	0,6341	0,238	-0,1373	-36,59	116,7434	-67,3654	-36,59	1,457	0	0	18,9674	18,7643	9239,109
Min:	0,3227	0,2316	-0,548	-68,04	0,1	-127,8392	-68,04	1,3557	0	0	0	0	0
Max:	1	0,6153	0	0	233,3411	0	0	4,6188	0	0	50,5009	50,1952	16422,166
Average:	0,6611	0,3942	-0,2312	-34,6246	126,4910	-63,6554	-33,9611	2,1919	0,0000	0,0000	3,7091	3,3880	991,0576
Standard Deviation:	0,2003	0,0879	0,1546	20,1681	54,2770	37,3806	20,0761	0,9156	0,0000	0,0000	11,6692	11,6580	3384,242
1st quartile	0,4854	0,3455	-0,336	-52,16	85,6845	-94,0436	-51,61	1,6867	0	0	0,2671	0	0
2nd quartile	0,6166	0,3706	-0,2401	-38,66	115,9135	-70,0422	-38,5	1,8459	0	0	0,3271	0	0
3rd quartile	0,8162	0,4421	-0,1105	-18,42	176,0903	-34,9848	-18,42	2,1019	0	0	0,4062	0	0
4th quartile	1	0,6153	0	0	233,3411	0	0	4,6188	0	0	50,5009	50,1952	16422,166
1st decile	0,4423	0,3042	-0,4402	-55,936	71,1535	-100,666	-55,936	1,5640	0	0	0,21448	0	0
2nd decile	0,4732	0,3200	-0,37944	-53,484	80,2016	-97,427	-52,782	1,6438	0	0	0,26128	0	0
3rd decile	0,5065	0,3502	-0,31574	-50,42	95,4041	-91,942	-49,386	1,7164	0	0	0,28352	0	0
4th decile	0,5784	0,3544	-0,27256	-43,038	106,9946	-79,571	-42,22	1,7898	0	0	0,2977	0	0
5th decile	0,6166	0,3706	-0,2401	-38,66	115,9135	-70,042	-38,5	1,8459	0	0	0,3271	0	0
6th decile	0,6648	0,4018	-0,197	-34,32	125,9061	-62,788	-33,592	1,9583	0	0	0,36206	0	0
7th decile	0,7457	0,4197	-0,14932	-27,81	149,8920	-46,705	-25,47628	2,0756	0	0	0,39768	0	0
8th decile	0,8731	0,4646	-0,06352	-12,686	186,2241	-24,899	-12,686	2,3614	0	0	0,42744	0	0
9th decile	1	0,5032	0	0	202,2085	0	0	3,7850	0	0	1,1248	0,84274	307,5085
10th decile	1	0,6153	0	0	233,3411	0	0	4,6188	0	0	50,5009	50,1952	16422,166

Figure 11: CCR-IO Envelopment Form Efficiency Frontier Projection

CCR - OO Efficiency Frontier Projections														
DMU	efficiency score	P(x1)	Δx1	% Δx1	P(x2)	Δx2	% Δx2	P(y1)	Δy1	%Δy1	P(y2)	Δy2	%Δy2
1	1	0,4917	0	0	233,3411	0	0	4,6188	0	0	0,1494	0	0
2	1,5508	0,5499	0	0	179,7666	0	0	2,3936	0,8501	55,08	0,551	0,1957	55,08
3	1,219	0,658	0	0	256,9836	0	0	5,6303	1,0115	21,9	1,217	0,0593	21,9
4	1	0,5738	0	0	193,6117	0	0	4,6188	0	0	0,3567	0	0
5	1,9845	0,6002	0	0	200,0043	0	0	3,4719	1,7224	98,45	0,5146	0,2553	98,45
6	2,5918	0,691	0	0	166,1859	0	0	4,1285	2,5356	159,18	0,5028	0,3088	159,18
7	1,1533	0,5295	0	0	257,2466	0	0	3,0753	0,4088	15,33	43,1604	0,0624	15,33
8	1,5171	0,5337	0	0	294,2567	0	0	3,1889	1,0869	51,71	76,6138	0,1581	51,71
9	1,6485	0,6946	0	0	203,7859	0	0	3,2377	1,2737	64,85	0,6458	0,2541	64,85
10	2,0393	0,7557	0	0	153,8909	0	0	3,6719	1,8713	103,93	0,6064	0,3091	103,93
11	1,7806	0,5699	0	0	214,5154	0	0	3,5381	1,5511	78,06	0,4757	0,2085	78,06
12	2,0902	0,7316	0	0	186,4152	0	0	3,9532	2,0619	109,02	0,589	0,3072	109,02
13	1,7887	0,6545	0	0	218,0818	0	0	4,2446	1,8716	78,87	0,5125	0,226	78,87
14	1,6607	0,525	0	0	180,3295	0	0	3,4315	1,3652	66,07	0,4146	0,1649	66,07
15	1	0,5991	0	0	178,5676	0	0	4,6188	0	0	0,0206	0	0
16	1,6322	0,5639	0	0	161,4293	0	0	2,4414	0,9456	63,22	0,5394	0,2089	63,22
17	3,1286	0,8054	0	0	152,3565	-0,2353	-0,1542	4,8756	3,3172	212,86	0	0	
18	1,4509	0,5525	0	0	219,2492	0	0	3,0288	0,9413	45,09	0,5084	0,158	45,09
19	2,1028	0,739	0	0	171,367	0	0	3,5971	1,8865	110,28	0,6171	0,3236	110,28
20	2,1405	0,5922	0	0	183,4001	0	0	3,1456	1,676	114,05	0,521	0,2776	114,05
21	1	0,4941	0	0	0,1	0	0	1,5654	0	0	0,3664	0	0
22	1,0188	0,6268	-0,0186	-2,8819	225,6247	0	0	2,2231	0,041	1,88	0,7245	0,0134	1,88
23	1,9627	0,685	0	0	201,4163	0	0	3,6193	1,7752	96,27	0,592	0,2904	96,27
24	1,5032	0,609	0	0	179,4247	0	0	2,8826	0,965	50,32	0,5624	0,1883	50,32
25	1,5452	0,5246	0	0	206,5521	0	0	2,6559	0,9371	54,52	2,5192	0,1785	54,52
26	1,4878	0,5271	0	0	207,0665	0	0	3,0928	1,014	48,78	0,462	0,1515	48,78
27	1,1336	0,4514	0	0	226,7736	0	0	2,0254	0,2387	13,36	56,577	0,0545	13,36
28	1	0,4939	0	0	214,7026	0	0	3,5766	0	0	0,3634	0	0
29	1,25	0,5526	0	0	185,0875	0	0	2,9421	0,5884	25	0,502	0,1004	25
30	1,3447	0,5244	0	0	185,296	0	0	2,2624	0,58	34,47	0,5441	0,1395	34,47
31	1,3385	0,4128	-0,2735	-39,8528	243,9313	0	0	2,1136	0,5345	33,85	0,7747	0,1959	33,85
32	2,2655	0,7938	0	0	163,2112	0	0	3,793	2,1188	126,55	0,6457	0,3607	126,55
33	2,1854	0,6884	0	0	170,4573	0	0	3,4791	1,8871	118,54	0,574	0,3113	118,54
34	2,2854	0,8156	0	0	178,3683	0	0	4,168	2,3442	128,54	0,6473	0,3641	128,54
35	2,1928	0,8636	0	0	227,3777	0	0	4,5411	2,4702	119,28	0,7173	0,3902	119,28
36	1	0,5422	0	0	192,1987	0	0	4,6188	0	0	0,2233	0	0
37	2,1638	0,5011	0	0	171,9356	0	0	2,9335	1,5778	116,38	0,4318	0,2323	116,38
38	1,5962	0,6428	0	0	59,3762	0	0	2,6924	1,0056	59,62	0,4771	0,1782	59,62
39	2,3957	0,8759	0	0	193,3439	0	0	4,6357	2,7007	139,57	0,6803	0,3963	139,57
40	1,1115	0,4604	0	0	220,7288	0	0	3,4149	0,3426	11,15	30,6353	0,0328	11,15
41	1,7572	0,7331	0	0	200,1435	0	0	3,2435	1,3977	75,72	0,6823	0,294	75,72
42	1,4723	0,6561	0	0	186,2168	0	0	2,8926	0,9279	47,23	0,6201	0,1989	47,23
43	1,7491	0,6141	0	0	186,4522	0	0	3,1304	1,3407	74,91	0,55	0,2355	74,91
44	1	0,4927	0	0	176,0903	0	0	1,7148	0	0	0,5573	0	0
45	2,0666	0,6297	0	0	139,7025	0	0	3,6995	1,9094	106,66	0,4509	0,2327	106,66
46	2,3735	0,7593	0	0	220,9075	0	0	4,202	2,4316	137,35	0,6333	0,3665	137,35
47	1,6302	0,6724	0	0	181,175	0	0	2,6055	1,0072	63,02	0,6622	0,256	63,02
48	1,6261	0,7176	0	0	178,4175	0	0	3,0405	1,1707	62,61	0,6615	0,2547	62,61
49	1,2258	0,5795	0	0	189,9284	0	0	2,9548	0,5443	22,58	0,5354	0,0986	22,58
50	1,577	0,3753	0	0	184,1089	0	0	2,2977	0,8407	57,7	29,9125	0,1172	57,7
Min:	1	0,4128	-0,2735	-39,8528	0,1	-0,2353	-0,1542	1,5654	0	0	0	0	0
Max:	3,1286	0,8759	0	0	294,2567	0	0	5,6303	3,3172	212,86	76,6138	0,3963	159,18
Average:	1,6563	0,6194	-0,0060	-0,8721	190,1386	-0,0048	-0,0031	3,3802	1,1883	65,6349	4,7528	0,1835	62,5677
Standard Deviation:	0,5010	0,1118	0,0391	5,6996	44,9996	0,0336	0,0220	0,8786	0,8419	50,0989	14,9900	0,1224	45,7440
1st quartile	1,2258	0,5295	0	0	178,3683	0	0	2,8926	0,5443	22,58	0,4771	0,0624	22,41
2nd quartile	1,6261	0,6002	0	0	186,4522	0	0	3,2435	1,014	62,61	0,5573	0,1959	61,115
3rd quartile	2,0666	0,691	0	0	214,7026	0	0	3,9532	1,8713	106,66	0,6615	0,2776	104,6125
4th quartile	3,1286	0,8759	0	0	294,2567	0	0	5,6303	3,3172	212,86	76,6138	0,3963	159,18
1st decile	1,0000	0,4937	0	0	159,9216	0,000	0	2,2545	0	0	0,36206	0	0
2nd decile	1,1454	0,5248	0	0	171,7082	0,000	0	2,6778	0,38232	14,542	0,45756	0,04582	14,148
3rd decile	1,3410	0,5453	0	0	178,9104	0,000	0	2,9472	0,69308	34,098	0,50504	0,11604	33,912
4th decile	1,5060	0,5707	0	0	183,7376	0,000	0	3,1003	0,94948	50,598	0,5362	0,16756	50,012
5th decile	1,6261	0,6002	0	0	186,4522	0,000	0	3,2435	1,014	62,61	0,5573	0,1959	61,115
6th decile	1,7314	0,6402	0	0	198,7258	0,000	0	3,5263	1,3603	73,142	0,60352	0,23104	67,838
7th decile	1,9758	0,6800	0	0	206,8607	0,000	0	3,6885	1,70384	97,578	0,64576	0,25506	94,53
8th decile	2,1179	0,7232	0	0	219,8410	0,000	0	4,1816	1,88674	111,788	0,6811	0,29928	109,776
9th decile	2,26948	0,7662	0	0	228,5704	0	0	4,6188	2,36168	126,948	1,47744	0,33102	121,461
10th decile	3,1286	0,8759	0	0	294,2567	0	0	5,6303	3,3172	212,86	76,6138	0,3963	159,18

Figure 12: CCR-OO Envelopment Form Efficiency Frontier Projection

0.2 Task 2

0.2.1 Matrix Representations of CCR- IO and CCR-OO Multiplier Formulations

The CCR Multiplier formulations are the Dual formulations of the CCR model proposed by Charnes, Cooper, and Rhodes (1978).

CCR - IO Matrix Format

$$\begin{aligned} &\text{Maximize z} = && \begin{bmatrix} Y_{1\times s} & 0_{1\times m} & \varepsilon_{1\times(n)} \end{bmatrix} \begin{bmatrix} u_{s\times 1} \\ v_{m\times 1} \\ s^{+}_{n\times 1} \end{bmatrix} \\ &\text{s.t.} && \begin{bmatrix} 0_{1\times s} & X_{1\times m} & 0_{1\times n} \\ Y_{n\times s} & -X_{n\times m} & In \times n \end{bmatrix} \begin{bmatrix} u_{s\times 1} \\ v_{m\times 1} \\ s^{+}_{n\times 1} \end{bmatrix} = \begin{bmatrix} 1_{1\times 1} \\ 0_{n\times 1} \end{bmatrix} \end{aligned} \tag{3}$$

Comments on the structure of the constraints matrix

$0_{1\times s}$ is a vector of zeros of these dimensions, while X is a vector of input values of $1 \times m$ dimensions. Taken together with the vector of zeros $0_{1\times n}$ these form the first degree constraints. Y is a matrix of input values of $n \times s$ dimensions while -X is a matrix of negative input values of $n \times m$ dimensions. I is an identity vector of $n \times n$ dimensions which, when the contraints matrix is multiplied with the decision vector, assigns on slack value per row.

CCR - OO Matrix Format

$$\begin{aligned} &\text{Minimize z} = && \begin{bmatrix} 0_{1\times s} & X_{1\times m} & -\varepsilon_{1\times(n)} \end{bmatrix} \begin{bmatrix} u_{s\times 1} \\ v_{m\times 1} \\ s^{+}_{n\times 1} \end{bmatrix} \\ &\text{s.t.} && \begin{bmatrix} Y_{1\times s} & 0_{1\times m} & 0_{1\times n} \\ Y_{n\times s} & -X_{n\times m} & In \times n \end{bmatrix} \begin{bmatrix} u_{s\times 1} \\ v_{m\times 1} \\ s^{+}_{n\times 1} \end{bmatrix} = \begin{bmatrix} 1_{1\times 1} \\ 0_{n\times 1} \end{bmatrix} \end{aligned} \tag{4}$$

Comments on the structure of objective function

For the objective function, infinitely small values of ϵ are preferred as these return a more accurate solution, allowing for less error.

Comments on the structure of the constraints matrix

Y is denotes a vector of input values of $1 \times s$ dimensions, while 0 $1 \times m$ denotes a vector of zeros of these dimensions. Taken together with the vector of zeros 0 $1 \times n$ these form the first degree constraints. Y is a matrix of input values of $n \times s$ dimensions while -X is a matrix of negative input values of $n \times m$ dimensions. I is an identity vector of $n \times n$ dimensions which, when the contraints matrix is multiplied with the decision vector, assigns on slack value per row.

0.2.2 Pseudo Code

Algorithm 3 Analysis 3 - CCR Input Oriented in Dual Form (dual_CCR_IO.m)

```
procedure ANALYSIS 3 - CCR INPUT ORIENTED IN DUAL FORM (DUAL_CCR_IO.M)
    Declare an empty matrix Z that will hold the results

    Declare the base column title that will be used for the output files in a cell array variables declared as "column_headers"

    Declare a vector of 0s that serves as a lower bound for the solution vector

    For each DMU
Declare the objective function "f"
Find its Aeq
Find its beq
Solve the optimization problem for the objective function
Append the achieved result to the solution matrix Z

    Loop from 1 to s using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form U_i" which stands for slack values

    Loop from 1 to m using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "V_i" which stands for slack values

    Loop from 1 to n using a variable "i" and horizontally append a string entry to the cell array variable "column_headers_2" in the form "S+i"

    Append needed elements "slacks sum" and "efficiency score" to the "column_headers" vector variable

    Save the current file name as a string into the variable "filename"

    Execute the custom function "export" passing the variable "Z"
```

Algorithm 4 Analysis 4 - CCR Output Oriented in Dual Form (dual_CCR_IO.m)

```
procedure ANALYSIS 4 - CCR OUTPUT ORIENTED IN DUAL FORM (DUAL_CCR_IO.M)

    Declare an empty matrix Z that will hold the results

    Declare the base column title that will be used for the output files in
    a cell array variables declared as "column_headers"

    Declare a vector of 0s that serves as a lower bound for the solution
    vector

    For each DMU
Declare the objective function "f"
Find its Aeq
Find its beq
Solve the optimization problem for the objective function
Append the achieved result to the solution matrix Z

    Loop from 1 to s using a variable "i" and horizontally append a string
    entry to the cell array variable "column_headers" in the form U_i" which
    stands for slack values

    Loop from 1 to m using a variable "i" and horizontally append a string
    entry to the cell array variable "column_headers" in the form "V_i" which
    stands for slack values

    Loop from 1 to n using a variable "i" and horizontally append a string
    entry to the cell array variable "column_headers_2" in the form "S+i"

    Append needed elements "slacks sum" and "efficiency score" to the
    "column_headers" vector variable

    Save the current file name as a string into the variable "filename"

    Execute the custom function "export" passing the variable "Z"
```

0.2.3 Matlab Implementation

For DMU 21 we changed entry from 0 to 5 in order to alleviate the difference between the DMUs and make the problem solvable.

Please find the MATLAB code used in Project 1 Task 1 in the appropriate folder on the memory stick, files titled "RUN.m", "dual_CCR_IO.m", and "dual_CCR_OO.m".

In addition, open functions "calculate_benchmarks.m", "calculate_frontier_io.m", "calculate_frontier_barnum_io.m", "calculate_frontier_oo.m" , "classify.m", "export.m", "optimize.m" for all analyses in project 1.

Opening all matlab files and executing "RUN.m" returns analysis output excel results for all tasks.

0.2.4 Analysis

Multiplier and Weights

Weights provided by the multiplier formulation are derived from the data and are the optimal weights for the chosen maximisation/ minimization context. Weightings taking zero values also reveal inefficiencies in the DMU, as well as relative importance of different inputs and outputs at specific DMUs (Cooper et. al, 2007).

As can be observed by the table (Figure 13) , the average weights assigned differ between the input and output oriented models. On average, input 2 and output 1 are assigned more weight than input 1 and output 2 in both models.

The decile analysis reveals the weight of 10 assigned to output 2 (V 2) as an outlier in both the input and output oriented case.

	CCR- IO Multiplier form Weights				CCR- OO Multiplier form Weights			
DMU	U_1	U_2	V_1	V_2	U_1	U_2	V_1	V_2
1	0,2165	0	2,0337	0	0,2165	0	2,0337	0
2	0,132	1,2416	1,3385	0,0015	0,2046	1,9254	2,0758	0,0023
3	0,1423	0,6017	1,5198	0	0,1735	0,7335	1,8526	0
4	0,1904	0,3381	1,3776	0,0011	0,1904	0,3381	1,3776	0,0011
5	0,1203	1,1317	1,2201	0,0013	0,2387	2,246	2,4214	0,0027
6	0,1129	1,062	1,145	0,0013	0,2926	2,7526	2,9676	0,0033
7	0,1326	1,262	1,8887	0	0,1529	1,4555	2,1782	0
8	0,1315	1,2519	1,8736	0	0,1996	1,8993	2,8424	0
9	0,1074	1,0102	1,0891	0,0012	0,177	1,6652	1,7953	0,002
10	0,1066	1,0033	1,0817	0,0012	0,2175	2,046	2,2058	0,0024
11	0,1274	1,1547	1,6436	0,0003	0,2268	2,056	2,9265	0,0005
12	0,1053	0,9909	1,0683	0,0012	0,2201	2,0712	2,233	0,0024
13	0,1103	1,0378	1,1188	0,0012	0,1973	1,8563	2,0012	0,0022
14	0,1364	1,2832	1,3835	0,0015	0,2265	2,131	2,2975	0,0025
15	0,2165	0	0,6852	0,0033	0,2165	0	0,6852	0,0033
16	0,133	1,2517	1,3494	0,0015	0,2171	2,043	2,2025	0,0024
17	0,2051	0	0,6491	0,0031	0,6417	0	2,0309	0,0098
18	0,131	1,1869	1,6895	0,0003	0,19	1,722	2,4512	0,0004
19	0,1063	1,0006	1,0787	0,0012	0,2236	2,1041	2,2684	0,0025
20	0,1243	1,1691	1,2604	0,0014	0,266	2,5026	2,698	0,003
21	0	2,729	0	10	0	2,729	0	10
22	0	1,3804	1,0234	0,0015	0	1,4063	1,0426	0,0015
23	0,1088	1,0238	1,1038	0,0012	0,2136	2,0095	2,1665	0,0024
24	0,1223	1,151	1,2409	0,0014	0,1839	1,7301	1,8653	0,002
25	0,1338	1,2737	1,9063	0	0,2068	1,9682	2,9455	0
26	0,1374	1,2449	1,772	0,0003	0,2044	1,8522	2,6364	0,0005
27	0,1555	1,4802	2,2153	0	0,1763	1,678	2,5112	0
28	0,1626	1,1518	2,0246	0	0,1626	1,1518	2,0246	0
29	0,1305	1,2275	1,3233	0,0015	0,1631	1,5344	1,6542	0,0018
30	0,1355	1,2746	1,3742	0,0015	0,1822	1,714	1,8479	0,002
31	0	1,2908	0,957	0,0014	0	1,7278	1,281	0,0019
32	0,1013	0,9534	1,0279	0,0011	0,2296	2,1599	2,3286	0,0026
33	0,1126	1,0596	1,1424	0,0013	0,2461	2,3157	2,4966	0,0027
34	0,0975	0,9172	0,9888	0,0011	0,2228	2,0961	2,2598	0,0025
35	0,0886	0,8334	0,8985	0,001	0,1942	1,8274	1,9701	0,0022
36	0,2165	0	1,2855	0,0016	0,2165	0	1,2855	0,0016
37	0,1429	1,3447	1,4498	0,0016	0,3093	2,9098	3,137	0,0034
38	0,1393	1,3102	1,4125	0,0015	0,2223	2,0913	2,2547	0,0025
39	0,0906	0,8525	0,9191	0,001	0,2171	2,0424	2,2019	0,0024
40	0,1744	1,2355	2,1719	0	0,1938	1,3733	2,4139	0
41	0,1035	0,9736	1,0497	0,0012	0,1818	1,7108	1,8445	0,002
42	0,1146	1,0781	1,1623	0,0013	0,1687	1,5873	1,7113	0,0019
43	0,1204	1,133	1,2214	0,0013	0,2106	1,9816	2,1364	0,0023
44	0	1,7945	2,0294	0	0	1,7945	2,0294	0
45	0,1259	1,1847	1,2773	0,0014	0,2602	2,4484	2,6396	0,0029
46	0,0984	0,926	0,9983	0,0011	0,2336	2,198	2,3696	0,0026
47	0,1132	1,0647	1,1479	0,0013	0,1845	1,7358	1,8714	0,0021
48	0,1079	1,0156	1,0949	0,0012	0,1755	1,6514	1,7804	0,002
49	0,1251	1,1772	1,2692	0,0014	0,1534	1,4431	1,5558	0,0017
50	0,1871	1,7802	2,6643	0	0,295	2,8075	4,2017	0
Min:	0	0	0	0	0	0	0	0
Max:	0,2165	2,729	2,2153	10	0,6417	2,9098	3,137	10
Average:	0,1214	1,0624	1,3058	0,2051	0,2000	1,7228	2,0777	0,2060
Standard Deviation	0,0483	0,4489	0,4225	1,4284	0,0920	0,6869	0,5812	1,4283
1st quartile	0,1066	0,9909	1,0787	0,001	0,177	1,5873	1,8479	0,0011
2nd quartile	0,1243	1,133	1,2409	0,0012	0,2046	1,8522	2,1665	0,0021
3rd quartile	0,1364	1,2517	1,4498	0,0014	0,2228	2,0913	2,4139	0,0025
4th quartile	0,2165	2,729	2,2153	10	0,6417	2,9098	3,137	10
1st decile	0,0902	0,5490	0,9494	0,0000	0,1533	0,6544	1,3592	0,0000
2nd decile	0,1046	0,9424	1,0410	0,0003	0,1747	1,4505	1,7893	0,0005
3rd decile	0,1083	1,0061	1,0914	0,0011	0,1829	1,6703	1,8677	0,0016
4th decile	0,1135	1,0601	1,1456	0,0012	0,1939	1,7283	2,0297	0,0020
5th decile	0,1243	1,1330	1,2409	0,0012	0,2046	1,8522	2,1665	0,0021
6th decile	0,1309	1,1756	1,3157	0,0013	0,2165	1,9789	2,2276	0,0024
7th decile	0,1335	1,2392	1,3811	0,0014	0,2191	2,0520	2,3162	0,0025
8th decile	0,1405	1,2667	1,6620	0,0015	0,2266	2,1149	2,4694	0,0026
9th decile	0,1776	1,3171	1,92996	0,00152	0,24892	2,34224	2,72688	0,00306
10th decile	0,2165	2,729	2,2153	10	0,6417	2,9098	3,137	10

Figure 13: CCR-IO and OO Multiplier Form Weights and Statistics

CCR IO and OO Envelopment Form Correlation Analysis					
s+_1	s+_2	s-_1	s-_2	slacks sum	efficiency score
#DIV/0!	0,988207369	0,99978021	1	0,988201497	-0,951819415

Figure 14: CCR Envelopment form IO and OO Correlation Analysis

Multiplier and Envelopment Forms

Solving the Multiplier form gives direct access to information on the optimal weights of inputs and outputs for each DMU (Cooper, et. al. 2007 p.52)

0.3 Task 3

The BCC model proposed by Banker, Charnes and Cooper (1984) allows for variable returns to scale(Cooper, et. al 2007 p. 89-93).

0.3.1 Matrix Representation of the BCC-IO and BCC-OO models in envelopment form

BCC - IO Matrix Format

$$\text{Minimize z} = \begin{bmatrix} 0_{1\times n} & -\varepsilon_{1\times(m+s)} & 1_{1\times 1} \end{bmatrix} \begin{bmatrix} \lambda_{n\times 1} \\ s^-_{m\times 1} \\ s^+_{s\times 1} \\ \theta_k \end{bmatrix}$$

$$\text{s.t.} \quad \begin{bmatrix} X_{m\times n} & I_{m\times m} & 0_{m\times s} & -X_{m\times 1} \\ Y_{s\times n} & 0_{s\times m} & -Is\times s & 0_{s\times 1} \\ 1_{1\times n} & 0_{1\times(m+s+1)} & & \end{bmatrix} \begin{bmatrix} \lambda_{n\times 1} \\ s^-_{m\times 1} \\ s^+_{s\times 1} \\ \theta_k \end{bmatrix} = \begin{bmatrix} 0_{m\times 1} \\ J_{s\times 1} \end{bmatrix} \tag{5}$$

Comment on structures of the constraints matrix

The matrix structure is identical to CCR - IO, except for the addition of the final row vector. In this row, $1_{1\times n}$ denotes a vector of ones of n rows. This element is responsible for the model change which adjusts for variable returns to scale. $0_{1\times(m+s+1)}$ denotes a vector of zeros of these dimensions.

BCC - OO Matrix Format

$$\text{Maximize z} = \begin{bmatrix} 0_{1\times n} & \varepsilon_{1\times(m+s)} & 1_{1\times 1} \end{bmatrix} \begin{bmatrix} \lambda_{n\times 1} \\ s^-_{m\times 1} \\ s^+_{s\times 1} \\ \phi_k \end{bmatrix}$$

$$\text{s.t.} \quad \begin{bmatrix} X_{m\times n} & I_{m\times m} & 0_{m\times(s+1)} & \\ Y_{s\times n} & 0_{s\times m} & -Is \times s & -Y_{s\times 1} \\ 1_{1\times n} & 0_{1\times(m+s+1)} & & \end{bmatrix} \begin{bmatrix} \lambda_{n\times 1} \\ s^-_{m\times 1} \\ s^+_{s\times 1} \\ \phi_k \end{bmatrix} = \begin{bmatrix} X_{m\times 1} \\ 0_{s\times 1} \end{bmatrix} \tag{6}$$

Comment on structures of the constraints matrix

The matrix structure is identical to CCR - OO, except for the addition of the final row vector. In this row, $1_{1\times n}$ denotes a vector of ones of n rows. This element is responsible for the model change which adjusts for variable returns to scale. $0_{1\times(m+s+1)}$ denotes a vector of zeros of these dimensions.

0.3.2 Pseudo Code

Algorithm 5 Analysis 5 - BCC Input Oriented in Envelopment Form (env_BCC_IO.m)

procedure ANALYSIS 5 - BCC INPUT ORIENTED IN ENVELOPMENT FORM (ENV_BCC_IO.M)

Declare an empty matrix Z that will hold the results

Declare the base column titles that will be used for the output files in a cell array variables declared as "column_headers" and "column_headers_2"

Declare the objective function "f"

Declare a vector of 0s that serves as a lower bound for the solution vector

For each DMU

Find its Aeq

Find its beq

Solve the optimization problem for the objective function

Append the achieved result to the solution matrix Z

Use the custom function "calculate_benchmarks" to save tuples of the type "benchmark group member 1 (benchmark value), benchmark group member 2 (benchmark value) . . . " into a matrix declared as "benchmarks"

Use the custom function "calculate_frontier_io" to save frontier related information in a matrix variable declared as "eff_frontier"

Use the custom function "classify" to save the classification of each DMU into a vector matrix variable declared as "classification"

Loop from 1 to n using a variable "i" and horizontally append a string entry to the cell array variable "column_headers_2" in the form "λi"

Loop from 1 to m using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "Si-i" which stands for slack values

Loop from 1 to s using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "So+i" which stands for slack values

Append needed elements "slacks sum" and "efficiency score" to the "column_headers" vector variable

Horizontally concatenate the matrix variable “benchmarks” with the λ values from the matrix Z

Assign the return value of the custom function “sum_slacks” to the variable Z

Save the current file name as a string into the variable “filename”

Execute the custom function “export” once for each of the result variables - “Z”, “benchmarks”, “eff_frontier”, and “classification”

Algorithm 6 Analysis 6 - BCC Output Oriented in Envelopment Form (env_BCC_OO.m)

procedure ANALYSIS 6 - BCC OUTPUT ORIENTED IN ENVELOPMENT FORM (ENV_BCC_OO.M)

Declare an empty matrix Z that will hold the results

Declare the base column titles that will be used for the output files in a cell array variables declared as "column_headers" and "column_headers_2"

Declare the objective function "f"

Declare a vector of 0s that serves as a lower bound for the solution vector

For each DMU

Find its Aeq

Find its beq

Solve the optimization problem for the objective function

Append the achieved result to the solution matrix Z

Use the custom function "calculate_benchmarks" to save tuples of the type "benchmark group member 1 (benchmark value), benchmark group member 2 (benchmark value) ..." into a matrix declared as "benchmarks"

Use the custom function "calculate_frontier_oo" to save frontier related information in a matrix variable declared as "eff_frontier"

Use the custom function "classify" to save the classification of each DMU into a vector matrix variable declared as "classification"

Loop from 1 to n using a variable "i" and horizontally append a string entry to the cell array variable "column_headers_2" in the form "λi"

Loop from 1 to m using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "Si-i" which stands for slack values

Loop from 1 to s using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "So+i" which stands for slack values

Append needed elements "slacks sum" and "efficiency score" to the "column_headers" vector variable

Horizontally concatenate the matrix variable “benchmarks” with the λ values from the matrix Z

Assign the return value of the custom function “sum_slacks” to the variable Z

Save the current file name as a string into the variable “filename”

Execute the custom function “export” once for each of the result variables - “Z”, “benchmarks”, “eff_frontier”, and “classification”

0.3.3 Matlab Implementation

Please find the MATLAB code used in Project 1 Task 1 in the appropriate folder on the memory stick, files titled ”RUN.m”, ”env_BCC_IO.m” and ”env_BCC_OO.m” .

In addition, open functions ”calculate_benchmarks.m”, ”calculate_frontier_io.m”, ”calculate_frontier_barnum_io.m”, ”calculate_frontier_oo.m” , ”classify.m”, ”export.m”, ”optimize.m” for all analyses in project 1. Opening all matlab files and executing ”RUN.m” returns analysis output excel results for all tasks.

0.3.4 Analysis

Score Ranking

The ranking is the relative efficiency of the DMUs (Banker, Charnes and Cooper, 1984). The BBC models adjusts for variable returns to scale, which we can see in the results ranking. Where the CCR models returned an efficiency score of 1 for 7 DMUs, the BCC returns this score for 9 DMUs. In addition to the DMUs scored 1 in the CCR, namely 1,4,15, 21,28,36 and 44, the BCC model assigns also assigns this score to DMU 22 and 50. This shows that the difference between CCR and BCC efficiency scores for these DMUs were due to returns to scale factors. As with CCR, the improvement potential of DMUs in the input oriented model is their shortfall from the benchmark value 1 (Figure 16), while the improvement potential of DMUs in the output oriented model is their values exceeding this benchmark (Figure 17).

	BCC - IO Scores		BCC- OO Scores		Comparative Analysis
DMU	efficiency score	Rank	efficiency score	Rank	Rank Difference
1	1	1	1	1	0
2	0,7986	23	1,5404	29	-6
3	0,8203	17	1	2	-12
4	1	2	1	3	0
5	0,6995	32	1,9658	46	29
6	0,6242	40	2,2357	49	-6
7	0,902	12	1,1445	13	-37
8	0,8019	22	1,5043	26	9
9	0,6732	36	1,4847	24	10
10	0,5968	44	1,6337	32	20
11	0,744	27	1,7626	39	-12
12	0,6054	42	1,7669	41	1
13	0,6878	34	1,6283	31	3
14	0,818	18	1,6344	33	-15
15	1	3	1	4	-1
16	0,776	24	1,6177	30	-6
17	0,5489	47	2,6786	50	-3
18	0,807	21	1,4339	21	0
19	0,5972	43	1,747	36	7
20	0,6955	33	2,1195	48	-15
21	1	4	1	5	-1
22	1	5	1	6	-1
23	0,6413	38	1,7724	42	-4
24	0,7571	25	1,4669	22	3
25	0,8156	19	1,5349	28	-9
26	0,8269	16	1,4801	23	-7
27	0,9952	11	1,0118	12	-1
28	1	6	1	7	-1
29	0,864	13	1,2445	16	-3
30	0,8625	14	1,3399	18	-4
31	0,7495	26	1,2286	15	11
32	0,558	46	1,7618	38	8
33	0,6242	41	1,9166	44	-3
34	0,5476	48	1,7607	37	11
35	0,5311	49	1,6369	34	15
36	1	7	1	8	-1
37	0,8073	20	2,1045	47	-27
38	0,7355	28	1,3242	17	11
39	0,5152	50	1,7663	40	10
40	0,9977	10	1,0058	11	-1
41	0,6352	39	1,5168	27	12
42	0,727	29	1,3687	19	10
43	0,7155	30	1,6993	35	-5
44	1	8	1	9	-1
45	0,7048	31	1,8754	43	-12
46	0,5649	45	1,9617	45	0
47	0,6877	35	1,4884	25	10
48	0,6623	37	1,4017	20	17
49	0,8452	15	1,2151	14	1
50	1	9	1	10	-1

Figure 15: CCR Envelopment form DEA Score Ranking

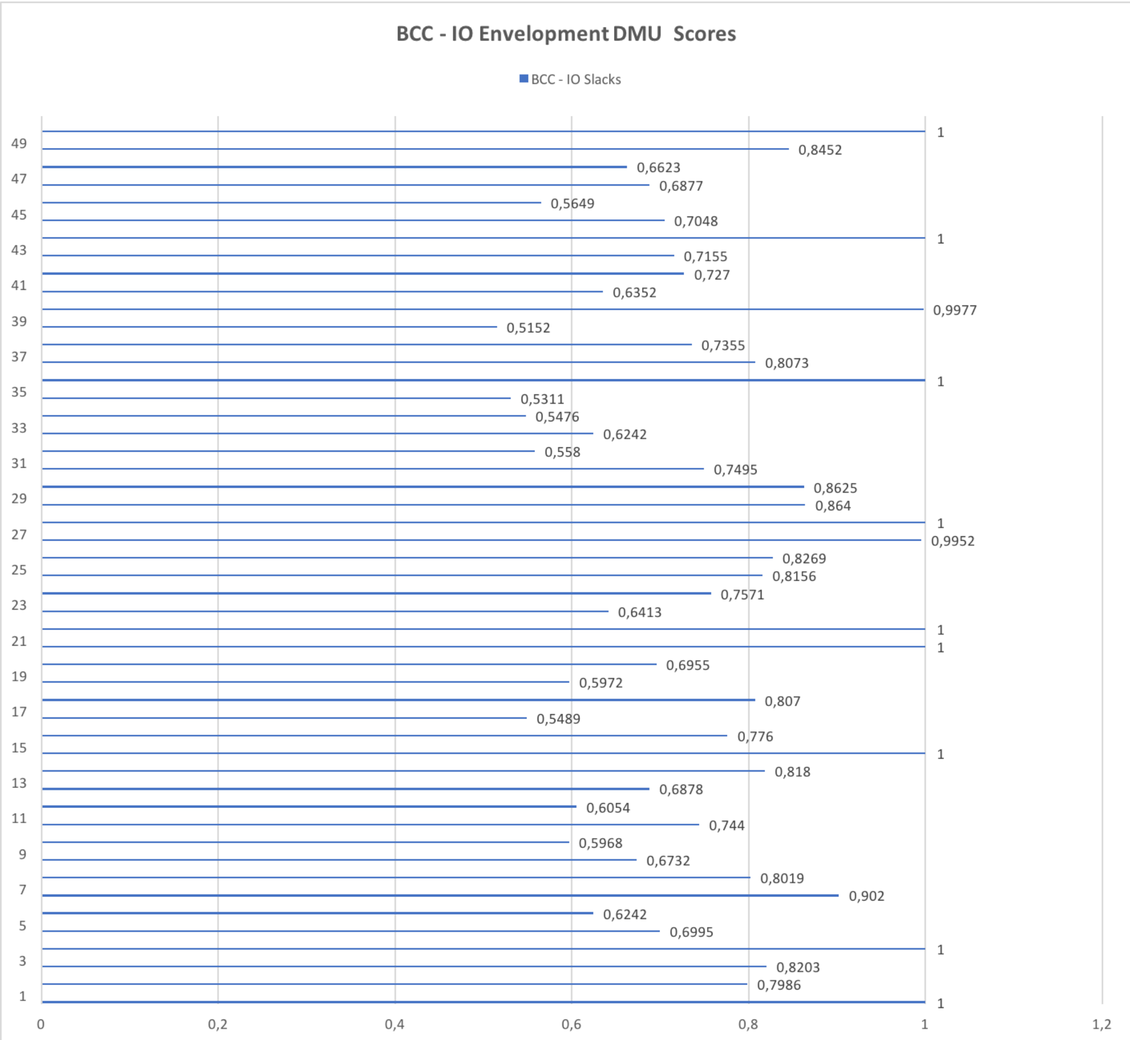

Figure 16: BCC-IO Envelopment form DEA Score Graph

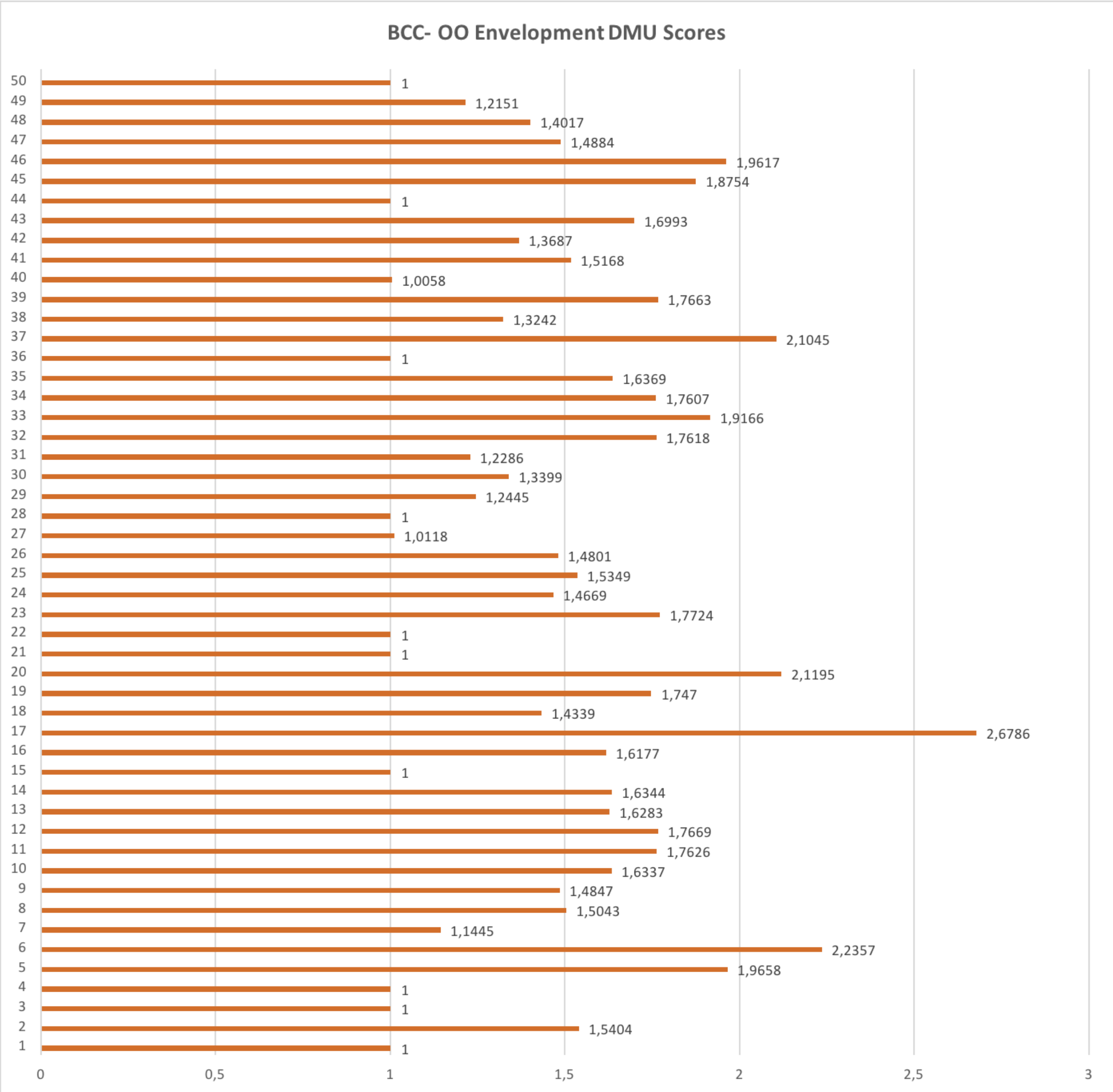

Figure 17: BCC-OO Envelopment form DEA Score Graph

Slacks

BCC - IO Slacks							BCC - OO Scores and Slacks						
DMU	Si-_1	Si-_2	So+_1	So+_2	slacks sum	efficiency sco	DMU	Si-_1	Si-_2	So+_1	So+_2	slacks sum	efficiency score
1	0	0	0	0	0	1	1	0	0	0	0	0	1
2	0	0	0,0222	0	0,0222	0,7986	2	0	0	0	0	0	1,5404
3	0	0,7277	0	0	0,7277	0,8203	3	0,0777	67,2207	0	0	67,2984	1
4	0	0	0	0	0	1	4	0	0	0	0	0	1
5	0	0	0	0	0	0,6995	5	0	0	0	0	0	1,9658
6	0	0	0	0,0802	0,0802	0,6242	6	0,1161	0	0	0	0,1161	2,2357
7	0	34,4054	0	0	34,4054	0,902	7	0	44,0427	0	0	44,0427	1,1445
8	0	44,4207	0	0	44,4207	0,8019	8	0	77,6749	0	0	77,6749	1,5043
9	0	0	0	0	0	0,6732	9	0,0796	0	0	0	0,0796	1,4847
10	0	0	0	0	0	0,5968	10	0,1774	0	0	0	0,1774	1,6337
11	0	0	0	0	0	0,744	11	0	3,7752	0	0	3,7752	1,7626
12	0	0	0	0	0	0,6054	12	0,1376	0	0	0	0,1376	1,7669
13	0	0	0	0	0	0,6878	13	0,0585	14,5537	0	0	14,6122	1,6283
14	0	0	0	0	0	0,818	14	0	0	0	0	0	1,6344
15	0	0	0	0	0	1	15	0	0	0	0	0	1
16	0	0	0,0507	0	0,0507	0,776	16	0	0	0	0	0	1,6177
17	0	0	0	0,292	0,292	0,5489	17	0,2216	0	0	0,0709	0,2925	2,6786
18	0	0	0	0	0	0,807	18	0	1,2343	0	0	1,2343	1,4339
19	0	0	0	0	0	0,5972	19	0,149	0	0	0	0,149	1,747
20	0	0	0,0207	0,0099	0,0306	0,6955	20	0	0	0	0	0	2,1195
21	0	0	0	0	0	1	21	0	0	0	0	0	1
22	0	0	0	0	0	1	22	0	0	0	0	0	1
23	0	0	0	0	0	0,6413	23	0,0787	0	0	0	0,0787	1,7724
24	0	0	0	0	0	0,7571	24	0,0096	0	0	0	0,0096	1,4669
25	0	0	0	0	0	0,8156	25	0	3,1374	0	0	3,1374	1,5349
26	0	0	0	0	0	0,8269	26	0	0	0	0	0	1,4801
27	0	43,1361	0	0	43,1361	0,9952	27	0	44,036	0	0	44,036	1,0118
28	0	0	0	0	0	1	28	0	0	0	0	0	1
29	0	0	0	0	0	0,864	29	0	0	0	0	0	1,2445
30	0	0	0	0	0	0,8625	30	0	0	0	0	0	1,3399
31	0	0	0,2017	0	0,2017	0,7495	31	0,0409	18,3066	0,242	0	18,5895	1,2286
32	0	0	0	0,0002	0,0002	0,558	32	0,2089	0	0	0	0,2089	1,7618
33	0	0	0	0,0092	0,0092	0,6242	33	0,1002	0	0	0	0,1002	1,9166
34	0	0	0	0	0	0,5476	34	0,2249	0	0	0	0,2249	1,7607
35	0	0	0	0	0	0,5311	35	0,2536	17,6198	0	0	17,8734	1,6369
36	0	0	0	0	0	1	36	0	0	0	0	0	1
37	0	0	0,1279	0,0437	0,1716	0,8073	37	0	0	0	0	0	2,1045
38	0	0	0	0,0284	0,0284	0,7355	38	0,119	0	0	0	0,119	1,3242
39	0	0	0	0	0	0,5152	39	0,2784	0	0	0	0,2784	1,7663
40	0	12,4121	0	0	12,4121	0,9977	40	0	12,6732	0	0	12,6732	1,0058
41	0	0	0	0	0	0,6352	41	0,1184	0	0	0	0,1184	1,5168
42	0	0	0	0	0	0,727	42	0,0493	0	0	0	0,0493	1,3687
43	0	0	0	0	0	0,7155	43	0,0142	0	0	0	0,0142	1,6993
44	0	0	0	0	0	1	44	0	0	0	0	0	1
45	0	0	0	0,0601	0,0601	0,7048	45	0,0699	0	0	0	0,0699	1,8754
46	0	0	0	0	0	0,5649	46	0,1518	12,2409	0	0	12,3927	1,9617
47	0	0	0,0034	0	0,0034	0,6877	47	0,0631	0	0	0	0,0631	1,4884
48	0	0	0	0	0	0,6623	48	0,1151	0	0	0	0,1151	1,4017
49	0	0	0	0	0	0,8452	49	0	0	0	0	0	1,2151
50	0	0	0	0	0	1	50	0	0	0	0	0	1
Min:	0	0	0	0	0	0,5152	1	0	0	0	0	0	1
Max:	0	44,4207	0,2017	0,292	44,4207	1	49	0,2784	77,6749	0,242	0,0709	77,6749	2,6786
Average:	0	2,75718367	0,00870612	0,01068776	2,776577551	0,76668571	25	0,05945918	6,45949796	0,00493878	0,00144694	6,52534286	1,505734694
Standard Dev	0	9,99973125	0,03440889	0,04392461	9,994429295	0,1519715	14,2886902	0,08007297	16,8200515	0,03457143	0,01012857	16,8162494	0,387351547
1st quartile	0	0	0	0	0	0,6413	13	0	0	0	0	0	1,2151
2nd quartile	0	0	0	0	0	0,7495	25	0	0	0	0	0,0796	1,5043
3rd quartile	0	0	0	0	0,0284	0,8625	37	0,1151	0	0	0	0,2925	1,7618
4th quartile	0	44,4207	0,2017	0,292	44,4207	1	49	0,2784	77,6749	0,242	0,0709	77,6749	2,6786
1st decile	0	0	0	0	0	0,56352	5,8	0	0	0	0	0	1
2nd decile	0	0	0	0	0	0,6242	10,6	0	0	0	0	0	1,0094
3rd decile	0	0	0	0	0	0,679	15,4	0	0	0	0	0	1,27638
4th decile	0	0	0	0	0	0,70694	20,2	0	0	0	0	0,02122	1,4405
5th decile	0	0	0	0	0	0,7495	25	0	0	0	0	0,0796	1,5043
6th decile	0	0	0	0	0	0,80598	29,8	0,05666	0	0	0	0,11888	1,62618
7th decile	0	0	0	0	0,00688	0,82426	34,6	0,07924	0	0	0	0,2185	1,72792
8th decile	0	0	0	0	0,05446	0,93928	39,4	0,11864	3,39252	0	0	3,39252	1,76654
9th decile	0	0,14554	0,00686	0,0136	0,37914	1	44,2	0,1837	17,75716	0	0	18,01662	1,96252
10th decile	0	44,4207	0,2017	0,292	44,4207	1	49	0,2784	77,6749	0,242	0,0709	77,6749	2,6786

Figure 18: BCC Envelopment form Slacks and Statistics

As with the CCR models, for BCC-IO and BCC- OO s_1^+ are the slacks associated with Input 1: the ratio CurrentLiabilities To Total Assets, s_2^+ are the slacks associated with Input 2: Numbers of Credit intervals, s_1^- are the slacks associated with Output 1: Profit before Tax to Current Liabilities, and s_2^- are the slacks associated with Output 2: Current assets to total liabilities.

If s_1^+ or s_2^+ have positive values it is possible to increase the corresponding inputs with these values without changing the intensity vector or violating any of the program constraints. The inverse applies to s_1^- and s_2^- and the corresponding outputs (Charnes, et. al. 1978).

Intensity Vectors and Benchmarks

The BCC model allows for different returns to scale, and each DMU is benchmarked against others operating under the same scale regime (Banker, Charnes and Cooper, 1984). The benchmark analysis therefore shows which DMUs in the selection operate under the same returns to scale (Cooper et.al, 2007 p.47)

BCC - IO Intensities and Benchmarks

DMU	Benchmarks	λ1	λ2	λ3	λ4	λ5	λ6	λ7	λ8	λ9	λ10	λ11	λ12	λ13	λ14	λ15	λ16	λ17	λ18	λ19	λ20	λ21	λ22	λ23	λ24	λ25	λ26	λ27	λ28	λ29	λ30	λ31	λ32	λ33	λ34	λ35	λ36	λ37	λ38	λ39	λ40	λ41	λ42	λ43	λ44	λ45	λ46	λ47	λ48	λ49	λ50
1	1(1)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	21(0.2058) 44(0.3349) 50(0.4593)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2058	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3349	0	0	0	0	0	0,4593
3	1(0.4147) 4(0.5853)	0,4147	0	0	0,5853	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	4(1)	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1(0.0221) 21(0.2614) 28(0.0916) 50(0.6249)	0,0221	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2614	0	0	0	0	0	0	0,0916	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6249
6	1(0.0278) 21(0.4443) 50(0.528)	0,0278	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4443	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,528
7	28(0.5298) 44(0.3356) 50(0.1346)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5298	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3356	0	0	0	0	0	0,1346
8	28(0.2847) 44(0.1607) 50(0.5546)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2847	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1607	0	0	0	0	0	0,5546
9	21(0.2718) 28(0.186) 44(0.3231) 50(0.219)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2718	0	0	0	0	0	0	0,186	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3231	0	0	0	0	0	0,219
10	1(0.0448) 21(0.5248) 28(0.0684) 50(0.362)	0,0448	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5248	0	0	0	0	0	0	0,0684	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,362
11	1(0.0101) 21(0.1736) 28(0.2261) 50(0.5902)	0,0101	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1736	0	0	0	0	0	0	0,2261	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5902
12	1(0.0654) 21(0.419) 28(0.0859) 50(0.4297)	0,0654	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,419	0	0	0	0	0	0	0,0859	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4297
13	1(0.0878) 21(0.2567) 28(0.2881) 50(0.3674)	0,0878	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2567	0	0	0	0	0	0	0,2881	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3674
14	1(0.1293) 21(0.2471) 28(0.0819) 50(0.5417)	0,1293	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2471	0	0	0	0	0	0	0,0819	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5417
15	15(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	21(0.3103) 44(0.2166) 50(0.4731)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3103	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2166	0	0	0	0	0	0,4731
17	1(0.0133) 21(0.5489) 50(0.4378)	0,0133	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5489	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4378
18	21(0.0709) 28(0.2617) 44(0.2647) 50(0.4027)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0709	0	0	0	0	0	0	0,2617	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2647	0	0	0	0	0	0,4027
19	1(0.0008) 21(0.4603) 28(0.0949) 50(0.4439)	0,0008	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4603	0	0	0	0	0	0	0,0949	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4439
20	21(0.3073) 50(0.6927)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3073	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6927
21	21(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	22(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	21(0.3225) 28(0.1591) 44(0.0575) 50(0.4609)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3225	0	0	0	0	0	0	0,1591	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0575	0	0	0	0	0	0,4609
24	21(0.2783) 28(0.1693) 44(0.278) 50(0.2744)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2783	0	0	0	0	0	0	0,1693	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,278	0	0	0	0	0	0,2744
25	21(0.0875) 28(0.086) 44(0.272) 50(0.5546)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0875	0	0	0	0	0	0	0,086	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,272	0	0	0	0	0	0,5546
26	21(0.1096) 28(0.272) 44(0.1297) 50(0.4887)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1096	0	0	0	0	0	0	0,272	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1297	0	0	0	0	0	0,4887
27	28(0.0901) 44(0.5385) 50(0.3715)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0901	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5385	0	0	0	0	0	0,3715
28	28(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	21(0.1807) 28(0.3765) 44(0.3066) 50(0.1361)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1807	0	0	0	0	0	0	0,3765	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3066	0	0	0	0	0	0,1361
30	21(0.117) 28(0.04) 44(0.4969) 50(0.3461)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,117	0	0	0	0	0	0	0,04	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4969	0	0	0	0	0	0,3461
31	21(0.0016) 22(0.1418) 44(0.8566)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0016	0,1418	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8566	0	0	0	0	0	0
32	1(0.0509) 21(0.5193) 50(0.4298)	0,0509	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5193	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4298
33	1(0.028) 21(0.4298) 50(0.5422)	0,028	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4298	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5422
34	1(0.0837) 21(0.4959) 28(0.0228) 50(0.3976)	0,0837	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4959	0	0	0	0	0	0	0,0228	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3976
35	21(0.3858) 28(0.2635) 44(0.053) 50(0.2977)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3858	0	0	0	0	0	0	0,2635	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,053	0	0	0	0	0	0,2977
36	36(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	21(0.2462) 50(0.7538)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2462	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7538
38	1(0.0461) 21(0.7755) 50(0.1784)	0,0461	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7755	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1784
39	1(0.1094) 21(0.4947) 28(0.037) 50(0.3589)	0,1094	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4947	0	0	0	0	0	0	0,037	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3589
40	1(0.105) 28(0.6056) 50(0.2895)	0,105	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6056	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2895
41	21(0.3171) 28(0.1285) 44(0.3185) 50(0.2358)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3171	0	0	0	0	0	0	0,1285	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3185	0	0	0	0	0	0,2358
42	21(0.2762) 28(0.1757) 44(0.4089) 50(0.1393)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2762	0	0	0	0	0	0	0,1757	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4089	0	0	0	0	0	0,1393
43	21(0.2914) 28(0.1272) 44(0.1224) 50(0.459)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2914	0	0	0	0	0	0	0,1272	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1224	0	0	0	0	0	0,459
44	44(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
45	1(0.0886) 21(0.4892) 50(0.4222)	0,0886	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4892	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4222
46	1(0.0461) 21(0.3449) 28(0.0614) 50(0.5476)	0,0461	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3449	0	0	0	0	0	0	0,0614	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5476
47	21(0.3046) 44(0.4331) 50(0.2623)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3046	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4331	0	0	0	0	0	0,2623
48	21(0.3656) 28(0.134) 44(0.346) 50(0.1545)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3656	0	0	0	0	0	0	0,134	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,346	0	0	0	0	0	0,1545
49	21(0.1759) 28(0.3921) 44(0.4012) 50(0.0308)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1759	0	0	0	0	0	0	0,3921	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4012	0	0	0	0	0	0,0308
50	50(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	Min:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Max:	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0,7538
	Average:	0,0484469390		0	0,032350		0	0	0	0	0	0	0	0	0	0,02	0	0	0	0	0	0,255336735	0,023302041	0	0	0	0	0	0,129385714	0	0	0	0	0	0	0	0,02	0	0	0	0	0	0	0	0,1562142960		0	0	0	0	0,3141489
	Standard Deviation:	0,1537513180		0	0,164020		0	0	0	0	0	0	0	0	0	0,14	0	0	0	0	0	0,222910563	0,143867776	0	0	0	0	0	0,192502482	0	0	0	0	0	0	0	0,14	0	0	0	0	0	0	0	0,2312548860		0	0	0	0	0,2175169
	1st quartile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0016	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1361
	2nd quartile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,25905	0	0	0	0	0	0	0,0649	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3647
Statistics	**3rd quartile**	0,0406	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,38075	0	0	0	0	0	0	0,1741	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,29945	0	0	0	0	0	0,47005
	4th quartile	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1
	1st decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2nd decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,02464
	3rd decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,08252	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,17123
	4th decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,17878	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,28346
	5th decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,25905	0	0	0	0	0	0	0,0649	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3647
	6th decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,29668	0	0	0	0	0	0	0,0907	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,08346	0	0	0	0	0	0,4252
	7th decile	0,02381	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,32922	0	0	0	0	0	0	0,14153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,26689	0	0	0	0	0	0,45909
	8th decile	0,04706	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4327	0	0	0	0	0	0	0,23322	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,32546	0	0	0	0	0	0,53074
	9th decile	0,09024	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,49824	0	0	0	0	0	0	0,29694	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,41132	0	0	0	0	0	0,55816
	10th decile	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1

Figure 19: BCC- IO Benchmarks, Intensity Vectors, and statistics

		BCC - IO Intensities and Benchmarks																																																	
DMU	Benchmarks	λ1	λ2	λ3	λ4	λ5	λ6	λ7	λ8	λ9	λ10	λ11	λ12	λ13	λ14	λ15	λ16	λ17	λ18	λ19	λ20	λ21	λ22	λ23	λ24	λ25	λ26	λ27	λ28	λ29	λ30	λ31	λ32	λ33	λ34	λ35	λ36	λ37	λ38	λ39	λ40	λ41	λ42	λ43	λ44	λ45	λ46	λ47	λ48	λ49	λ50
1	1(1)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	21(0.2058) 44(0.3349) 50(0.4593)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2058	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3349	0	0	0	0	0	0,4593
3	1(0.4147) 4(0.5853)	0,4147	0	0	0,5853	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	4(1)	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1(0.0221) 21(0.2614) 28(0.0916) 50(0.6249)	0,0221	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2614	0	0	0	0	0	0	0,0916	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6249
6	1(0.0278) 21(0.4443) 50(0.528)	0,0278	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4443	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,528
7	28(0.5298) 44(0.3356) 50(0.1346)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5298	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3356	0	0	0	0	0	0,1346
8	28(0.2847) 44(0.1607) 50(0.5546)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2847	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1607	0	0	0	0	0	0,5546
9	21(0.2718) 28(0.186) 44(0.3231) 50(0.219)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2718	0	0	0	0	0	0	0,186	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3231	0	0	0	0	0	0,219
10	1(0.0448) 21(0.5248) 28(0.0684) 50(0.362)	0,0448	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5248	0	0	0	0	0	0	0,0684	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,362
11	1(0.0101) 21(0.1736) 28(0.2261) 50(0.5902)	0,0101	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1736	0	0	0	0	0	0	0,2261	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5902
12	1(0.0654) 21(0.419) 28(0.0859) 50(0.4297)	0,0654	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,419	0	0	0	0	0	0	0,0859	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4297
13	1(0.0878) 21(0.2567) 28(0.2881) 50(0.3674)	0,0878	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2567	0	0	0	0	0	0	0,2881	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3674
14	1(0.1293) 21(0.2471) 28(0.0819) 50(0.5417)	0,1293	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2471	0	0	0	0	0	0	0,0819	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5417
15	15(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	21(0.3103) 44(0.2166) 50(0.4731)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3103	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2166	0	0	0	0	0	0,4731
17	1(0.0133) 21(0.5489) 50(0.4378)	0,0133	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5489	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4378
18	21(0.0709) 28(0.2617) 44(0.2647) 50(0.4027)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0709	0	0	0	0	0	0	0,2617	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2647	0	0	0	0	0	0,4027
19	1(0.0008) 21(0.4603) 28(0.0949) 50(0.4439)	0,0008	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4603	0	0	0	0	0	0	0,0949	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4439
20	21(0.3073) 50(0.6927)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3073	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6927
21	21(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	22(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	21(0.3225) 28(0.1591) 44(0.0575) 50(0.4609)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3225	0	0	0	0	0	0	0,1591	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0575	0	0	0	0	0	0,4609
24	21(0.2783) 28(0.1693) 44(0.278) 50(0.2744)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2783	0	0	0	0	0	0	0,1693	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,278	0	0	0	0	0	0,2744
25	21(0.0875) 28(0.086) 44(0.272) 50(0.5546)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0875	0	0	0	0	0	0	0,086	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,272	0	0	0	0	0	0,5546
26	21(0.1096) 28(0.272) 44(0.1297) 50(0.4887)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1096	0	0	0	0	0	0	0,272	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1297	0	0	0	0	0	0,4887
27	28(0.0901) 44(0.5385) 50(0.3715)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0901	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5385	0	0	0	0	0	0,3715
28	28(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	21(0.1807) 28(0.3765) 44(0.3066) 50(0.1361)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1807	0	0	0	0	0	0	0,3765	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3066	0	0	0	0	0	0,1361
30	21(0.117) 28(0.04) 44(0.4969) 50(0.3461)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,117	0	0	0	0	0	0	0,04	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4969	0	0	0	0	0	0,3461
31	21(0.0016) 22(0.1418) 44(0.8566)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0016	0,1418	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8566	0	0	0	0	0	0
32	1(0.0509) 21(0.5193) 50(0.4298)	0,0509	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5193	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4298
33	1(0.028) 21(0.4298) 50(0.5422)	0,028	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4298	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5422
34	1(0.0837) 21(0.4959) 28(0.0228) 50(0.3976)	0,0837	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4959	0	0	0	0	0	0	0,0228	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3976
35	21(0.3858) 28(0.2635) 44(0.053) 50(0.2977)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3858	0	0	0	0	0	0	0,2635	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,053	0	0	0	0	0	0,2977
36	36(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	21(0.2462) 50(0.7538)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2462	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7538
38	1(0.0461) 21(0.7755) 50(0.1784)	0,0461	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,7755	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1784
39	1(0.1094) 21(0.4947) 28(0.037) 50(0.3589)	0,1094	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4947	0	0	0	0	0	0	0,037	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3589
40	1(0.105) 28(0.6056) 50(0.2895)	0,105	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,6056	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2895
41	21(0.3171) 28(0.1285) 44(0.3185) 50(0.2358)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3171	0	0	0	0	0	0	0,1285	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3185	0	0	0	0	0	0,2358
42	21(0.2762) 28(0.1757) 44(0.4089) 50(0.1393)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2762	0	0	0	0	0	0	0,1757	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4089	0	0	0	0	0	0,1393
43	21(0.2914) 28(0.1272) 44(0.1224) 50(0.459)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,2914	0	0	0	0	0	0	0,1272	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1224	0	0	0	0	0	0,459
44	44(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
45	1(0.0886) 21(0.4892) 50(0.4222)	0,0886	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4892	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4222
46	1(0.0461) 21(0.3449) 28(0.0614) 50(0.5476)	0,0461	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3449	0	0	0	0	0	0	0,0614	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,5476
47	21(0.3046) 44(0.4331) 50(0.2623)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3046	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4331	0	0	0	0	0	0,2623
48	21(0.3656) 28(0.134) 44(0.346) 50(0.1545)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3656	0	0	0	0	0	0	0,134	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,346	0	0	0	0	0	0,1545
49	21(0.1759) 28(0.3921) 44(0.4012) 50(0.0308)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1759	0	0	0	0	0	0	0,3921	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4012	0	0	0	0	0	0,0308
50	50(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	Min:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Max:	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0,7538
	Average:	0,0484469	0	0	0,0323	0	0	0	0	0	0	0	0	0	0	0,02	0	0	0	0	0	0,255336735	0,023302041	0	0	0	0	0	0,129385714	0	0	0	0	0	0	0	0,02	0	0	0	0	0	0	0	0,1562142	0	0	0	0	0	0,3141489
	Standard Deviation:	0,1537513	0	0	0,164	0	0	0	0	0	0	0	0	0	0	0,14	0	0	0	0	0	0,222910563	0,143867776	0	0	0	0	0	0,192502482	0	0	0	0	0	0	0	0,14	0	0	0	0	0	0	0	0,2312548	0	0	0	0	0	0,2175169
	1st quartile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,0016	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,1361
	2nd quartile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,25905	0	0	0	0	0	0	0,0649	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3647
Statistics	3rd quartile	0,0406	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,38075	0	0	0	0	0	0	0,1741	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,29945	0	0	0	0	0	0,47005
	4th quartile	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1
	1st decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2nd decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,02464
	3rd decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,08252	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,17123
	4th decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,17878	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,28346
	5th decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,25905	0	0	0	0	0	0	0,0649	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,3647
	6th decile	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,29668	0	0	0	0	0	0	0,0907	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,08346	0	0	0	0	0	0,4252
	7th decile	0,02381	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,32922	0	0	0	0	0	0	0,14153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,26689	0	0	0	0	0	0,45909
	8th decile	0,04706	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,4327	0	0	0	0	0	0	0,23322	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,32546	0	0	0	0	0	0,53074
	9th decile	0,09024	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,49824	0	0	0	0	0	0	0,29694	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,41132	0	0	0	0	0	0,55816
	10th decile	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1

Figure 20: BCC- OO Benchmarks, Intensity Vectors and statistics

Correlation Analysis

The IO and OO versions of the BCC model show lower degrees of correlation than the IO and OO versions of the CCR model.

BCC IO and OO Correlation Analysis: Slacks and Efficiency Score					
Si-_1	Si-_2	So+_1	So+_2	slacks sum	efficiency score
#DIV/0!	0,768207509	0,81798851	0,933930691	0,767409402	-0,797194797

Figure 21: BCC Envelopment form IO and OO Correlation Analysis

Classification

BCC - IO Classifications		BCC - OO Classifications	
DMU	**Classification**	**DMU**	**Classification**
1	Fully efficient	1	Fully efficient
4	Fully efficient	4	Fully efficient
15	Fully efficient	15	Fully efficient
21	Fully efficient	21	Fully efficient
22	Fully efficient	22	Fully efficient
28	Fully efficient	28	Fully efficient
36	Fully efficient	36	Fully efficient
44	Fully efficient	44	Fully efficient
50	Fully efficient	50	Fully efficient
2	Technically inefficient but mix-efficient	31	Inefficient
3	Technically inefficient but mix-efficient	3	Technically efficient but mix efficient
5	Technically inefficient but mix-efficient	2	Technically inefficient but mix-efficient
6	Technically inefficient but mix-efficient	5	Technically inefficient but mix-efficient
7	Technically inefficient but mix-efficient	6	Technically inefficient but mix-efficient
8	Technically inefficient but mix-efficient	7	Technically inefficient but mix-efficient
9	Technically inefficient but mix-efficient	8	Technically inefficient but mix-efficient
10	Technically inefficient but mix-efficient	9	Technically inefficient but mix-efficient
11	Technically inefficient but mix-efficient	10	Technically inefficient but mix-efficient
12	Technically inefficient but mix-efficient	11	Technically inefficient but mix-efficient
13	Technically inefficient but mix-efficient	12	Technically inefficient but mix-efficient
14	Technically inefficient but mix-efficient	13	Technically inefficient but mix-efficient
16	Technically inefficient but mix-efficient	14	Technically inefficient but mix-efficient
17	Technically inefficient but mix-efficient	16	Technically inefficient but mix-efficient
18	Technically inefficient but mix-efficient	17	Technically inefficient but mix-efficient
19	Technically inefficient but mix-efficient	18	Technically inefficient but mix-efficient
20	Technically inefficient but mix-efficient	19	Technically inefficient but mix-efficient
23	Technically inefficient but mix-efficient	20	Technically inefficient but mix-efficient
24	Technically inefficient but mix-efficient	23	Technically inefficient but mix-efficient
25	Technically inefficient but mix-efficient	24	Technically inefficient but mix-efficient
26	Technically inefficient but mix-efficient	25	Technically inefficient but mix-efficient
27	Technically inefficient but mix-efficient	26	Technically inefficient but mix-efficient
29	Technically inefficient but mix-efficient	27	Technically inefficient but mix-efficient
30	Technically inefficient but mix-efficient	29	Technically inefficient but mix-efficient
31	Technically inefficient but mix-efficient	30	Technically inefficient but mix-efficient
32	Technically inefficient but mix-efficient	32	Technically inefficient but mix-efficient
33	Technically inefficient but mix-efficient	33	Technically inefficient but mix-efficient
34	Technically inefficient but mix-efficient	34	Technically inefficient but mix-efficient
35	Technically inefficient but mix-efficient	35	Technically inefficient but mix-efficient
37	Technically inefficient but mix-efficient	37	Technically inefficient but mix-efficient
38	Technically inefficient but mix-efficient	38	Technically inefficient but mix-efficient
39	Technically inefficient but mix-efficient	39	Technically inefficient but mix-efficient
40	Technically inefficient but mix-efficient	40	Technically inefficient but mix-efficient
41	Technically inefficient but mix-efficient	41	Technically inefficient but mix-efficient
42	Technically inefficient but mix-efficient	42	Technically inefficient but mix-efficient
43	Technically inefficient but mix-efficient	43	Technically inefficient but mix-efficient
45	Technically inefficient but mix-efficient	45	Technically inefficient but mix-efficient
46	Technically inefficient but mix-efficient	46	Technically inefficient but mix-efficient
47	Technically inefficient but mix-efficient	47	Technically inefficient but mix-efficient
48	Technically inefficient but mix-efficient	48	Technically inefficient but mix-efficient
49	Technically inefficient but mix-efficient	49	Technically inefficient but mix-efficient

Figure 22: BCC Classifications

Projection of DMUs on efficiency frontier

An important difference between the projections of the BCC- IO and CCR-IO is the projections for DMU 22 and 50 which are classified as Technically efficient under BBC-IO. The projections, i.e. the needed improvement in these DMUs is therefore eliminated in the BCC-IO projections suggesting that the efficiency discrepancies observed under CCR-IO were purely due to returns to scale.

This Illustrates the applicability of the BCC model to managerial contexts. Although the CCR model might identify improvement potentials in the firm, it is fruitless to address these with efficiency improvement interventions if the observed inefficiencies are only due to returns to scale, which usually remain fixed in the short run. This way, the BCC model can assist managers rule out and prioritize areas of improvement.

BCC - IO Projections

DMU	P(x1)	Δx1	%	P(x2)	Δx2	%	P(y1)	Δy1	%	P(y2)	Δy2	%
1	0,4917	0	0	233,34	0	0	4,619	0	0	0,149	0	0
2	0,4169	-0,133	-24,177	143,56	-36,2272	-20,152	1,543	0	0	0,355	0	0
3	0,5397	-0,118	-17,97	210,8	-46,1799	-17,97	4,619	0	0	0,998	0,728	268,7918
4	0,5738	0	0	193,61	0	0	4,619	0	0	0,357	0	0
5	0,4198	-0,18	-30,05	139,9	-60,1013	-30,05	1,75	0	0	0,259	0	0
6	0,4314	-0,26	-37,58	103,65	-62,4527	-37,58	1,593	0	0	0,194	0	0
7	0,4776	-0,052	-9,8	232,04	-25,2102	-9,8	2,667	0	0	34,81	34,41	8455,897
8	0,428	-0,106	-19,81	235,96	-58,2923	-19,81	2,102	0	0	44,73	44,42	14532,95
9	0,4676	-0,227	-32,68	137,19	-66,5972	-32,68	1,964	0	0	0,392	0	0
10	0,451	-0,305	-40,32	91,842	-62,0488	-40,32	1,801	0	0	0,297	0	0
11	0,424	-0,146	-25,6	159,6	-54,9159	-25,6	1,987	0	0	0,267	0	0
12	0,4429	-0,289	-39,46	112,86	-73,5594	-39,46	1,891	0	0	0,282	0	0
13	0,4502	-0,204	-31,22	150	-68,0851	-31,22	2,373	0	0	0,287	0	0
14	0,4294	-0,096	-18,2	147,51	-32,82	-18,2	2,066	0	0	0,25	0	0
15	0,5991	0	0	178,57	0	0	4,619	0	0	0,021	0	0
16	0,3869	-0,177	-31,39	125,27	-36,2109	-22,431	1,496	0	0	0,331	0	0
17	0,4421	-0,363	-45,11	83,466	-68,8342	-45,11	1,558	0	0	0	0	
18	0,4458	-0,107	-19,3	176,93	-42,3151	-19,3	2,088	0	0	0,35	0	0
19	0,4413	-0,298	-40,28	102,34	-69,0266	-40,28	1,711	0	0	0,294	0	0
20	0,3911	-0,201	-33,946	127,54	-55,866	-30,461	1,47	0	0	0,243	0	0
21	0,4941	0	0	0,1	0	0	1,565	0	0	0,366	0	0
22	0,6454	0	0	225,62	0	0	2,182	0	0	0,711	0	0
23	0,4393	-0,246	-35,87	129,17	-72,248	-35,87	1,844	0	0	0,302	0	0
24	0,4611	-0,148	-24,29	135,84	-43,5823	-24,29	1,918	0	0	0,374	0	0
25	0,4279	-0,097	-18,44	168,46	-38,0882	-18,44	1,719	0	0	0,328	0	0
26	0,4359	-0,091	-17,31	171,22	-35,8432	-17,31	2,079	0	0	0,311	0	0
27	0,4492	-0,002	-0,48	225,69	-1,0885	-0,48	1,787	0	0	43,54	43,14	10566,52
28	0,4939	0	0	214,7	0	0	3,577	0	0	0,363	0	0
29	0,4775	-0,075	-13,6	159,92	-25,1719	-13,6	2,354	0	0	0,402	0	0
30	0,4523	-0,072	-13,75	159,82	-25,4782	-13,75	1,683	0	0	0,405	0	0
31	0,3127	-0,374	-54,441	182,83	-61,3065	-25,133	1,579	0	0	0,579	0	0
32	0,443	-0,351	-44,2	91,072	-72,1393	-44,2	1,674	0	0	0,285	0	0
33	0,4297	-0,259	-37,58	106,39	-64,0578	-37,58	1,592	0	0	0,263	0	0
34	0,4466	-0,369	-45,24	97,675	-80,6938	-45,24	1,824	0	0	0,283	0	0
35	0,4586	-0,405	-46,89	120,76	-106,617	-46,89	2,071	0	0	0,327	0	0
36	0,5422	0	0	192,2	0	0	4,619	0	0	0,223	0	0
37	0,2767	-0,225	-44,792	138,76	-33,2599	-19,344	1,356	0	0	0,2	0	0
38	0,4728	-0,17	-26,45	43,643	-15,705	-26,45	1,687	0	0	0,299	0	0
39	0,4513	-0,425	-48,48	99,611	-93,7331	-48,48	1,935	0	0	0,284	0	0
40	0,4594	-0,001	-0,23	220,22	-0,5077	-0,23	3,072	0	0	12,71	12,41	4214,27
41	0,4657	-0,267	-36,48	127,13	-73,0124	-36,48	1,846	0	0	0,388	0	0
42	0,477	-0,179	-27,3	135,38	-50,8372	-27,3	1,965	0	0	0,421	0	0
43	0,4394	-0,175	-28,45	133,41	-53,0457	-28,45	1,79	0	0	0,314	0	0
44	0,4927	0	0	176,09	0	0	1,715	0	0	0,557	0	0
45	0,4438	-0,186	-29,52	98,402	-41,2402	-29,52	1,79	0	0	0,218	0	0
46	0,4289	-0,33	-43,51	124,79	-96,1168	-43,51	1,77	0	0	0,267	0	0
47	0,459	-0,213	-31,736	124,59	-56,5843	-31,232	1,598	0	0	0,406	0	0
48	0,4753	-0,242	-33,77	118,17	-60,2516	-33,77	1,87	0	0	0,407	0	0
49	0,4898	-0,09	-15,48	160,53	-29,4009	-15,48	2,411	0	0	0,437	0	0
50	0,3753	0	0	184,11	0	0	1,457	0	0	0,203	0	0

BCC- OO Projections

DMU	P(x1)	Δx1	%	P(x2)	Δx2	%	P(y1)	Δy1	%	P(y2)	Δy2	%
1	0,4917	0	0	233,3411	0	0	4,6188	0	0	0,1494	0	0
2	0,5499	0	0	179,7666	0	0	1,002	-0,5414	-35,08	0,2307	-0,1247	-35,08
3	0,658	0	0	256,9836	0	0	4,6965	0,0777	1,6823	67,491	0,0777	28,7
4	0,5738	0	0	193,6117	0	0	4,6188	0	0	0,3567	0	0
5	0,6002	0	0	200,0043	0	0	0,89	-0,8595	-49,13	0,1319	-0,1274	-49,13
6	0,691	0	0	166,1859	0	0	0,8286	-0,7643	-47,982	0,0868	0,0089	4,5789
7	0,5295	0	0	257,2466	0	0	2,3298	-0,3368	-12,63	44,398	-0,0514	-12,63
8	0,5337	0	0	294,2567	0	0	1,3974	-0,7046	-33,52	77,878	-0,1025	-33,52
9	0,6946	0	0	203,7859	0	0	1,4024	-0,5617	-28,597	0,2639	-0,0483	-12,332
10	0,7557	0	0	153,8909	0	0	1,2795	-0,521	-28,938	0,182	0,062	20,866
11	0,5699	0	0	214,5154	0	0	1,1272	-0,8598	-43,27	3,9268	-0,1156	-43,27
12	0,7316	0	0	186,4152	0	0	1,2081	-0,6832	-36,125	0,1595	0,0153	5,4299
13	0,6545	0	0	218,0818	0	0	1,5158	-0,8573	-36,125	14,73	-0,0521	-18,171
14	0,525	0	0	180,3295	0	0	1,2641	-0,8021	-38,82	0,1527	-0,0969	-38,82
15	0,5991	0	0	178,5676	0	0	4,6188	0	0	0,0206	0	0
16	0,5639	0	0	161,4293	0	0	0,9247	-0,5711	-38,18	0,2043	-0,1262	-38,18
17	0,8054	0	0	152,5209	-0,0709	-0,0465	0,8033	-0,755	-48,45	0	0,2216	65535
18	0,5525	0	0	219,2492	0	0	1,4558	-0,6317	-30,26	1,4787	-0,106	-30,26
19	0,739	0	0	171,367	0	0	1,1282	-0,5825	-34,05	0,168	0,0235	8,0145
20	0,5922	0	0	183,4001	0	0	0,6933	-0,7762	-52,82	0,1148	-0,1286	-52,82
21	0,4941	0	0	0,1	0	0	1,5654	0	0	0,3664	0	0
22	0,6454	0	0	225,6247	0	0	2,1821	0	0	0,7111	0	0
23	0,685	0	0	201,4163	0	0	1,1191	-0,7249	-39,312	0,1702	-0,0528	-17,489
24	0,609	0	0	179,4247	0	0	1,3169	-0,6008	-31,329	0,2551	-0,1095	-29,264
25	0,5246	0	0	206,5521	0	0	1,1198	-0,599	-34,85	3,3508	-0,1141	-34,85
26	0,5271	0	0	207,0665	0	0	1,4044	-0,6743	-32,44	0,2098	-0,1007	-32,44
27	0,4514	0	0	226,7736	0	0	1,7658	-0,0209	-1,17	44,44	-0,0048	-1,17
28	0,4939	0	0	214,7026	0	0	3,5766	0	0	0,3634	0	0
29	0,5526	0	0	185,0875	0	0	1,8912	-0,4625	-19,65	0,3227	-0,0789	-19,65
30	0,5244	0	0	185,296	0	0	1,2556	-0,4268	-25,37	0,302	-0,1026	-25,37
31	0,4443	-0,242	-35,263	243,9313	0	0	1,3261	-0,253	-16,02	18,778	-0,0668	-11,543
32	0,7938	0	0	163,2112	0	0	1,1592	-0,515	-30,763	0,1618	0,0857	30,052
33	0,6884	0	0	170,4573	0	0	0,9309	-0,6611	-41,526	0,137	-0,0254	-9,6676
34	0,8156	0	0	178,3683	0	0	1,2608	-0,563	-30,868	0,1609	0,1025	36,202
35	0,8636	0	0	227,3777	0	0	1,5187	-0,5522	-26,664	17,82	0,1263	38,614
36	0,5422	0	0	192,1987	0	0	4,6188	0	0	0,2233	0	0
37	0,5011	0	0	171,9356	0	0	0,6442	-0,7115	-52,48	0,0948	-0,1047	-52,48
38	0,6428	0	0	59,3762	0	0	1,3928	-0,2939	-17,425	0,2257	0,0458	15,334
39	0,8759	0	0	193,3439	0	0	1,374	-0,561	-28,993	0,1608	0,1552	54,662
40	0,4604	0	0	220,7288	0	0	3,0546	-0,0178	-0,58	12,966	-0,0017	-0,58
41	0,7331	0	0	200,1435	0	0	1,3354	-0,5105	-27,656	0,256	-0,0139	-3,5793
42	0,6561	0	0	186,2168	0	0	1,4847	-0,48	-24,431	0,3077	-0,0642	-15,235
43	0,6141	0	0	186,4522	0	0	1,0674	-0,7223	-40,357	0,185	-0,1152	-36,634
44	0,4927	0	0	176,0903	0	0	1,7148	0	0	0,5573	0	0
45	0,6297	0	0	139,7025	0	0	1,0244	-0,7657	-42,775	0,1163	-0,0319	-14,641
46	0,7593	0	0	220,9075	0	0	1,0543	-0,716	-40,446	12,377	0,021	7,876
47	0,6724	0	0	181,175	0	0	1,137	-0,4613	-28,862	0,273	-0,0702	-17,277
48	0,7176	0	0	178,4175	0	0	1,449	-0,4208	-22,504	0,2902	-0,0015	-0,3675
49	0,5795	0	0	189,9284	0	0	1,9838	-0,4267	-17,7	0,3595	-0,0773	-17,7
50	0,3753	0	0	184,1089	0	0	1,457	0	0	0,2031	0	0

Figure 23: BCC Projections

0.4 Task 4

0.4.1 Matrix representation of the Barnum et. al. models in envelopment form

Barnum model in Matrix Format

Barnum et. al. (2017) relaxed the assumption that neither inputs nor outputs are substitutable amongs themselves. Below is the input oriented model of its dual formulation:

$$\text{Minimize z} = \begin{bmatrix} 0_{1\times J} & 1_{1\times 1} - \varepsilon_{1\times N} & -\varepsilon_{1\times M} \end{bmatrix} \begin{bmatrix} \alpha_{J\times 1} \\ \beta_{1\times 1} \\ s^{-}_{(N+M)\times 1} \end{bmatrix}$$
$$\text{s.t.} \begin{bmatrix} 0_{M\times 1} & Y^t_{M\times J} I_{((M+N)/2\times(M+N)/2)} 0_{((M+N)/2\times(M+N)/2)} \\ X^t_{N\times 1} & -X^t_{N\times J} 0_{((M+N)/2\times(M+N)/2)} I_{((M+N)/2\times(M+N)/2)} \\ 0_{1\times 1} & 1_{1\times J} 0_{1\times(M+N)} \end{bmatrix} \begin{bmatrix} \alpha_{J\times 1} \\ \beta_{1\times 1} \\ s^{-}_{(N+M)\times 1} \end{bmatrix}$$
$$= \begin{bmatrix} Y^t_{M\times 1} \\ 0_{N\times 1} \\ 1_{1\times 1} \end{bmatrix} \tag{7}$$

Comment on structures of the constraints matrix

0 $M \times 1$ denotes a vector of zeros of those dimensions, while Y^t denotes a matrix of input values of $M \times J$ dimensions. The block of I and 0 matricies of $((M+N)/2 \times (M+N)/2)$ dimensions combine to an identity matrix assigning the negative slack values in the decision vector to different rows. X^t denotes a matrix of input variables of $N \times 1$ dimensions, while $-X^t$ denotes a matrix of negative input variables of $(N \times J)$ dimensions. 0 1×1 is a zero, while 1 $1 \times J$ is a vector of ones. 0 $1 \times (M+N)$ is a vector of Zeros.

0.4.2 Pseudo Code

Algorithm 7 Analysis 7 - Barnum in Envelopment Form (env_barnum_IO.m)

procedure ANALYSIS 7 - BARNUM IN ENVELOPMENT FORM (ENV_BARNUM_IO.M)

Declare an empty matrix Z that will hold the results

Declare the base column title that will be used for the output files in a cell array variable declared as "column_headers"

Declare the objective function "f"

Declare a vector of 0s that serves as a lower bound for the solution vector

For each DMU
Find its Aeq
Find its beq
Solve the optimization problem for the objective function
Append the achieved result to the solution matrix Z

Use the custom function "calculate_benchmarks" to save tuples of the type "benchmark group member 1 (benchmark value), benchmark group member 2 (benchmark value) ..." into a matrix declared as "benchmarks"

Use the custom function "calculate_frontier_io" to save frontier related information in a matrix variable declared as "eff_frontier"

Loop from 1 to n using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "αi"

Append the string element "β" to the cell array variable "column_headers"

Loop from 1 to m using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "Si-i" which stands for slack values

Loop from 1 to s using a variable "i" and horizontally append a string entry to the cell array variable "column_headers" in the form "So+i" which stands for slack values

Append needed elements "slacks sum" and "efficiency score" to the "column_headers" vector variable

Horizontally concatenate the matrix variable “benchmarks” with the λ values from the matrix Z

Assign the return value of the custom function “sum_slacks” to the variable Z

Save the current file name as a string into the variable “filename”

Execute the custom function “export” once for each of the result variables - “Z”, “benchmarks”, “eff_frontier”

0.4.3 Matlab Implementation

Please find the MATLAB code used in Project 1 Task 1 in the appropriate folder on the memory stick, files titled ”RUN.m”, ”env_barnum_IO.m” .

In addition, open functions ”calculate_benchmarks.m”, ”calculate_frontier_io.m”, ”calculate_frontier_barnum_io.m”, ”calculate_frontier_oo.m” , ”classify.m”, ”export.m”, ”optimize.m” for all analyses in project 1.

Opening all matlab files and executing ”RUN.m” returns analysis output excel results for all tasks.

0.5 References

Banker, R., Charnes, A., and Cooper, W. (1984). Some Models for Estimating Technical and Scale Inefficiencies in Data Envelopment Analysis. Management Science, 30(9), 1078-1092.

Barnum D., Coupet J., Gleason J., McWilliams A. and Parhankangas A. (2017), Impact of input substitution and output transformation on data envelopment analysis decisions. Applied Economics. 49(15), 1543-1556.

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

Cooper, W.W. et al., (2007). Data envelopment analysis a comprehensive text with models, applications, references and DEA-solver software. Second.ed, New York: Springer.