A Multi-objective Optimisation Approach to Compute the Efficient Frontier in Data Envelopment Analysis – Running title: An MOO Approach to Compute the Efficient Frontier in DEA

Matthias Ehrgott[a,*], Maryam Hasannasab[b], Andrea Raith[c]

[a] *Department of Management Science, Lancaster University, Lancaster LA1 4XY, United Kingdom*
[b] *Department of Accounting and Finance, The University of Auckland, Auckland, New Zealand*
[c] *Department of Engineering Science, The University of Auckland, Auckland 1142, New Zealand*

Abstract

Data envelopment analysis is a linear programming-based operations research technique for performance measurement of decision-making units. In this paper, we investigate data envelopment analysis from a multi-objective point of view to compute both the efficient extreme points and efficient facets of the technology set simultaneously. We introduce a dual multi-objective linear programming formulation of data envelopment analysis in terms of input and output prices and propose a procedure based on objective space algorithms for multi-objective linear programmes to compute the efficient frontier. We show that using our algorithm, the efficient extreme points and facets of the technology set can be computed without solving any optimisation problems. We conduct computational experiments to demonstrate that the algorithm can compute the efficient frontier within seconds to a few minutes of computation time for real world data envelopment analysis instances. For large scale artificial data sets our algorithm is faster than computing the efficiency scores of all decision making units via linear programming.

Keywords: Data envelopment analysis, efficient frontier, linear programming, multi-objective optimisation, duality, objective space algorithm

1. Introduction

Data envelopment analysis (DEA), which assesses the efficiency of a group of comparable decision-making units (DMUs) with common inputs and outputs was originally introduced by Charnes et al. (1978) for a constant returns to scale technology. Banker et al. (1984) extended the Charnes et al. formulation to assess the efficiency of a DMU for the variable returns to scale case. Other formulations are possible, see Cooper et al. (2007) for more details on DEA models and their extensions. These DEA models require the solution of a linear programme (LP) for each

*Corresponding author, phone number +44 1524 593888
Email addresses: m.ehrgott@lancaster.ac.uk (Matthias Ehrgott), maryam.hasannasab@auckland.ac.nz (Maryam Hasannasab), a.raith@auckland.ac.nz (Andrea Raith)

DMU to determine its efficiency score.

Dulá (2002) provides an overview of computational aspects of DEA, including preprocessing, standard procedures and enhancements. He shows that the number of DMUs, the number of inputs and outputs as well as the proportion of efficient DMUs are the determinants of the computational effort to run DEA on a data set. The number and size of LPs that have to be solved clearly grow as the number of DMUs, inputs, and outputs grow. The proportion of efficient DMUs also affects computation time in enhanced implementations of the standard approach (Dulá 2002). This has been confirmed by Dulá (2008), who presents a comprehensive computational study involving DEA problems with up to 100,000 DMUs. Dulá (2008) explores the impact of different LP algorithms including interior point methods as well as acceleration techniques and DEA-specific enhancements. Dulá (2011) presents a two-phase algorithm for DEA, which first identifies the extreme efficient DMUs and then scores the remaining ones. The algorithm is tested on large data sets, similar to those used in Dulá (2008). Furthermore, Bougnol et al. (2012) propose the use of interior point algorithms specifically to obtain non-zero multiplier weights.

In this paper, we are interested in the computation of the efficient frontier of the DEA technology set, introducing an algorithm to compute both the extreme points and efficient facets. Because the DEA technology set is a polyhedron, it can be described internally by means of its extreme points and extreme rays, or externally, by means of its facets. Switching from one description to the other is a problem studied in computational geometry. It is combinatorial in nature, hence algorithms to solve it have exponential running time in the worst case. Nevertheless, the computation of the efficient facets of a technology set is of interest in DEA. This is because, given the facets of the technology set, the computation of many other quantities of interest is straightforward and possible in closed form, see e.g. Olesen and Petersen (2003). Moreover, in research investigating the existence of well defined efficiency measures based on closest projections to the efficient frontier, Aparicio and Pastor (2013, 2014) work with efficient facets to provide answers.

Several strategies to obtain facets of the technology set associated with efficient DMUs have been proposed. Yu, Wei, Brockett and Zhou (1996) present an enumerative tree search algorithm that identifies those subsets of DMUs, which are on the same facet. The algorithm solves LPs related to subsets of DMUs to check whether the subsets define efficient facets. Olesen and Petersen (2003) propose two methods. In the first one, they first identify the extreme efficient DMUs by solving LPs. Then mixed integer LPs are solved for identification of all facets containing each of those DMUs. The second approach is convex hull generation. Briec and Leleu (2003) also use convex hull algorithms. Jahanshahloo et al. (2005) present an approach that repeatedly solves a 0-1 programme to find the efficient facets of the technology set. Jahanshahloo et al. (2007) offer an enumerative method that first finds all efficient DMUs then identifies those that share the same facet and finally constructs that facet. Davtalab-Olyaie et al. (2014) first identify the extreme

efficient DMUs and then perform a procedure that requires the repeated solution of mixed integer programmes to identify the efficient facets. Jahanshahloo et al. (2010) propose an algorithm that finds weakly efficient facets of the technology set. In the same vein, Davtalab-Olyaie et al. (2015) propose an algorithm to find weakly efficient facets. While all of these papers present numerical examples, no detailed numerical studies are presented in any of them. We also note that all of these approaches first compute efficient DMUs and then explore either subsets of them or solve mixed integer linear programmes to identify sets of DMUs that define efficient facets of the technology set.

Our approach for identifying efficient facets of the technology set is to consider DEA from the point of view of multi-objective optimisation, in particular multi-objective linear programming (MOLP). Relationships between DEA and MOLP have been explored in the past. Stewart (1996) demonstrates a link between ratio efficiency in DEA and a distance measure in input-output space based on a linear value function. Joro et al. (1998) show that structurally the DEA formulation to identify efficient DMUs is quite similar to MOLP models based on the reference point or the reference direction approach to generate efficient solutions. Yun et al. (2001) combine generalised DEA and genetic algorithms for (approximately) generating efficient frontiers in multi-objective optimisation problems, while Yougbaré and Teghem (2007) establish relationships between the notions of DEA efficiency and Pareto optimal solutions in multi-objective optimisation problems. Most relevant for our research is the work of Charnes et al. (1985), Yu, Wei and Brockett (1996) and Hosseinzadeh Lotfi et al. (2008), who propose a formulation of DEA as an MOLP, which we adopt.

An MOLP is an optimisation problem with multiple, conflicting, linear objectives, linear constraints, and continuous variables. The area of MOLP has attracted the attention of researchers since the 1960s, and readers are referred to Ehrgott (2005) and Steuer (1985) for introductions to multi-objective optimisation in general and MOLP in particular. These works refer to extensions of the simplex method for solving MOLPs. For this paper, however, a more recent category of algorithms is of interest. They work in objective space (the space of objective function vectors) of the MOLP, rather than in the space of feasible solutions, as simplex algorithms do. The outer approximation algorithm of Benson (1998) is one of these algorithms which finds all (weakly) nondominated extreme points and facets of the feasible set in the objective space of the MOLP. In this paper, we will adapt a dual version of that algorithm originally described in Ehrgott et al. (2012) and further developed by Hamel et al. (2014).

Based on the MOLP formulation of Yu, Wei and Brockett (1996), we derive a novel dual MOLP formulation of DEA in the input-output price space. We propose a specialised variant of the dual outer approximation algorithm for DEA problems. This algorithm finds all efficient extreme points and all hyperplanes defining the efficient frontier of the technology set. In other

words, our algorithm computes simultaneously both an inner (by means of extreme points) as well as an outer (by means of facets) description of the technology set. We show that the algorithm does not require to solve any LPs (or mixed integer programmes as is commonly done by researchers aiming to describe the efficient frontier of the technology set, as discussed above). Its running time is therefore mainly affected by the number and percentage of efficient DMUs. This feature distinguishes it from existing algorithms used to compute the facets of the technology set. In particular, our algorithm can determine the efficient facets of the technology set for many real-world DEA instances in reasonable time and is very fast on randomly generated data sets with few inputs and outputs but many DMUs in which the proportion of efficient DMUs is small.

The paper is organised as follows. In Section 2 for the convenience of the reader, we provide the relevant background on DEA. Section 3 introduces a novel dual MOLP formulation of DEA. In Section 4 we derive our algorithm for solving the MOLP formulation of DEA and prove our main result about the algorithm, namely that it does not require the solution of any optimisation problems. Throughout this section we use a numerical example to demonstrate the steps of the algorithm. In Section 5, we present computational results comparing the time required by the standard DEA approach of solving one LP per DMU, Benson's algorithm to compute the efficient frontier and our new algorithm using publicly available real-world data as well as large-scale artificial data. Our results demonstrate rather strikingly that the new algorithm is very fast except for large scale artifificial data with a large number of efficient DMUs.

2. Preliminaries

Throughout the paper, we use the following notation for the componentwise order of vectors in $\mathbb{R}^p$. Let $\boldsymbol{\eta}^1, \boldsymbol{\eta}^2 \in \mathbb{R}^d$ for some $d > 1$. Then

$$\begin{aligned}
\boldsymbol{\eta}^1 \leqq \boldsymbol{\eta}^2 \quad &\text{if and only if} \quad \eta^1_k \leqq \eta^2_k \text{ for } k = 1, \dots, d, \\
\boldsymbol{\eta}^1 \leq \boldsymbol{\eta}^2 \quad &\text{if and only if} \quad \eta^1_k \leqq \eta^2_k \text{ for } k = 1, \dots, d \text{ and } \boldsymbol{\eta}^1 \neq \boldsymbol{\eta}^2, \\
\boldsymbol{\eta}^1 < \boldsymbol{\eta}^2 \quad &\text{if and only if} \quad \eta^1_k < \eta^2_k \text{ for } k = 1, \dots, d.
\end{aligned}$$

The nonnegative orthant of $\mathbb{R}^d$ is denoted $\mathbb{R}^d_{\geqq}$.

2.1. Data Envelopment Analysis

In this section, we present some basic concepts in DEA that are used throughout this article. For a more complete discussion, readers can refer to Cooper et al. (2007) and Cook and Seiford (2009).

We assume that there are n DMUs. Each DMU j, $j \in \{1, \dots, n\}$, is associated with a vector $(\boldsymbol{x}^j, \boldsymbol{y}^j)$, where $\boldsymbol{x}^j = (x^j_1, \dots, x^j_m) \geqq 0$ is a vector of m observed inputs and $\boldsymbol{y}^j = (y^j_1, \dots, y^j_s) \geqq 0$

is a vector of s observed outputs. Let

$$\Gamma \coloneqq \{(\boldsymbol{x}^1, \boldsymbol{y}^1), \ldots, (\boldsymbol{x}^n, \boldsymbol{y}^n)\}$$

be the set of input-output vectors of all DMUs.

Banker et al. (1984) define the variable returns to scale (VRS) technology set as

$$\mathcal{T}_\Lambda = \left\{ (\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^m \times \mathbb{R}^s : \boldsymbol{x} \geqq \sum_{j=1}^{n} \lambda_j \boldsymbol{x}^j, \boldsymbol{y} \leqq \sum_{j=1}^{n} \lambda_j \boldsymbol{y}^j, \boldsymbol{\lambda} \in \Lambda \right\},$$

where $\Lambda \coloneqq \{\boldsymbol{\lambda} \in \mathbb{R}^n_{\geqq} : \sum_{j=1}^{n} \lambda_j = 1\}$. The efficiency score, θ^*_o, of decision making unit 'o' operating under VRS can be evaluated by the input oriented envelopment form of the BCC model of DEA, introduced by Banker et al. (1984), as follows:

$$\begin{aligned} \theta^*_o = \min_{\boldsymbol{\lambda}, \theta} \theta, & \qquad (1) \\ \text{s.t.} \quad \sum_{j=1}^{n} \lambda_j x^j_i &\leqq \theta x^o_i \text{ for } i = 1, \ldots, m, \\ \sum_{j=1}^{n} \lambda_j y^j_r &\geqq y^o_r \text{ for } r = 1, \ldots, s, \\ \boldsymbol{\lambda} &\in \Lambda. \end{aligned}$$

Model (2), the dual to model (1), is known as the input oriented multiplier form of the VRS model of DEA.

$$\begin{aligned} \theta^*_o = \max_{\boldsymbol{\nu}, \boldsymbol{\mu}, \mu_0} \sum_{r=1}^{s} \mu_r y^o_r + \mu_0, & \qquad (2) \\ \text{s.t.} \quad -\sum_{i=1}^{m} \nu_i x^j_i + \sum_{r=1}^{s} \mu_r y^j_r + \mu_0 &\leqq 0 \text{ for } j = 1, \ldots, n, \\ \sum_{i=1}^{m} \nu_i x^o_i &= 1, \\ \boldsymbol{\nu}, \boldsymbol{\mu} &\geqq 0, \end{aligned}$$

In (2), μ_0 is the unconstrained dual variable to the equality constraint $\sum_{j=1}^{n} \lambda_j = 1$ of the condition $\boldsymbol{\lambda} \in \Lambda$ in (1). In a similar fashion, output oriented envelopment and multiplier forms of the BCC model of DEA can be formulated.

DMU o is called weakly DEA efficient if θ^*_o is equal to 1. It is called DEA efficient if and only if $\theta^*_o = 1$ and all constraints are binding at all optimal solutions of problem (1). A DMU that is not weakly DEA efficient is called DEA inefficient. We also introduce formally the definition of Pareto-Koopmans efficiency (or full efficiency) from the DEA literature, see Cooper et al. (2007).

Definition 1 (Pareto-Koopmans Efficiency). *A DMU o is Pareto-Koopmans efficient if and only if there is no input-output vector $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{T}_\Lambda$ such that $\boldsymbol{x} \leqq \boldsymbol{x}^o$ and $\boldsymbol{y} \geqq \boldsymbol{y}^o$ and $(\boldsymbol{x}, \boldsymbol{y}) \neq (\boldsymbol{x}^o, \boldsymbol{y}^o)$.*

It is well known that Pareto-Koopmans efficiency is equivalent to DEA efficiency, for more information and details see Cooper et al. (2007). With this equivalence, a link between DEA and MOLP can be established, as we shall see in Section 2.2 below Remark 1. In this paper, we apply the concepts of DEA efficiency and Pareto-Koopmans efficiency to all input-output vectors in $\mathcal{T}_\Lambda$ that either correspond to an existing DMU or to any convex combination of the existing DMUs.

The input-output vectors $(\boldsymbol{x}^j, \boldsymbol{y}^j)$ of (weakly) DEA efficient DMUs are located on the boundary of $\mathcal{T}_\Lambda$, whereas the input-output vectors of DEA inefficient DMUs are interior points of $\mathcal{T}_\Lambda$. The set of all efficient input-output vectors of $\mathcal{T}_\Lambda$ is called the efficient frontier of the technology set. Moreover, since by definition $\mathcal{T}_\Lambda$ is convex and any input-output vector in $\mathcal{T}_\Lambda$ that does not correspond to an existing DMU is constructed as a convex combination of some existing DMUs, it is clear that the (Pareto-Koopmans) efficient vertices of $\mathcal{T}_\Lambda$ are the input-output vectors of existing DMUs. Since this observation will be fundamental for our algorithm in Section 4 below, we explicitly state it as Remark 1.

Remark 1. *It follows from the definition of $\mathcal{T}_\Lambda$ that every efficient vertex of $\mathcal{T}_\Lambda$ is the input-output vector $(\boldsymbol{x}^j, \boldsymbol{y}^j)$ of an efficient DMU.*

2.2. DEA and MOLP

By Definition 1, a DMU is efficient if and only if, within $\mathcal{T}_\Lambda$ it is not possible to increase any of its outputs without increasing some of its inputs (or decreasing some other output), or vice versa, if it is not possible to decrease any of its inputs without decreasing some of its outputs (or increasing some other input). In fact, it is possible to formulate DEA simultaneously for all DMUs as an MOLP, as shown by Yu, Wei and Brockett (1996) and Hosseinzadeh Lotfi et al. (2008), who draw on the additive DEA model of Charnes et al. (1985) to identify Pareto-Koopmans efficient points in DEA. The goal of this MOLP is to identify input-output vectors that minimise the input component $\boldsymbol{x}$ while maximising the output component $\boldsymbol{y}$, such that $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{T}_\Lambda$. Hence the MOLP formulation of a variable returns to scale DEA model is

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\lambda}} (x_1, \ldots, x_m, -y_1, \ldots, -y_s)^\top, \qquad (3)\\ \text{s.t.} \quad -\sum_{j=1}^{n} \lambda_j \boldsymbol{x}^j + \boldsymbol{x} &\geqq 0,\\ \sum_{j=1}^{n} \lambda_j \boldsymbol{y}^j - \boldsymbol{y} &\geqq 0,\\ \sum_{j=1}^{n} \lambda_j &= 1,\\ \boldsymbol{\lambda} &\geqq 0. \end{aligned}$$

Using $I_{r\times r}$ to denote the identity matrix of size $r \times r$ and $0_{r\times c}$ respectively $1_{r\times c}$ to denote matrices with all zero respectively one entries of size $r \times c$, we define matrices $X = [\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n] \in$

$\mathbb{R}^{m\times n}$ and $Y = [\boldsymbol{y}^1, \ldots, \boldsymbol{y}^n] \in \mathbb{R}^{s\times n}$ for vectors $(\boldsymbol{x}^j, \boldsymbol{y}^j) \in \Gamma$ and construct matrices C, A and vector b as in (4), i.e.,

$$C = \begin{bmatrix} I_{m\times m} & 0_{m\times s} & 0_{m\times n} \\ 0_{s\times m} & -I_{s\times s} & 0_{s\times n} \end{bmatrix}, A = \begin{bmatrix} I_{m\times m} & 0_{m\times s} & -X \\ 0_{s\times m} & -I_{s\times s} & Y \\ 0_{1\times m} & 0_{1\times s} & 1_{1\times n} \\ 0_{1\times m} & 0_{1\times s} & -1_{1\times n} \\ 0_{n\times m} & 0_{n\times s} & I_{n\times n} \end{bmatrix}, \text{ and } \boldsymbol{b} = \begin{bmatrix} 0_{(m+s)\times 1} \\ 1 \\ -1 \\ 0_{n\times 1} \end{bmatrix}. \tag{4}$$

Denoting $\boldsymbol{\xi} = (\boldsymbol{x}^\top, \boldsymbol{y}^\top, \boldsymbol{\lambda}^\top)^\top$ and defining $p = m + s$ and $q = m + s + n + 2$, this choice of C, A, and b allows problem (3) to be expressed as a standard MOLP (5):

$$\min\{C\boldsymbol{\xi} : \boldsymbol{\xi} \in \mathcal{X}\}. \tag{5}$$

Here, the set of feasible solutions of problem (5) is a is the polyhedron $\mathcal{X} := \{\boldsymbol{\xi} :\in \mathbb{R}^q, A\boldsymbol{\xi} \geqq \boldsymbol{b}\}$. Specifically, C and A are $p \times (m + s + n)$ and $q \times (m + s + n)$ matrices, respectively, and $\boldsymbol{b}$ is a vector in $\mathbb{R}^q$. The image of the feasible set $\mathcal{X}$ under the objective function mapping C is denoted by $\mathcal{Y} := \{\boldsymbol{\eta} = C\boldsymbol{\xi} : \boldsymbol{\xi} \in \mathcal{X}\}$, a polyhedron in $\mathbb{R}^p$.

Definition 2.

1. *A feasible solution $\hat{\boldsymbol{\xi}} \in \mathcal{X}$ is called* (weakly) efficient if there is no other feasible solution $\boldsymbol{\xi} \in \mathcal{X}$ such that $C\boldsymbol{\xi}(<) \leq C\hat{\boldsymbol{\xi}}$. $(\mathcal{X}_{wE})\mathcal{X}_E$ denotes the set of all (weakly) efficient solutions.
2. A point $\hat{\boldsymbol{\eta}} \in \mathcal{Y}$ is called *(weakly) nondominated if there is some $\boldsymbol{\xi} \in \mathcal{X}_{(w)E}$ such that $\hat{\boldsymbol{\eta}} = C\boldsymbol{\xi}$. Thus, the (weakly) nondominated set $\mathcal{Y}_{(w)N}$ is the image of the (weakly) efficient set $\mathcal{X}_{(w)E}$.*

The goal of MOLP (5) is to find all (weakly) nondominated points of $\mathcal{Y}$ and for each $\boldsymbol{\eta} \in \mathcal{Y}_{(w)N}$ one $\boldsymbol{\xi} \in \mathcal{X}_{(w)E}$ such that $C\boldsymbol{\xi} = \boldsymbol{\eta}$. There are several methods in the MOLP literature that compute nondominated points and efficient solutions of an MOLP problem, see Wiecek et al. (2016) and references therein. Benson's outer approximation algorithm (Benson 1998) is a procedure that finds all (weakly) nondominated points of the feasible set $\mathcal{Y}$ in the objective space of an MOLP. The most recent version of that algorithm is described in Hamel et al. (2014) and can obviously be applied to the MOLP formulation of DEA in (3).

By Theorem 1, each (weakly) nondominated point of the outcome set $\mathcal{Y} = C\mathcal{X}$ of problem (3) corresponds to a (weakly) DEA efficient input-output vector in $\mathcal{T}_\Lambda$.

Theorem 1 (Yu, Wei and Brockett (1996)). *Each (weakly) efficient solution of (3) corresponds to a (weakly) DEA efficient input-output vector and vice versa.*

3. The Dual DEA Problem

In this section, we derive a new dual MOLP formulation of DEA and then show the relationship between the primal and the dual MOLP. Furthermore, we derive an algorithm to compute the efficient frontier of $\mathcal{T}_\lambda$.

Henceforth we refer to (5) as the primal problem. According to the geometric duality theory developed by Heyde and Löhne (2008), there is an associated dual problem, which consists in the maximisation of a linear function over the dual feasible set

$$\mathcal{U} := \{(\boldsymbol{\pi}, \boldsymbol{\omega}) \in \mathbb{R}^q \times \mathbb{R}^p : A^\top \boldsymbol{\pi} = C^\top \boldsymbol{\omega},\ \boldsymbol{e}^\top \boldsymbol{\omega} = 1,\ (\boldsymbol{\pi}, \boldsymbol{\omega}) \geqq 0\}.$$

The linear objective function of the dual problem is defined by matrix

$$D := \begin{bmatrix} 0_{p-1\times q} & I_{p-1\times p-1} & 0_{p-1\times 1} \\ \boldsymbol{b}^\top & 0_{1\times q-1} & 0 \end{bmatrix},$$

which maps $\mathcal{U}$ to $\mathcal{V} := D\mathcal{U} \subseteq \mathbb{R}^p$, a polyhedron in $\mathbb{R}^p$. Formally, the dual problem can now be stated as

$$\overset{\mathcal{K}}{\max}\{D(\boldsymbol{\pi}, \boldsymbol{\omega}) : (\boldsymbol{\pi}, \boldsymbol{\omega}) \in \mathcal{U}\}, \tag{6}$$

where we have to define the meaning of the maximisation operator $\overset{\mathcal{K}}{\max}$ through the specification of a (partial) order on $\mathbb{R}^p$.

We recall that (partial) orders can be defined by means of cones (see Ehrgott (2005)). The partial order "$\leqq$" is defined by the cone $\mathbb{R}^p_{\geqq} := \{\boldsymbol{\eta} \in \mathbb{R}^p : \boldsymbol{\eta}_k \geqq 0, k = 1, \ldots, p\}$, which is the nonnegative orthant of $\mathbb{R}^p$ by the definition $\boldsymbol{\eta}^1 \leqq \boldsymbol{\eta}^2$ if and only if $\boldsymbol{\eta}^2 - \boldsymbol{\eta}^1 \in \mathbb{R}^p_{\geqq}$. The order we use in the dual problem (6) is defined by the cone

$$\mathcal{K} := \{\boldsymbol{\vartheta} \in \mathbb{R}^p : \vartheta_1 = \vartheta_2 = \ldots = \vartheta_{p-1} = 0, \vartheta_p \geqq 0\}.$$

Hence, $\boldsymbol{\vartheta}^1 \geqq_\mathcal{K} (>_\mathcal{K})\ \boldsymbol{\vartheta}^2$ if and only if $\vartheta^1_k = \vartheta^2_k$ for $k = 1, \ldots, p-1$ and $\vartheta^1_p \geqq (>)\ \vartheta^2_p$. Hence, vectors $\boldsymbol{\vartheta}^1$ and $\boldsymbol{\vartheta}^2$ can only be compared if their first $p-1$ components coincide and the comparison is decided on the p^{th} component.

Based on the order defined by $\mathcal{K}$, we now have a definition for $\mathcal{K}$-nondominated points in the dual objective space $\mathbb{R}^p$.

Definition 3. *A point $\hat{\boldsymbol{\vartheta}} \in \mathcal{V}$ is $\mathcal{K}$-nondominated if there is no other $\boldsymbol{\vartheta} \in \mathcal{V}$ such that $\boldsymbol{\vartheta} >_\mathcal{K} \hat{\boldsymbol{\vartheta}}$. Any $(\boldsymbol{\pi}, \boldsymbol{\omega}) \in \mathcal{U}$ such that $D(\boldsymbol{\pi}, \boldsymbol{\omega})$ is $\mathcal{K}$-nondominated is called a $\mathcal{K}$-efficient solution of the dual MOLP (6). The sets of $\mathcal{K}$-nondominated points of $\mathcal{V}$ and $\mathcal{K}$-efficient solutions of $\mathcal{U}$ are denoted as $\mathcal{V}_N$ and $\mathcal{U}_E$, respectively.*

In analogy to solving MOLP (5), solving the dual MOLP (6) means finding the set $\mathcal{V}_N$ and for each $\boldsymbol{\vartheta} \in \mathcal{V}_N$ a $\mathcal{K}$-efficient solution $(\boldsymbol{\pi}, \boldsymbol{\omega}) \in \mathcal{U}$.

We now define the extended primal and dual objective sets as $\mathcal{P} = \mathcal{Y} + \mathbb{R}^p_{\leqq}$ and $\mathcal{D} = \mathcal{V} - \mathcal{K}$ which are polyhedral sets in $\mathbb{R}^p$. As pointed out in Ehrgott et al. (2012), $\mathcal{P}_N = \mathcal{Y}_N$ and $\mathcal{V}_N = \mathcal{D}_N$. Both $\mathcal{P}$ and $\mathcal{D}$ have dimension p and therefore contain interior points. Moreover $\mathcal{P}_{wN}$ and $\mathcal{D}_N$ are equal to the boundaries of $\mathcal{P}$ and $\mathcal{D}$, respectively. It also holds that all vertices of $\mathcal{P}$ are nondominated and all vertices of $\mathcal{D}$ are $\mathcal{K}$-nondominated. $\mathcal{P}$ and $\mathcal{D}$ are therefore easier to work with than $\mathcal{Y}$ and $\mathcal{V}$. We also assume that $\mathcal{P}$ is $\mathbb{R}^p$-bounded from below, i.e., that there exists some $\boldsymbol{\eta}^0 \in \mathbb{R}^p$ such that $\boldsymbol{\eta}^0 \leqq \boldsymbol{\eta}$ for all $\boldsymbol{\eta} \in \mathcal{P}$. In the context of the DEA MOLP (3), we can choose $\boldsymbol{\eta}^0 = \boldsymbol{\eta}^I$, the ideal point defined by

$$\eta_i^I := \min\left\{x_i^j : j = 1, \ldots, n\right\} \text{ for } i = 1, \ldots, m, \tag{7}$$

$$\eta_{m+r}^I := \min\left\{-y_r^j : j = 1, \ldots, n\right\} \text{ for } r = 1, \ldots, s. \tag{8}$$

This implies that there also exists $\boldsymbol{\vartheta}^0 \in \mathbb{R}^p$ such that $\boldsymbol{\vartheta}^0 \geqq_{\mathcal{K}} \boldsymbol{\vartheta}$ for all $\boldsymbol{\vartheta} \in \mathcal{D}$.

We can formally derive the dual MOLP formulation of DEA, by substituting C, A and $\boldsymbol{b}$ from (4) into (6). This gives (9).

$$\begin{aligned}
&\underset{\boldsymbol{\pi},\boldsymbol{\omega}}{\overset{\mathcal{K}}{\max}}(\omega_1, \ldots, \omega_{m+s-1}, \pi_{m+s+1} - \pi_{m+s+2}), && && (9)\\
&\text{s.t. } \pi_i && = \omega_i \text{ for } i = 1, \ldots, m+s,\\
-\sum_{i=1}^{m} \pi_i x_i^j + \sum_{r=1}^{s} \pi_{m+r} y_r^j + \pi_{m+s+1} - \pi_{m+s+2} + \pi_{m+s+j} &= 0 \text{ for } j = 1, \ldots, n,\\
\sum_{i=1}^{m+s} \omega_i &= 1,\\
\boldsymbol{\pi} &\geqq 0,\\
\boldsymbol{\omega} &\geqq 0.
\end{aligned}$$

Setting $\pi_0 = \pi_{m+s+1} - \pi_{m+s+2}$ as an unconstrained variable, removing variables π_{m+s+j} for $j = 1, \ldots, n$ and substituting ω_i with π_i for $i = 1, \ldots, m+s$ we obtain a dual MOLP form (10)of DEA . This argument can be summarised as Proposition 1.

Proposition 1. *The dual MOLP of primal MOLP (3) is*

$$\begin{aligned}
&\underset{\boldsymbol{\pi},\pi_0}{\overset{\mathcal{K}}{\max}}(\pi_1,\ldots,\pi_m,\pi_{m+1},\ldots,\pi_{m+s-1},\pi_0), \qquad (10)\\
&s.t. \quad -\sum_{i=1}^{m}\pi_i x_i^j+\sum_{r=1}^{s}\pi_{m+r}y_r^j+\pi_0 \;\leqq\; 0 \text{ for } j=1,\ldots,n,\\
&\qquad\qquad \sum_{i=1}^{m}\pi_i+\sum_{r=1}^{s}\pi_{m+r} \;=\; 1,\\
&\qquad\qquad \pi_i \;\geqq\; 0 \text{ for } i=1,\ldots,m,\\
&\qquad\qquad \pi_{m+r} \;\geqq\; 0 \text{ for } r=1,\ldots,s.
\end{aligned}$$

It is clear that the feasible sets in the outcome space of problems (9) and (10) are the same, so that applying our algorithm to (10) will produce the same results as applying it to (9). From now on, denote by $\mathcal{U}$ the feasible set of problem (10) and by $\mathcal{X}$ the feasible set of problem (3). Hence, as defined in Section 3, the extended primal and dual outcome spaces are defined as $\mathcal{P} \coloneqq C\mathcal{X}+\mathbb{R}^{m+s}_{\leqq}$ and $\mathcal{D} \coloneqq D\mathcal{U}-\mathcal{K}$, respectively. We refer to the illustrative example in Section 4 and Figure 2 for illustrations of $\mathcal{P}$ and $\mathcal{D}$.

The dual MOLP (10) seeks to identify (normalised) nonnegative prices π_i, for $i=1,\ldots m+s$, of inputs and outputs and a profit π_0, such that the sum of priced outputs minus priced inputs is less than or equal to $-\pi_0$ for all existing DMUs, while the vector $(\pi_1,\ldots,\pi_m,\pi_{m+1},\ldots,\pi_{m+s-1},\pi_0)$ is $\mathcal{K}$-nondominated, and, by the definition of $\mathcal{K}$, profit is maximised. We note that (10) does not refer to a specific DMU and hence considers all DMUs simultaneously, like MOLP (3) does. Note that the variable π_0 is unrestricted in LP (10), so that negative profit (loss) is possible.

4. Developing the Algorithm

Heyde and Löhne (2008) establish relationships between the primal and dual MOLP problems (5) and (6) and the corresponding polyhedral sets $\mathcal{P}$ and $\mathcal{D}$ in the respective outcome spaces. We summarise the main results of geometric duality as stated in Heyde and Löhne (2008) and Ehrgott et al. (2012) for the primal and dual MOLP formulations of DEA (3) and (10) in Theorem 2.

Let $\boldsymbol{\vartheta} \in \mathcal{D}$ and $\boldsymbol{\eta} \in \mathcal{P}$ and define

$$\boldsymbol{\omega}(\boldsymbol{\vartheta}) \coloneqq \left(\vartheta_1,\ldots,\vartheta_{p-1},1-\sum_{k=1}^{p-1}\vartheta_k\right), \qquad (11)$$

$$\hat{\boldsymbol{\omega}}(\boldsymbol{\eta}) \coloneqq (\eta_1-\eta_p,\ldots,\eta_{p-1}-\eta_p,-1), \qquad (12)$$

and let $H(\boldsymbol{\vartheta})$ and $\hat{H}(\boldsymbol{\eta})$ be hyperplanes in $\mathbb{R}^p$, defined for $\boldsymbol{\vartheta},\boldsymbol{\eta} \in \mathbb{R}^p$, as follows:

$$H(\boldsymbol{\vartheta}) \;\coloneqq\; \{\boldsymbol{\eta} : \boldsymbol{\omega}(\boldsymbol{\vartheta})^\top\boldsymbol{\eta}=\vartheta_p\}, \qquad (13)$$

$$\hat{H}(\boldsymbol{\eta}) \;\coloneqq\; \{\boldsymbol{\vartheta} : \hat{\boldsymbol{\omega}}(\boldsymbol{\eta})^\top\boldsymbol{\vartheta}=-\eta_p\}. \qquad (14)$$

Theorem 2. *The following statements hold.*

- *Point $\boldsymbol{\vartheta} \in \mathcal{D}$ is a $\mathcal{K}$-nondominated vertex of $\mathcal{D}$ if and only if $H(\boldsymbol{\vartheta}) \cap \mathcal{P}$ is a weakly nondominated facet of $\mathcal{P}$.*
- *Point $\boldsymbol{\eta} \in \mathcal{P}$ is a nondominated vertex of $\mathcal{P}$ if and only if $\hat{H}(\boldsymbol{\eta}) \cap \mathcal{D}$ is a $\mathcal{K}$-nondominated facet of $\mathcal{D}$.*

Theorem 2 states that there are one-to-one relationships between vertices of $\mathcal{D}$ and facets of $\mathcal{P}$ as well as between the vertices of $\mathcal{P}$ and the facets of $\mathcal{D}$.

Example 1. *Consider a DEA example with $n = 1,000$ DMUs with a single input and a single output where points are sampled from a bivariate normal distribution. The PPS technology set $\mathcal{T}_\Lambda$ is shown in Figure 1.*

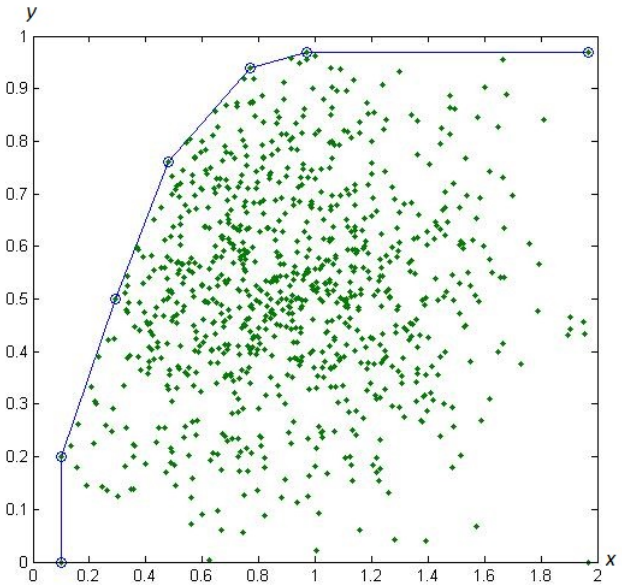


Figure 1: ~~PPS~~ Technology set $\mathcal{T}_\Lambda$ for the illustrative example.

Figure 2 demonstrates the extended feasible sets $\mathcal{P}$ and $\mathcal{D}$ of Example 1.

The five vertices of $\mathcal{P}$ are listed in Table 1. These points correspond to the five efficient DMUs, but with the negatives of the output values. According to (14) and Theorem 2, a vertex $\boldsymbol{\eta}$ of $\mathcal{P}$ corresponds to a facet $\{(\vartheta_1, \vartheta_2) : (\eta_1 - \eta_2)\vartheta_1 - \vartheta_2 + \vartheta_1 = 0\}$ of $\mathcal{D}$ as listed in Table 4.

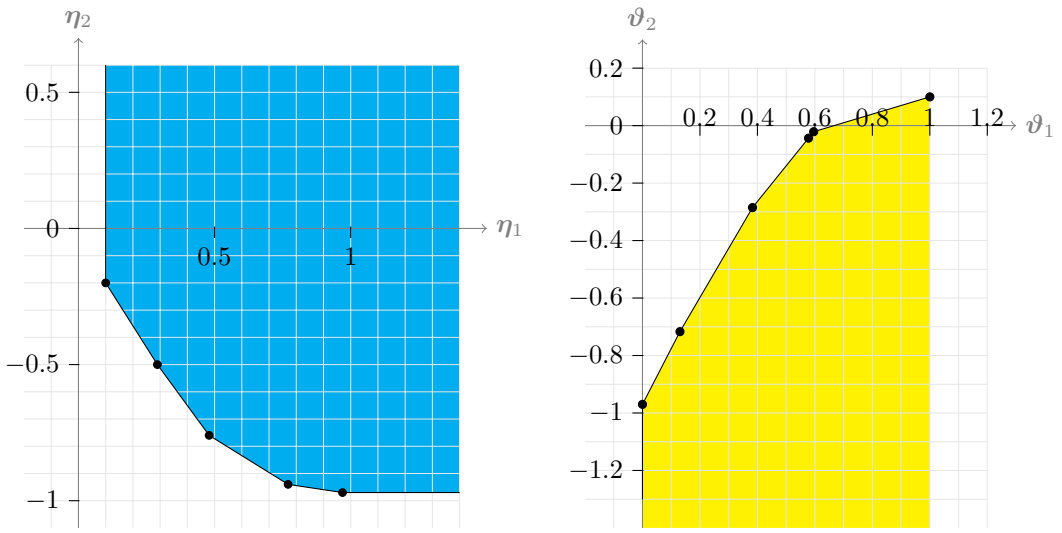
Table 1: The vertices of $\mathcal{P}$.

η^1	η^2	η^3	η^4	η^5
0.480	0.770	0.970	0.100	0.290
-0.760	-0.940	-0.970	-0.200	-0.500

Table 2: The vertices of $\mathcal{D}$.

ϑ^1	ϑ^2	ϑ^3	ϑ^4	ϑ^5	ϑ^6
1.000	0.000	0.383	0.612	0.130	0.578
0.100	-0.970	-0.285	-0.016	-0.717	-0.044

The vertices of $\mathcal{D}$ are shown in Table 2. According to (13) and Theorem 2 a vertex $\boldsymbol{\vartheta}$ of $\mathcal{D}$ corresponds to a facet $\{(\eta_1, \eta_2) : \vartheta_1\eta_1 + (1 - \vartheta_1)\eta_2 - \vartheta_2 = 0\}$ of $\mathcal{P}$ as listed in Table 3.



(a) The set $\mathcal{P}$ obtained by geometric duality.

(b) The set $\mathcal{D}$ obtained by Algorithm 1.

Figure 2: The extended feasible sets $\mathcal{P}$ and $\mathcal{D}$ in primal and dual objective space for the illustrative example.

Table 3: The facets of the $\mathcal{P}$.

$\eta_1 - 0.100$	=	0
$\eta_2 + 0.970$	=	0
$0.383\eta_1 + 0.617\eta_2 + 0.285$	=	0
$0.612\eta_1 + 0.388\eta_2 + 0.016$	=	0
$0.578\eta_1 + 0.422\eta_2 + 0.044$	=	0
$0.130\eta_1 + 0.870\eta_2 + 0.717$	=	0

Table 4: The facets of $\mathcal{D}$.

$1.240\vartheta_1 - \vartheta_2 - 0.760$	=	0
$1.710\vartheta_1 - \vartheta_2 - 0.940$	=	0
$1.940\vartheta_1 - \vartheta_2 - 0.970$	=	0
$0.300\vartheta_1 - \vartheta_2 - 0.200$	=	0
$0.790\vartheta_1 - \vartheta_2 - 0.500$	=	0

We now develop our algorithm to compute the efficient frontier of $\mathcal{T}_\Lambda$ adapting the dual algorithm as described in Hamel et al. (2014) to solve (10) and applying geometric duality as in Theorem 2 to obtain the efficient extreme points and efficient facets of $\mathcal{T}_\Lambda$. The adaptation shows that significant simplifications of the dual Benson algorithm are possible for DEA problems and that, in particular, the algorithm does not require solving any LPs.

Ehrgott et al. (2012) introduce the dual pair of LPs (15),

$$\begin{aligned} &\text{Problem } P(\boldsymbol{\vartheta}): \quad \min\{\boldsymbol{\omega}(\boldsymbol{\vartheta})^\top C\boldsymbol{\xi} : \boldsymbol{\xi} \in \mathcal{X}\}, \\ &\text{Problem } D(\boldsymbol{\vartheta}): \quad \max\{\boldsymbol{b}^\top \boldsymbol{\pi} : \boldsymbol{\pi} \geqq 0, A^\top \boldsymbol{\pi} = C^\top \boldsymbol{\omega}(\boldsymbol{\vartheta})\}. \end{aligned} \tag{15}$$

$P(\boldsymbol{\vartheta})$ has the same feasible set as MOLP (5), but minimises a weighted sum of its objectives, where the weight vector $\boldsymbol{\omega}$ is defined as $\boldsymbol{\omega}(\boldsymbol{\vartheta})$ by a point $\boldsymbol{\vartheta}$ in $\mathcal{D}$ as defined in (11). An optimal solution $\boldsymbol{\xi}^*$ defines a supporting hyperplane $\{\boldsymbol{\vartheta} \in \mathbb{R}^p : \boldsymbol{\omega}(\boldsymbol{\vartheta})^\top (C\boldsymbol{\xi}^*) = \vartheta_p\}$ to $\mathcal{D}$ at boundary point $(\boldsymbol{\omega}(\boldsymbol{\vartheta}), \boldsymbol{\omega}(\boldsymbol{\vartheta})^\top (C\boldsymbol{\xi}^*))$.

The idea of the dual Benson algorithm is to construct a sequence of polyhedra $\mathcal{S}^k, k = 0, 1, \ldots$ containing $\mathcal{D}$ by adding supporting hyperplanes of $\mathcal{D}$ to their description until, after a finite number of iterations, $\mathcal{S}^{k-1} = \mathcal{D}$ when the algorithm terminates with both an inequality and a vertex description of $\mathcal{D}$. At this stage the dual problem is solved, because $\mathcal{V}_\mathcal{N} = \mathcal{D}_N$.

The algorithm first constructs a polyhedron $\mathcal{S}^0 := \{\boldsymbol{\vartheta} \in \mathbb{R}^{m+s} : \boldsymbol{\omega}(\boldsymbol{\vartheta}) \geqq 0, \boldsymbol{\omega}(\boldsymbol{\vartheta})^\top (C\boldsymbol{\xi}^o) - \vartheta_{m+s} \geqq 0\}$ such that $\mathcal{D} \subset \mathcal{S}^0$. Here $\boldsymbol{\xi}^o$ is an optimal solution of $P(\hat{\boldsymbol{\vartheta}})$ for interior point $\hat{\boldsymbol{\vartheta}}$ of $\mathcal{D}$. In every iteration, the dual algorithm chooses a vertex $\boldsymbol{\vartheta}^k$ of $\mathcal{S}^{k-1}$ not contained in $\mathcal{D}$. If $\boldsymbol{\omega}(\boldsymbol{\vartheta}^k)^\top C\boldsymbol{\xi} < \vartheta^k_{m+s}$ an optimal solution $\boldsymbol{\eta}^*$ of $P(\boldsymbol{\vartheta}^k))$ together with $H(\boldsymbol{\eta}^*)$ defines a supporting hyperplane of $\mathcal{D}$. If, on the other hand, $\boldsymbol{\omega}(\boldsymbol{\vartheta}^k)^\top C\boldsymbol{\xi} = \vartheta^k_{m+s}$ then $\boldsymbol{\vartheta}^k \in \mathcal{D}_N$ and another vertex of $\mathcal{S}^{k-1}$ needs to be chosen. $\mathcal{S}^k$ is defined by intersecting $\mathcal{S}^{k-1}$ with the halfspace containing $\mathcal{D}$ until, at termination, $\mathcal{S}^{k-1} = \mathcal{D}$. Detailed descriptions of the general steps of the dual Benson algorithm can be found in Ehrgott et al. (2012) and Hamel et al. (2014).

4.1. The Dual DEA Algorithm

In this section, we provide the specification of those steps taking into account the structure of the dual MOLP (3). Specifically, these are the determination of the initial interior point $\hat{\boldsymbol{\vartheta}}$ of $\mathcal{D}$ and the solution of $P(\boldsymbol{\vartheta})$. Then we provide and explain pseudocode for the algorithm and show the steps of the algorithm on Example 1. Finally, we discuss the complexity of the algorithm.

To find an interior point of $\mathcal{D}$ we have Lemma **??**, the proof of which is straightforward.

Lemma 1. *lem:intP An interior point of $\mathcal{D}$ is available as*

$$\hat{\boldsymbol{\vartheta}} := \left(\frac{1}{m+s}, \ldots, \frac{1}{m+s}, min\left\{-y_r^j : r = 1, \ldots, s; j = 1, \ldots, n\right\} \right). \tag{16}$$

To determine the initial polyhedron $\mathcal{S}^0$, as well as to find a supporting hyperplane of $\mathcal{D}$ during the iterations, it is necessary to solve LPs of the form $P(\boldsymbol{\vartheta})$ in (15). Substituting C, A and b from (4) in $P(\boldsymbol{\vartheta})$ yields (17)

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\lambda}} \sum_{i=1}^{m} \vartheta_i x_i - \sum_{r=1}^{s-1} \vartheta_{m+r} y_r - \left(1 - \sum_{k=1}^{m+s-1} \vartheta_k\right) y_s, & \qquad (17) \\ \text{s.t.} \quad -\sum_{j=1}^{n} \lambda_j \boldsymbol{x}^j + \boldsymbol{x} &\geqq 0, \\ \sum_{j=1}^{n} \lambda_j \boldsymbol{y}^j - \boldsymbol{y} &\geqq 0, \\ \sum_{j=1}^{n} \lambda_j &= 1, \\ \boldsymbol{\lambda} &\geqq 0. \end{aligned}$$

LP (17) is feasible for any $\boldsymbol{\vartheta}$ such that $\vartheta_k \geqq 0, k = 1, \ldots, m+s-1$ satisfying $\sum_{k=1}^{m+s-1} \vartheta_k \leqq 1$ and has a finite optimal objective value. In fact, the objective function of (17) is a weighted sum of the vector valued objective of (3), where the weight vector $(\vartheta_1, \ldots, \vartheta_{m+s})$ is calculated from $\boldsymbol{\vartheta} \in \mathbb{R}^{m+s-1}$ so that $\sum_{k=1}^{m+s-1} \vartheta_k \leqq 1$. By the theory of multi-objective linear programming (Isermann 1974), any optimal solution $\boldsymbol{\xi}^* = (\boldsymbol{x^*}, \boldsymbol{y^*}, \boldsymbol{\lambda^*})$ of (17) is a weakly efficient solution of MOLP (3). In Theorem 3, we exploit the fact that any LP has an optimal solution at an extreme point of its feasible set, and that according to Remark 1 the nondominated extreme points of $\mathcal{P}$ are elements of Γ.

Theorem 3. *Let $\boldsymbol{\vartheta} \in \mathbb{R}^{m+s-1}$ such that $\vartheta_k \geqq 0, k = 1, \ldots, m+s-1$ and $\sum_{i=1}^{m+s-1} \vartheta_i \leqq 1$. Define*

$$\pi(j) \;\coloneqq\; \sum_{i=1}^{m} \vartheta_i x_i^j - \sum_{r=1}^{s-1} \vartheta_{m+r} y_r^j - \left(1 - \sum_{i=1}^{m+s-1} \vartheta_i\right) y_s^j \text{ for } j = 1, \ldots, n, \tag{18}$$

$$j^*(\boldsymbol{\vartheta}) \;\coloneqq\; \arg\min\{\pi(j) : j = 1, \ldots, n\} \text{ and} \tag{19}$$

$$\pi_0^*(\boldsymbol{\vartheta}) \;\coloneqq\; \pi(j^*(\boldsymbol{\vartheta})). \tag{20}$$

Then vector $(\boldsymbol{x}^{j^*(\boldsymbol{\vartheta})}, \boldsymbol{y}^{j^*(\boldsymbol{\vartheta})}, \boldsymbol{e}^{j^*(\boldsymbol{\vartheta})})$ is an optimal solution to $P_2(\boldsymbol{\vartheta})$ in (17), where $(\boldsymbol{x}^{j^*(\boldsymbol{\vartheta})}, \boldsymbol{y}^{j^*(\boldsymbol{\vartheta})}) \in \Gamma$. Moreover, the optimal values of (17) and its dual are equal to $\pi_0^*(\boldsymbol{\vartheta})$.

Proof. The dual $D(\boldsymbol{\vartheta})$ of LP $P(\boldsymbol{\vartheta})$ in the form of (17) can be written as shown in (21).

$$\begin{aligned}
&\max_{\boldsymbol{\pi}, \pi_0} \pi_0, && && \\
&\text{s.t. } \pi_i &&= \vartheta_i \text{ for } i = 1, \ldots, m+s-1, \\
&\pi_{m+s} &&= 1 - \sum_{i=1}^{m+s-1} \vartheta_i \\
-\sum_{i=1}^{m} \pi_i x_i^j + \sum_{r=1}^{s} \pi_{m+r} y_r^j + \pi_0 &&\leqq& \;0 \text{ for } j = 1, \ldots, n, \\
\pi_i &&\geqq& \;0 \text{ for } i = 1, \ldots, m, \\
\pi_{m+r} &&\geqq& \;0 \text{ for } r = 1, \ldots, s.
\end{aligned} \tag{21}$$

The first $m+s$ components of $\boldsymbol{\pi}$, fixed by the first two sets of constraints, are readily eliminated, leaving the simplified LP

$$\begin{aligned}
&\max_{\pi_0} \pi_0, \\
&\text{s.t. } \pi_0 \;\leqq\; \sum_{i=1}^{m} \vartheta_i x_i^j - \sum_{r=m+1}^{s-1} \vartheta_r y_r^j - \left(1 - \sum_{i=1}^{m+s-1} \vartheta_i\right) y_s^j \text{ for } j = 1, \ldots, n.
\end{aligned}$$

Since $\boldsymbol{x}^j$ and $\boldsymbol{y}^j$ are known data and $\boldsymbol{\vartheta}$ is given, the optimal value of π_0 is equal to $\pi_0^*(\boldsymbol{\vartheta})$ in (20).

According to its definition, $\pi_0^*(\boldsymbol{\vartheta})$ is obtained at $j^*(\boldsymbol{\vartheta})$. Thus, the optimal value of problem (17) is equal to $\pi_0^*(\boldsymbol{\vartheta})$, by linear programming duality. On the other hand, $(\boldsymbol{x}^{j^*(\boldsymbol{\vartheta})}, \boldsymbol{y}^{j^*(\boldsymbol{\vartheta})}) \in \Gamma$

so that it is easy to check that $(\boldsymbol{x}^{j^*(\boldsymbol{\vartheta})}, \boldsymbol{y}^{j^*(\boldsymbol{\vartheta})}, \boldsymbol{e}^{j^*(\boldsymbol{\vartheta})})$ is a feasible solution of LP (17) and the objective value of $P(\boldsymbol{\vartheta})$ for this solution is equal to $\pi_0^*(\boldsymbol{\vartheta})$. Hence $(\boldsymbol{x}^{j^*(\boldsymbol{\vartheta})}, \boldsymbol{y}^{j^*(\boldsymbol{\vartheta})}, \boldsymbol{e}^{j^*(\boldsymbol{\vartheta})})$ is an optimal solution to problem (17) and the proof is complete. ■

As a consequence of Theorem 3, it is therefore not necessary to solve any LP in order to solve MOLP (10) (and thus (3)) by our algorithm, which is stated in Algorithm 1.

Algorithm 1 The Dual DEA Algorithm

Input: DEA data Γ

1: Set $\hat{\boldsymbol{\vartheta}}$ as in (16)

2: Find $j^*(\hat{\boldsymbol{\vartheta}})$ as defined in (19)

3: Set

$$\mathcal{S}^0 := \left\{ \boldsymbol{\vartheta} \in \mathbb{R}^{m+s} : \sum_{i=1}^{m} x_i^{j^*(\hat{\boldsymbol{\vartheta}})} \vartheta_i - \sum_{r=1}^{s-1} y_r^{j^*(\hat{\boldsymbol{\vartheta}})} \vartheta_{m+r} - \left(1 - \sum_{i=1}^{m+s-1} \vartheta_i\right) y_s^{j^*(\hat{\boldsymbol{\vartheta}})} - \vartheta_{m+s} \geqq 0, \right.$$
$$\left. \sum_{i=1}^{m+s-1} \vartheta_i \leqq 1, \vartheta_i \geqq 0 \text{ for } i = 1, \ldots, m+s-1 \right\},$$

find $\operatorname{vert} \mathcal{S}^0$ by intersecting the halflines $\{te^g : t \geqq 0\}$ for $g = 1, \ldots m+s-1$ with the the hyperplane defining $\mathcal{S}^0$ and set $k := 1$

4: **while** There is a vertex $\boldsymbol{\vartheta}^k \in \mathcal{S}^{k-1}$ such that $\vartheta_{m+s}^k > \boldsymbol{\pi}_0^*(\boldsymbol{\vartheta}^k)$ **do**

5: Find $j^*(\boldsymbol{\vartheta}^k)$ as defined in (19)

6: Set $(x^k, y^k) := (x^{j^*(\boldsymbol{\vartheta}^k)}, y^{j^*(\boldsymbol{\vartheta}^k)})$

7:

$$\text{Set } \mathcal{S}^k := \mathcal{S}^{k-1} \cap \left\{ \boldsymbol{\vartheta} \in \mathbb{R}^{m+s} : \sum_{i=1}^{m} \vartheta_i x_i^k - \sum_{r=m+1}^{m+s-1} \vartheta_r y_r^k - \left(1 - \sum_{i=1}^{m+s-1} \vartheta_i\right) y_s^k - \vartheta_{m+s} \geqq 0 \right\}$$

8: Update $\operatorname{vert} \mathcal{S}^k$ using the on-line vertex enumeration algorithm of Chen et al. (1991)

9: Set $k := k + 1$

10: **end while**

Output: Set of vertices and set of facets of $\mathcal{D}$. Set of efficient DMUs and set of facet defining hyperplanes of $\mathcal{T}_\Lambda$ using Theorem 2.

Step 1 finds an interior point $\hat{\boldsymbol{\vartheta}}$ of $\mathcal{D}$. Step 2 solves $P(\hat{\boldsymbol{\vartheta}})$ and Step 3 uses this information to construct the initial polyhedron $\mathcal{S}^0$ and computes its vertices. Due to the definition of $\mathcal{D}$, the vertices of S^0 are defined by intersecting the halflines $\{te^g : t \geqq 0\}$ for $g = 1, \ldots, m+s-1$ with the hyperplane defining S^0. Therefore, $\operatorname{vert} \mathcal{S}^0$ is readily available. The main loop of the algorithm is the while loop from lines 4 to 10. In each iteration k, if a vertex $\boldsymbol{\vartheta}^k$ of $\mathcal{S}^{k-1}$ not contained in $\mathcal{D}$ exists, the problem $P(\boldsymbol{\vartheta}^k)$ is solved making use of Theorem 3 (Lines 5 and 6). This defines a

hyperplane, which is used to update $\mathcal{S}^{k-1}$ to $\mathcal{S}^k$ (Step 7). Finally, the vertex set of $\mathcal{S}^k$ is updated using the on-line vertex enumeration algorithm of Chen et al. (1991). It takes the vertices of the polyhedron $\mathcal{S}^{k-1}$ and the hyperplane as inputs and computes the extreme points of the polyhedron $\mathcal{S}^k$ resulting from intersecting the original polyhedron $\mathcal{S}^{k-1}$ with one of the halfspaces defined by the hyperplane. The while loop ends once all vertices of $\mathcal{S}^{k-1}$ belong to $\mathcal{D}$. The facet defining hyperplanes of $\mathcal{T}_\Lambda$ are then known using the geometric duality result of Theorem 2.

We now show in detail the application of Algorithm 1 to Example 1.

Example 2. *An interior point of* $\mathcal{D}$ *is* $\hat{\boldsymbol{\vartheta}} = (0.5, -1.97)^\top$. *LP* $P(\hat{\boldsymbol{\vartheta}})$ *is solved by finding DMU* $j^*(\boldsymbol{\vartheta})$ *according to* (19) *and its input-output vector* $(x^{j^*(\boldsymbol{\vartheta})}, y^{j^*(\boldsymbol{\vartheta})})^\top = (0.48, 0.76)^\top$. *Therefore* $\mathcal{S}^0$ *is initialised as* $\mathcal{S}^0 \coloneqq \{(\vartheta_1, \vartheta_2) : 1.24\vartheta_1 - \vartheta_2 - 0.76 \geqq 0, 0 \leqq \vartheta_1 \leqq 1\}$ *and* $\operatorname{vert} \mathcal{S}^0 = \{(1, 0.48), (0, -0.76)\}$, *which are illustrated on the left side of Figure 3.*

Iteration 1: *Vertex* $\boldsymbol{\vartheta}^1 = (0, -0.76)^\top$ *is selected. The LP* $P(\boldsymbol{\vartheta}^1)$ *is solved by finding* $j^*(\boldsymbol{\vartheta}^1)$ *and* $(x^{j^*(\boldsymbol{\vartheta}^1)}, y^{j^*(\boldsymbol{\vartheta}^1)}) = (0.97, 0.97)$ *which defines* $\mathcal{S}^1 = \mathcal{S}^0 \cap \{(\vartheta_1, \vartheta_2) : 1.94\vartheta_1 - \vartheta_2 - 0.97 \geqq 0\}$ *with vertex set* $\operatorname{vert} \mathcal{S}^1 = \{(1, 0.48), (0.3, 0.388), (0, -0.97)\}$. *The set* $\mathcal{S}^1$ *is shown on the right side of Figure 3.*

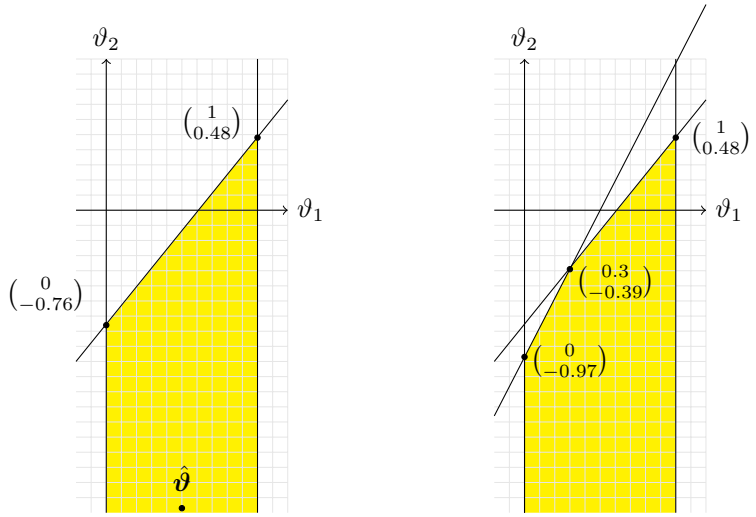


Figure 3: The initial polyhedron $\mathcal{S}^0$ and the first iteration of the dual algorithm.

Iteration 2: *Vertex* $\boldsymbol{\vartheta}^2 = (0, -0.97)$ *is chosen for processing. Since* $\boldsymbol{\vartheta}^2$ *is a vertex of* $\mathcal{D}$, $\boldsymbol{\vartheta}^2 = (0.3, -0.388)$ *is chosen. Because it is not a vertex of* $\mathcal{D}$ *we continue by solving* $P(\boldsymbol{\vartheta}^2)$. *To do that we find* $j^*(\boldsymbol{\vartheta}^2) =$ *and* $(x^{j^*(\boldsymbol{\vartheta}^2)}, y^{j^*(\boldsymbol{\vartheta}^2)}) = (0.77, 0.94)$ *which leads to the update* $\mathcal{S}^2 \coloneqq \mathcal{S}^1 \cap \{(\vartheta_1, \vartheta_2) : 1.71\vartheta_1 - \vartheta_2 - 0.94 \geqq 0\}$ *with vertex set* $\operatorname{vert} \mathcal{S}^2 = \{(1, 0.48), (0.383, -0.285), (0.1304, -0.717), (0, -0.97)\}$. *The set* $\mathcal{S}^2$ *is shown on the left of Figure 4.*

Iteration 3: *We first choose* $\boldsymbol{\vartheta}^3 = (0.1304, -0.717)^\top$*, which turns out to be a vertex of* $\mathcal{D}$*. Next,* $\boldsymbol{\vartheta}^3 = (0.383, -0.2851)^\top$ *is selected, which again is a vertex of* $\mathcal{D}$*. Finally, the third vertex of* $\mathcal{S}^2$*,* $\boldsymbol{\vartheta}^3 = (1, 0.48)^\top$*, is not a vertex of* $\mathcal{D}$*. Hence, we proceed with solving LP* $P(\boldsymbol{\vartheta}^3)$ *by finding* $j^*(\boldsymbol{\vartheta}^3)$ *and* $(x^{j^*(\boldsymbol{\vartheta}^3)}, y^{j^*(\boldsymbol{\vartheta}^3)})^\top = (0.1, 0.2)^\top$*, which yields* $\mathcal{S}^3 := \mathcal{S}^2 \cap \{(\vartheta_1, \vartheta_2) : 0.3\vartheta_1 - \vartheta_2 - 0.2 \geqq 0\}$ *with vertex set* $\operatorname{vert} \mathcal{S}_3 = \{(1, 0.1), (0.5957, -0.02128), (0.383, -0.2851), (0.1304, -0.717), (0, -0.97)\}$*. Set* $\mathcal{S}^3$ *is shown on the right of Figure 4.*

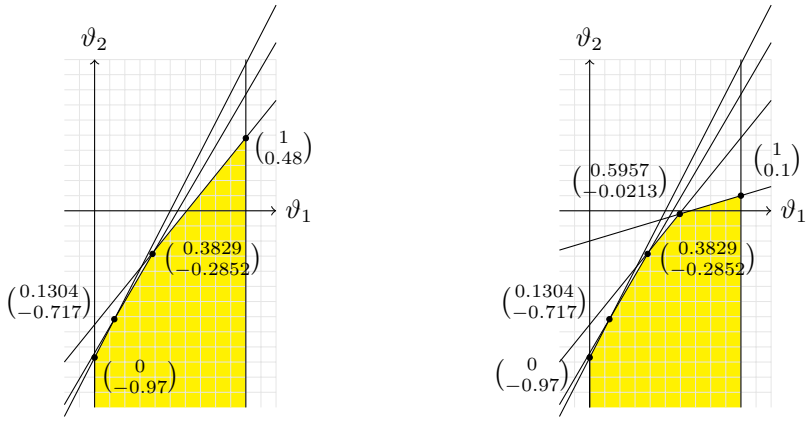


Figure 4: The second and third iterations of the dual algorithm.

Iteration 4: *Vertex* $\boldsymbol{\vartheta}^4 = (0.5957, -0.0213)^\top$ *of* $\mathcal{S}^3$ *is chosen, which is not a vertex of* $\mathcal{D}$*. Hence we continue with solving* $P(\boldsymbol{\vartheta}^4)$ *by finding* $j^*(\boldsymbol{\vartheta}^4)$ *and* $(x^{j^*(\boldsymbol{\vartheta}^4)}, y^{j^*(\boldsymbol{\vartheta}^4)})^\top = (0.29, 0.5)^\top$*, which provides the update* $\mathcal{S}^4 := \mathcal{S}^3 \cap \{(\vartheta_1, \vartheta_2) : 0.79\vartheta_1 - \vartheta_2 - 0.5 \geqq 0\}$ *with the vertex set* $\operatorname{vert} \mathcal{S}^4 = \{(1, 0.1), (0.6122, -0.0163), (0.5778, -0.0436), (0.383, -0.2851), (0.1304, -0.717), (0.0, -0.97)\}$*. Set* $\mathcal{S}^4$ *is shown in Figure 5.*

Iteration 5: *We first test vertex* $\boldsymbol{\vartheta}^5 = (0.5778, -0.04356)$*, which is a vertex* $\mathcal{D}$*. With the second choice of* $\boldsymbol{\vartheta}^5 = (0.6122, -0.01633)$ *we also find that* $\boldsymbol{\vartheta}^5$ *is confirmed as vertex of* $\mathcal{D}$*. The last remaining vertex of* $\mathcal{S}^4$ *that is not already confirmed as a vertex of* $\mathcal{D}$ *is* $\boldsymbol{\vartheta}^5 = (1, 0.1)$*. Since* $t = \vartheta_2^5 - \pi^*(\boldsymbol{\vartheta}^5) = 0$*,* $(1, 0.1)$ *is also a vertex of* $\mathcal{D}$*. Hence the algorithm terminates with* $\mathcal{S}^4 = \mathcal{D}$ *as shown in Figure 5*

The dual algorithm has therefore computed six vertices and five facets of $\mathcal{D}$*. These are shown in Figure 5. Because of Theorem 3 it was not necessary to solve any LP to do this. Exploiting the geometric duality result of Theorem 2, we know that the facets of* $\mathcal{D}$ *correspond to extreme points of* $\mathcal{P}$*, which by Remark 1 are efficient DMUs. Moreover, the extreme points of* $\mathcal{D}$ *correspond to the facets of* $\mathcal{P}$*, and checking formula (13) we can confirm that the hyperplanes defined by the extreme points of* $\mathcal{D}$ *indeed coincide with the facets of* $\mathcal{P}$ *computed by the primal algorithm.*

Finally, let us discuss the worst case complexity of Algorithm 1. As we have demonstrated, it

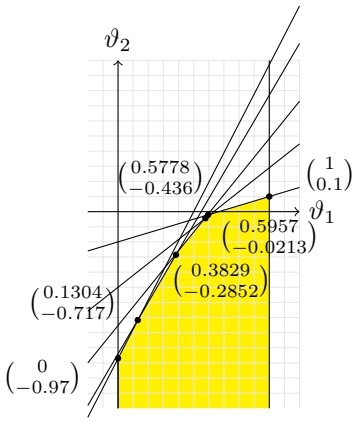


Figure 5: The final polyhedron $\mathcal{S}^4$ obtained in iteration four and confirmed in iteration five of the dual algorithm.

is not necessary to solve any optimization problem during the course of the algorithm. Lines 1 – 3 can be executed in linear time. Hence, the complexity of the algorithm is determined by the number of iterations of the while loop in lines 4 – 10 and the complexity of each iteration. Due to Theorem 2, the number of extreme points of $\mathcal{D}$ is the same as the number of facets of $\mathcal{P}$. According to the upper bound theorem of McMullen (1970), the number of facets of a polyhedron $\mathcal{P}$ in $m+s$ dimensions having (at most) n extreme points can be very large. Hence, our algorithm suffers from the same combinatorial problem from which all other algorithms for finding the efficient frontier of $\mathcal{T}_\Lambda$ suffer: It enumerates a potentially very large (exponential) number of facets. Within each iteration of the while loop, the most time consuming operation is the update of the vertex set. According to Chen et al. (1991), the complexity of the on-line vertex enumeration algorithm for a polyhedron $\mathcal{S}^k$ in dimension $m+s$ is $O((m+s)(|\operatorname{vert}\mathcal{S}^{k-1}|+v)$, where v is the number of new vertices. However, despite the worst case exponential runtime of Algorithm 1, it turns out that in practical applications of DEA the operations of each iteration of the while loop can be carried out very fast, so that the algorithm performs well, as we shall demonstrate in Section 5.

5. Numerical Results

In this section we present numerical results obtained by Algorithm 1 compared with computing the efficient frontier of $\mathcal{T}_\Lambda$ with Benson's algorithm as described in Hamel et al. (2014). For comparison, we also list the time taken to solve one LP per DMU to identify the efficient DMUs. Comparing the runtime of Algorithm 1 with the standard DEA approach gives an indication of its performance for computing the efficient frontier of $\mathcal{T}_\Lambda$ since the initial step for existing algorithms (Olesen and Petersen (2003), Davtalab-Olyaie et al. (2014)) consists of solving DEA LP models to identify extreme efficient DMUs. All computations were carried out on a PC with Intel (R) Core(TM) i3 processor with 4 GB RAM and 1GHz speed under a Windows 8 64 bit operating

system. The algorithms were coded in Matlab R2012a using Gurobi 5.5 as LP solver where needed.

All but the last of the instances in Table 5 are available online at http://www.etm.pdx.edu/dea/dataset/DEAresults.asp?Criteria=Svalue and are taken from real-world DEA application studies listed with their dataset ID in the first column of Table 5. The last instance is from Shermann and Gold (1985). The second column summarises the size of the instance, i.e., number of inputs, outputs and DMUs in the format I-O-D. The following three columns (EDMUs, % Eff and Facets) provide information about the efficient frontier of $\mathcal{T}_\Lambda$, namely the number of efficient DMUs, their percentage among all DMUs and the number of (weakly) efficient facets of $\mathcal{T}_\Lambda$. The next three columns provide the runtimes (in seconds) of the three methods. DEA shows the time taken for solving one LP per DMU, whereas Primal gives the computation time of the primal Benson algorithm. The last column displays the computation time for Algorithm 1.

Table 5: Numerical results for real-world instances.

Dataset ID	I-O-D	EDMUs	% Eff	Facets	DEA	Primal	Dual
1	4 – 2 – 69	29	42.03%	734	3.85	327.68	3.16
17	2 – 3 – 47	9	19.15%	61	2.06	15.12	0.07
50	1 – 4 – 62	13	20.97%	97	3.98	57.14	0.13
88	4 – 3 – 12	12	100.00%	362	1.46	118.54	0.73
99	5 – 3 – 24	16	66.67%	1,172	2.06	586.47	4.02
105	5 – 3 – 96	52	54.17%	6,831	2.95	1,015.34	78.45
108	1 – 4 – 73	23	31.51%	199	2.27	95.45	2.02
51	3 – 7 – 52	39	75.00%	10,826	2.33	1,713.25	149.18
53	5 – 3 – 49	24	48.98%	1,239	2.07	935.33	23.53
94	3 – 2 – 15	10	66.67%	45	0.28	8.57	0.52
89	3 – 1 – 31	6	19.35%	20	0.47	29.35	0.02
128	6 – 6 – 15	11	73.33%	3,248	0.07	483.49	4.91
131	6 – 5 – 23	12	52.17%	2,252	0.05	537.43	2.32
SG	3 – 4 – 14	12	85.71%	285	0.06	228.35	0.22

Table 5 shows that the standard DEA approach is faster than the primal Benson algorithm for all of these small instances. This behaviour is typical for algorithms that provide a description of the facial structure of $\mathcal{T}_\Lambda$ and is probably the main reason why such algorithms are not more widely used. Algorithm 1 on the other hand is faster than DEA on six of the 14 tested instances. We note that the instances for which the primal Benson algorithm and Algorithm 1 take the longest are those for which $\mathcal{T}_\Lambda$ has the largest number of efficient facets. Considering that both algorithms explicitly provide an inequality description of $\mathcal{T}_\Lambda$, which the standard DEA approach does not, it is expected that the computation time is largely determined by the number of efficient facets of $\mathcal{T}_\Lambda$. Since the primal Benson algorithm has to solve a number of LPs proportional to the number of facets, its running time can be very large for instances with a large number of efficient facets. However, and most importantly, Table 5 shows that the computation of extreme points and facets of $\mathcal{T}_\Lambda$ can be completed in a reasonable amount of time by Algorithm 1 for the real world instances of Table 5, less than two and a half minutes at most and about 20 seconds on average.

To investigate the issue of running time and its relationships with problem characteristics further, we next test random instances of larger size. We generated 10 instances each with $100,000$

DMUs and two inputs, one output (Group 1); 10,000 DMUs and two inputs, three outputs (Group 2); and 5,000 DMUs with three inputs, three outputs (Group 3), respectively. The results are summarised in Table 6 in the same format as in Table 5, but with average numbers over the 10 instances of each group.

Table 6: Numerical results for random instances.

Group	I-O-D	EDMUs	% Eff	Facets	DEA	Primal	Dual
1	$2-1-100,000$	24	0.24	47	31,0231.24	195.38	1.88
2	$2-3-10,000$	65	0.65	297	29,758.12	938.24	11.77
3	$3-3-5,000$	118	2.36	1,381	3,626.07	1,562.43	85.63

For these instances, the number of LPs that are solved by the standard DEA approach is much bigger than the number of efficient facets, and hence the primal Benson algorithm beats the standard DEA approach despite the fact that the size of the LPs which Benson's algorithm solves is growing throughout the algorithm. However, it is here where the real merit of Algorithm 1 is revealed. Because the number of efficient DMUs is a small proportion of the number of DMUs and Algorithm 1 does not have to solve any LP to determine the facets of $\mathcal{T}_\Lambda$, it is much much faster than the primal Benson algorithm and the standard DEA approach. As the number of inputs and outputs grows, we expect the proportion of efficient DMUs in random data sets to increase, which is the case here. Clearly, the most important factor influencing computation time for both the primal Benson algorithm and Algorithm 1 for these random data sets is the proportion of efficient DMUs and hence the number of facets.

Finally, to explore the limits of Algorithm 1, we use some of the data sets from Dulá (2014), with a large number of DMUs and a large proportion of efficient DMUs (and therefore as it turns out a large number of facets). These instances have at least 2,500 DMUs. The input-output vectors of efficient DMUs are uniformly distributed on the boundary of a technology set in the nonnegative orthant, as explained in detail in Dulá (2014). They are constructed in such a way that the number of efficient DMUs is a fixed percentage of the total number of DMUs. We again present the results in the same format as in Tables 5 and 6.

Table 7: Numerical results for instances from Dulá (2014).

Instance	I-O-D	EDMUs	% Eff	Facets	DEA	Primal	Dual
1	2-3-2,500	25	1%	313	347.38	310.14	1.52
2	2-3-5,000	50	1%	789	2,085.50	980.56	10.21
3	2-3-7,500	75	1%	1,229	3,654.43	1,959.24	31.10
4	2-3-10,000	100	1%	1,777	4,335.26	2,059.37	75.42
5	4-6-2,500	25	1%	23,124	318.23	7,967.25	265.14
6	2-3-2,500	325	13%	6,839	336.02	953.26	310.33
7	2-3-2,500	625	25%	14,011	320.48	5,465.46	462.35
8	2-3-2,500	1,250	50%	29,984	305.22	9,789.54	875.30

Table 7 contains three types of instances: In the first four instances, the number of DMUs grows, but the proportion of efficient DMUs is small, in fact fixed at 1% of all DMUs. Here,

both Benson's algorithm and Algorithm 1 are faster than solving one LP per DMU. However, the number of facets grows with the number of efficient DMUs and this is the main factor driving the computation time of the primal Benson algorithm. On the other hand, the dual algorithm benefits from the small proportion of efficient DMUs and is much faster (a factor of 70 to almost 100) than both the primal Benson algorithm and the standard DEA approach. One percent of the $2,500$ DMUs of Instance 5 are also efficient, but Instance 5 has a larger number of inputs and outputs compared to Instances 1-4, a fact that considerably increases the number of efficient facets, hence resulting in a computation time of over 2.5 hours for the primal Benson algorithm in Instance 5. Algorithm 1, on the other hand, can still compute more than 20,000 facets in less time than the standard DEA approach solves 2,500 LPs. For Instances 6, 7 and 8 an increasing percentage of 13%, 25% and 50% of their $2,500$ DMUs are efficient. Due to the constant number of DMUs, inputs and outputs of these instances, the DEA runtime is similar, between around 5 and 6 minutes for each instance. However, the instances have an increasing and very large number of facets, which means that the primal Benson algorithm has a very large number of LPs to solve, resulting in very long computation times. Algorithm 1 also experiences long computation times as the number of efficient DMUs increases, however this is less than 15 minutes and less than 3 times the time it takes to solve the 2,500 LPs. We note that although all the computations in the dual algorithm are simple, a very large number of computations are executed. This increase in runtime is, of course, not unexpected, since the problem of computing the facets of a polyhedron is a combinatorial one, and the number of facets of a polyhedron increases rapidly with the number of extreme points.

6. Conclusions and Future Work

In this paper, we have developed a new dual MOLP formulation for DEA. We have derived an algorithm to compute the efficient extreme points and efficient facets of $\mathcal{T}_\Lambda$. This algorithm builds on a dual version of Benson's outer approximation algorithm, but uses the structure of our new dual MOLP formulation in the space of input-output prices to considerably simplify its iterations. Thus the computation of efficiency scores for inefficient DMUs, their reference set and target values is possible in closed form. Most importantly, we have shown that using Algorithm 1, it is possible to compute the efficient frontier of $\mathcal{T}_\Lambda$ without solving any LPs or other optimisation problems. The computational advantage of the dual algorithm versus other algorithms, especially as indicated by comparing its runtimes against the standard DEA approach, which is often a first step in those algorithms, is quite notable. This advantage is most pronounced in DEA instances with large numbers of DMUs, only a few of which are efficient. However, if the percentage of efficient DMUs is large, the polyhedral structure of $\mathcal{T}_\Lambda$ becomes more complicated, and the number of facet defining inequalities grows rapidly and this becomes a limiting factor eventually. Nevertheless, we

note that most real-world applications of DEA are such that Algorithm 1 would run in very reasonable time. Even the most challenging instance with 5 inputs plus outputs and 2,500 DMUs is solved in less than 15 minutes, computing almost 30,000 facets in the process.

We have focused on the BCC model of DEA, but one can replace $\sum_{j=1}^{n} \lambda_j = 1$ with $\sum_{j=1}^{n} \lambda_j \geqq 1$ or $\sum_{j=1}^{n} \lambda_j \leqq 1$ in model (3) and derive the MOLP form of increasing returns to scale (IRS) and decreasing returns to scale (DRS) DEA models (see also Cooper et al. 2007). Theorem 1 is satisfied for these MOLP models, see Yu, Wei and Brockett (1996) for more details. The dual MOLP forms of IRS and DRS models can be obtained, respectively, by adding the constraint $\pi_0 \geqq 0$ and $\pi_0 \leqq 0$ to model (10). Benson's algorithm and Algorithm 1 can then be applied to compute the efficient frontier of the technology set of IRS and DRS models. The main difference is in the initialisation step where the ideal point needs to be identified. Similar considerations can also be made for the constant returns to scale or CCR model of DEA.

In future research, we aim to develop acceleration techniques to speed up the algorithm. In general, our research shows that investigating the structure of DEA problems and the algorithmic performance of algorithms for DEA is worthwhile and that significant gains in computational performance may be made.

References

Aparicio, J. and Pastor, J. T. (2013), 'A well-defined efficiency measure for dealing with closest targets in dea', *Applied Mathematics and Computation* **219**(17), 9142–9154.

Aparicio, J. and Pastor, J. T. (2014), 'Closest targets and strong monotonicity on the strongly efficient frontier in DEA', *Omega* **44**, 51–57.

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

Benson, H. P. (1998), 'An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem', *Journal of Global Optimization* **13**, 1–24.

Bougnol, M. I., Dulá, J. H. and Rouse, P. (2012), 'Interior point methods in DEA to determine non-zero multiplier weights', *Computers & Operations Research* **39**, 698–708.

Briec, W. and Leleu, H. (2003), 'Dual representations of non-parametric technologies and measurement of technical efficiency', *Journal of Productivity Analysis* **20**(1), 71–96.

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

Charnes, A., Cooper, W. W., Golany, B., Seiford, L. and Stutz, J. (1985), 'Foundations of data envelopment analysis for Pareto-Koopmans efficient empirical production functions', *Journal of Econometrics* **30**(1-2), 91–107.

Chen, P. C., Hansen, P. and Jaumard, B. (1991), 'On-line and off-line vertex enumeration by adjacency lists', *Operations Research Letters* **10**, 403–409.

Cook, W. D. and Seiford, L. M. (2009), 'Data envelopment analysis (DEA) thirty years on', *European Journal of Operational Research* **192**, 1–17.

Cooper, W. W., Seiford, L. M. and Tone, K. (2007), *Data Envelopment Analysis: A Comprehensive Text with Models, Applications, References and DEA-solver Software*, Springer, Berlin.

Davtalab-Olyaie, M., Roshdi, I., Jahanshahloo, G. and Asgharian, M. (2014), 'Characterizing and finding full dimensional efficient facets in DEA: A variable returns to scale specification', *Journal of the Operational Research Society* **65**, 1453–1464.

Davtalab-Olyaie, M., Roshdi, I., Partovi Nia, V. and Asgharian, M. (2015), 'On characterizing full dimensional weak facets in DEA with variable returns to scale technology', *Optimization* **64**(11), 2455–2476.

Dulá, J. (2014), 'Large DEA Data', http://www.people.vcu.edu/~jdula/LargeScaleDEAdata/.

Dulá, J. H. (2002), 'Computation in DEA', *Pesquisa Operacional* **22**(2), 165–182.

Dulá, J. H. (2008), 'A computational study of DEA with massive data sets', *Computers & Operations Research* **35**, 1191–1203.

Dulá, J. H. (2011), 'An algorithm for data envelopment analysis', *INFORMS Journal on Computing* **23**, 284–296.

Ehrgott, M. (2005), *Multicriteria Optimization*, Springer, Berlin.

Ehrgott, M., Löhne, A. and Shao, L. (2012), 'A dual variant of Benson's outer approximation algorithm', *Journal of Global Optimization* **52**, 757–778.

Hamel, A. H., Löhne, A. and Rudloff, B. (2014), 'Benson type algorithms for linear vector optimization and applications', *Journal of Global Optimization* **59**, 811–836.

Heyde, F. and Löhne, A. (2008), 'Geometric duality in multiple objective linear programming', *SIAM Journal on Optimization* **19**, 836–845.

Hosseinzadeh Lotfi, F., Noora, A. A., Jahanshahloo, G. R., Jablonsky, J., Mozaffari, M. R. and Gerami, J. (2008), 'An MOLP based procedure for finding efficient units in DEA models', *Central European Journal of Operations Research* **17**, 1–11.

Isermann, H. (1974), 'Proper efficiency and the linear vector maximum problem', *Operations Research* **22**, 189–191.

Jahanshahloo, G. R., Lotfi, F. H. and Akbarian, D. (2010), 'Finding weak defining hyperplanes of PPS of the BCC model', *Applied Mathematical Modelling* **34**(11), 3321–3332.

Jahanshahloo, G. R., Lotfi, F. H., Rezai, H. Z. and Balf, F. R. (2007), 'Finding strong defining hyperplanes of production possibility set', *European Journal of Operational Research* **177**, 42–54.

Jahanshahloo, G. R., Lotfi, F. H. and Zohrehbandian, M. (2005), 'Finding the piecewise linear frontier production function in data envelopment analysis', *Applied Mathematics and Computation* **163**(1), 483–488.

Joro, T., Korhonen, P. and Wallenius, J. (1998), 'Structural comparison of data envelopment analysis and multiple objective linear programming', *Management Science* **44**, 962–970.

McMullen, P. (1970), 'The maximum number of faces of a convex polytope', *Mathematika* **18**, 179–184.

Olesen, O. B. and Petersen, N. C. (2003), 'Identification and use of efficient faces and facets in DEA', *Journal of Productivity Analysis* **20**(3), 323–360.

Shermann, H. D. and Gold, F. (1985), 'Bank branch operating efficiency: Evaluation with data envelopment analysis', *Journal of Banking & Finance* **9**(2), 297–315.

Steuer, R. E. (1985), *Multiple Criteria Optimization: Theory, Computation and Application*, John Wiley & Sons, New York.

Stewart, T. J. (1996), 'Relationships between data envelopment analysis and multicriteria decision analysis', *Journal of the Operational Research Society* **47**, 654–665.

Wiecek, M. M., Ehrgott, M. and Engau, A. (2016), Continuous multiobjective programming, *in* S. Greco, M. Ehrgott and J. R. Figueira, eds, 'Multicriteria Decision Analysis: State of the Art Surveys', Vol. 233 of *International Series in Operations Research & Management Science*, Springer, New York, chapter 18, pp. 739–815.

Yougbaré, J. W. and Teghem, J. (2007), 'Relationships between Pareto optimality in multi-objective 0–1 linear programming and DEA efficiency', *European Journal of Operational Research* **183**, 608–617.

Yu, G., Wei, Q. and Brockett, P. (1996), 'A general data envelopment analysis model: A unification and extension of existing methods for efficiency analysis of decision making units', *Annals of Operations Research* **66**, 47–89.

Yu, G., Wei, Q., Brockett, P. and Zhou, L. (1996), 'Construction of all DEA efficient surfaces of the production possibility set under the general data envelopment analysis model', *European Journal of Operational Research* **95**, 491–510.

Yun, Y. B., Nakayama, H., Tanino, T. and Arakawa, M. (2001), 'Generation of efficient frontiers in multiobjective optimization problems by generalized data envelopment analysis', *European Journal of Operational Research* **129**, 586–595.