

# The vector linear program solver *Bensolve* – notes on theoretical background

Andreas Löhne

*Friedrich Schiller University Jena  
Department of Mathematics  
07737 Jena  
Germany*

Benjamin Weißing\*

*Martin Luther University Halle–Wittenberg  
Department of Mathematics  
06099 Halle (Saale)  
Germany*

---

## Abstract

*Bensolve* is an open source implementation of Benson’s algorithm and its dual variant. Both algorithms compute primal and dual solutions of vector linear programs (VLP), which include the subclass of multiple objective linear programs (MOLP). The recent version of *Bensolve* can treat arbitrary vector linear programs whose upper image does not contain lines. This article surveys the theoretical background of the implementation. In particular, the role of VLP duality for the implementation is pointed out. Some numerical examples are provided. In contrast to the existing literature we consider a less restrictive class of vector linear programs.

*Keywords:* vector linear programming, linear vector optimization, multiple objective optimization

*2010 MSC:* 90C29, 90C05, 52B55, 15A39

---

## 1. Introduction

Let us start with the formulation of a *multiple objective linear program*, which is a special case of a *vector linear program*. Given positive integers  $n, m, q$  and data of the form

$$P \in \mathbb{R}^{q \times n},$$

$$B \in \mathbb{R}^{m \times n}, a \in (\mathbb{R} \cup \{-\infty\})^{m \times 1}, b \in (\mathbb{R} \cup \{+\infty\})^{m \times 1},$$

and

$$l \in (\mathbb{R} \cup \{-\infty\})^{n \times 1}, s \in (\mathbb{R} \cup \{+\infty\})^{n \times 1},$$

---

\*corresponding author

*Email addresses:* [andreas.loehne@uni-jena.de](mailto:andreas.loehne@uni-jena.de) (Andreas Löhne),  
[benjamin.weissing@mathematik.uni-halle.de](mailto:benjamin.weissing@mathematik.uni-halle.de) (Benjamin Weißing)

we consider the problem

$$\min Px \quad \text{subject to} \quad a \leq Bx \leq b, \quad l \leq x \leq s. \quad (\text{MOLP})$$

Minimization is understood with respect to the component-wise ordering in  $\mathbb{R}^q$ .

In the more general setting of a vector linear program, one considers a partial ordering  $\leq_C$  generated by a polyhedral convex ordering cone  $C \subseteq \mathbb{R}^q$  that does not contain lines, or equivalently in this setting, is pointed. For vectors  $y, z \in \mathbb{R}^q$ , we define

$$y \leq_C z \quad : \iff \quad z - y \in C.$$

The polyhedral cone  $C$  can be given either by a matrix  $Y \in \mathbb{R}^{q \times o}$  as

$$C = \{y \in \mathbb{R}^q \mid \exists v \in \mathbb{R}^o : y = Yv, v \geq 0\}, \quad (1)$$

or by a matrix  $Z \in \mathbb{R}^{q \times p}$  as

$$C = \{y \in \mathbb{R}^q \mid Z^T y \geq 0\}. \quad (2)$$

Throughout, (1) is referred to as **cone**-representation and (2) is called **dual cone**-representation. This can be motivated as  $C$  is generated by the columns of the matrix  $Y$  (in the sense of (1)), whereas the dual cone

$$C^* := \{w \in \mathbb{R}^q \mid \forall y \in C : y^T w \geq 0\}$$

of  $C$  is generated by the columns of the matrix  $Z$ . The latter statement is a consequence of Farkas' lemma. The resulting vector linear program is

$$\min_C Px \quad \text{subject to} \quad a \leq Bx \leq b, \quad l \leq x \leq s. \quad (\text{VLP})$$

The recent version of *Bensolve* assumes further that the cone  $C$  is *solid*, i.e., it has dimension  $q$ , or equivalently, nonempty interior. This assumption is equivalent to  $\text{rank } Y = q$ . Further,  $C$  is free of lines if and only if  $\text{rank } Z = q$ . This implies that necessarily  $o \geq q$  and  $p \geq q$ .

We close this section with some bibliographical remarks. It seems that Dauer [1] and Dauer & Liu [2] started to propagate the objective space approach in multiple objective linear programming, saying that on the one hand it is more efficient, and on the other hand it is sufficient in practice to generate only the minimal (or non-dominated) vertices in the objective space (or image space). The algorithms of *Bensolve* are based on the papers by Benson [3, 4] from 1998. Since then, several extensions, simplifications and improvements have been published, among them the introduction of a dual algorithm in [5], the extension to the unbounded case in [6], approximate variants in [7], the extension to pointed solid polyhedral ordering cones in [8], a simplification which requires only one LP per iteration (independently) in [9] and [8], and a complexity analysis in [10]. Recently, it was shown that *Bensolve* can be used to compute projections of polyhedra [11]. In contrast to the literature, see e.g. [8], we make weaker assumptions: The geometric duality parameter vector  $c \in \mathbb{R}^q$  is required to satisfy  $c_q \neq 0$  rather than  $c_q = 1$ . The choice of ordering cones is therefore less restricted. Moreover, we derive geometric dual programs for maximization problems as well as for problems with box constraints.

## 2. Solution concepts

In contrast to most classical textbooks on vector optimization and multiple objective programming, *Bensolve* is based on a recent solution concept, which has been introduced in [12] and [6]. This concept entails the notions of minimality *and* infimum attainment, which are equivalent in the scalar linear programming case, but do diverge with increasing image space dimension. While in the classical literature essentially only minimality is taken into account, we will show in the following that *both* conceptions are essential for a proper solution concept. To this end, we start with the classical scalar theory and extend it to the case of multiple image space dimensions.

Consider the scalar linear program

$$\min x \mapsto cx \quad \text{s.t. } x \in S, \quad (\text{LP})$$

where  $S \subseteq \mathbb{R}^n$  is some feasible set defined by linear inequalities. Finding a *solution* to (LP) means to find a feasible variable  $\bar{x} \in S$  whose image  $c\bar{x}$  equals the minimum of the set of objective values  $\{cx \mid x \in S\} =: c[X]$ .

The generalization of minimality to higher image space dimension with respect to the ordering induced by  $C$  is a standard notion:

**Definition 1.** A point  $y \in A \subseteq \mathbb{R}^q$  is called *C-minimal* in  $A$ , if for every point  $\tilde{y} \in A$  the following implication holds:

$$\tilde{y} \leq_C y \Rightarrow \tilde{y} = y.$$

The set of all  $C$ -minimal points of a set  $A$  is denoted by  $\text{Min}_C A$ .

The generalization of the term 'solution', however, is not so obvious. Likewise to the scalar case, a solution to a problem should be a single entity. But, a single minimizer is *not* an appropriate solution, even though it is referred to as *efficient solution* in the classical literature. The reason is that a single minimizer can be obtained by solving a scalarized problem, which in the present setting is usually a linear program. The problem to determine a single minimizer is therefore a matter of scalar optimization. Thus, the *single entity* mentioned above should be a *set* of minimizers.

The problem to compute *all* extreme minimizers has been investigated, see [13, 14], but for many applications, this approach does not seem to be tractable. This can be seen already by considering the scalar case: The problem to compute all extreme minimizers (i.e. a representation of all solutions) of a linear program is NP hard: Consider the feasibility problem. This leads to vertex enumeration, which is NP hard, see e.g. [15, Proposition 5].

Dauer [1] and many followers aimed at characterizing *every* minimal element in the *objective space*. A corresponding set of minimizers is what we understand to be a *solution* to a vector linear program. Surprisingly this idea can be defined via *infimum attainment*.

Let us consider the scalar case first. An alternative characterization of a solution for (LP), which does not rely on minimality, is provided by means of infimum attainment: The image  $c\bar{x}$  of a solution  $\bar{x}$  to (LP) is the infimum of the image of the feasible set  $c[S]$ , which may be expressed equivalently by

$$\{c\bar{x}\} + \mathbb{R}_+ \subseteq c[S] + \mathbb{R}_+. \quad (3)$$

The generalization of the right hand side of inequality (3) in the case of multiple image space dimensions features prominently in the solution concept for (VLP) and is therefore denoted by a dedicated term:

**Definition 2.** The *upper image* of (VLP) is the Minkowski sum of the image  $P[S]$  of the feasible set  $S$  and the ordering cone  $C$ :

$$\mathcal{P} := P[S] + C. \quad (4)$$

The *lower image*, in contrast, is obtained by adding the negative of the ordering cone to the image:

$$P[S] - C. \quad (5)$$

As a generic term for both notions we use *extended image*.

The upper image can be understood as an infimum, too: It serves as infimum in a complete lattice which embeds the image space  $\mathbb{R}^q$ . As the theoretical details are beyond the scope of the present article, the interested reader is referred to [12, 6] for a thorough discussion. Here, we focus on the practical implications the consideration of the upper image yields. Observe that the right hand side of inequality (3) describes all the information in the image space that matters for a solution to (LP): While the image  $c[S]$  may be a closed interval  $[c\bar{x}, y]$ , the actual value of  $y$  or even the existence of  $y$  is not of interest for a minimization problem. Adding the ordering cone ( $\mathbb{R}_+$  in the case of scalar minimization) to the image  $c[S]$  of the feasible set conveys this fact. Before showing that this feature of the upper image is retained in higher image space dimensions (Proposition 8), we provide the notion of minimal directions in order to be able to describe the upper image geometrically. Hitherto we recapitulate that polyhedra may be represented by different means:

- (i) In terms of finitely many points and directions, called V-representation.
- (ii) As intersection of finitely many affine halfspaces, called H-representation.

**Definition 3.**  $y^h \in \mathbb{R}^q \setminus \{0\}$  is a *C-minimal direction* in  $A \subseteq \mathbb{R}^q$  if for some  $y \in A$  the point  $y + \mu y^h$  is C-minimal in  $A$  for every scalar  $\mu \geq 0$ .

Minimizers now can be defined as feasible elements generating minimal points and minimal directions. We set  $\ker P := \{x \in \mathbb{R}^n \mid Px = 0\}$ .

**Definition 4** (Feasibility).

A point  $\bar{x} \in \mathbb{R}^n$  is called *feasible* for (VLP) if

$$\bar{x} \in S := \{x \in \mathbb{R}^n \mid a \leq Bx \leq b, l \leq x \leq s\}.$$

A direction  $\bar{x}^h \in \mathbb{R}^n \setminus \ker P$  is called *feasible* for (VLP) if

$$\bar{x}^h \in S^h := \{x \in \mathbb{R}^n \mid 0 \cdot a \leq Bx \leq 0 \cdot b, 0 \cdot l \leq x \leq 0 \cdot s\},$$

where we define  $0 \cdot \pm\infty = \pm\infty$ .

**Definition 5** (Minimizer). A feasible point  $x \in S$  is called *minimizer* for (VLP) if its image  $Px$  is C-minimal in  $P[S]$ . Likewise, a direction  $x^h \in S^h \setminus \ker P$  is called *minimizer* for (VLP) if  $Px^h$  is a C-minimal direction in  $P[S]$ .

An infimizer of (VLP) is a pair of sets that generates the upper image. For a set  $B \subseteq \mathbb{R}^q$ ,  $\text{conv } B$  denotes its convex hull. The cone generated by a non-empty set  $B \subseteq \mathbb{R}^q$  is denoted by  $\text{cone } B := \{tx \mid x \in B, t \geq 0\}$ . We set  $\text{cone } \emptyset := \{0\}$ .

**Definition 6** (Infimizer). A pair  $(\bar{S}, \bar{S}^h)$ , where  $\bar{S}$  is a nonempty finite set of feasible points and  $\bar{S}^h$  is a finite set of feasible directions is called a *finite infimizer* of (VLP) if

$$\text{conv } P[\bar{S}] + \text{cone } P[\bar{S}^h] + C \supseteq P[S] + C. \quad (6)$$

We note here that *minimality* is a kind of “local” property, whereas *infimum attainment* characterizes the relevant parts of the image of (VLP) “as a whole.” A solution to (VLP) combines both conceptions:

**Definition 7** (Solution). A finite infimizer  $(\bar{S}, \bar{S}^h)$  is said to be a *solution* of (VLP) if the sets  $\bar{S}$  and  $\bar{S}^h$  consist of minimizers only.

It is imperative, of course, that no minimal point is lost during the transition from the image  $P[S]$  to the upper image  $\mathcal{P}$ . This is ensured by the following proposition:

**Proposition 8.** *Suppose  $\mathcal{P}$  contains a vertex. Then*

- (i)  $\text{Min}_C \mathcal{P} = \text{Min}_C P[S]$ .
- (ii) *Every vertex of  $\mathcal{P}$  is  $C$ -minimal.*
- (iii) *An extreme direction of  $\mathcal{P}$  is  $C$ -minimal if and only if it does not belong to  $C$ .*

*Proof.* The first two statements are well-known and can be proven straightforward. To prove (iii), first note that no  $c \in C \setminus \{0\}$  may serve as a minimal direction, as for any  $y \in \mathcal{P}$  we have

$$y \leq_C y + c \quad \text{and} \quad y \neq y + c.$$

Now let an extremal direction  $\bar{y} \in \mathcal{P}$  be given, that is, there exists  $y \in \mathcal{P}$  such that  $F := \{y + t\bar{y} \mid t \geq 0\}$  is a 1-dimensional face of  $\mathcal{P}$ . Now consider an element  $\tilde{y} \in \mathcal{P}$  with  $\tilde{y} \leq_C y + t\bar{y}$  for given  $t \geq 0$ , which means that there exists  $c \in C$  with  $\tilde{y} + c = y + t\bar{y}$ . For any given  $\lambda \in (0, 1)$  we have  $\tilde{y} + \frac{1}{1-\lambda}c \in \mathcal{P}$ . From  $y + t\bar{y} = \lambda\tilde{y} + (1-\lambda)\left(\tilde{y} + \frac{1}{1-\lambda}c\right) \in F$  we conclude  $\tilde{y} \in F$  and  $\tilde{y} + \frac{1}{1-\lambda}c \in F$ . This means either  $\bar{y}$  is a nonnegative (as  $\mathcal{P}$  does not contain lines) multiple of  $c$  or  $c = 0$ . Therefore, if  $\bar{y} \notin C$ ,  $y + t\bar{y}$  is  $C$ -minimal for every  $t \geq 0$ , proving minimality of  $\bar{y}$ .  $\square$

The preceding proposition shows that for obtaining a solution to (VLP) it is sufficient to find the vertices  $Px$  and minimal extreme directions  $Px^h$  of the upper image as well as preimages  $x \in S, x^h \in S^h$  which generate them. In fact, this is how *Bensolve* works: A vertex representation of the upper image is found along with corresponding preimages for the vertices and minimal directions. Actually, *Bensolve* does even more: The V-representation found by *Bensolve* is *irreducible*, that is, if any element of the solution is left out, it will not generate the upper image  $\mathcal{P}$  in the sense of (6).

Those extremal directions in the V-representation which are not minimal (namely those belonging to  $C$ ) are not a part of the solution, but nevertheless they are necessary for a complete description of the upper image. To emphasize their significance, these directions are labeled specifically:

**Definition 9.** The set of extreme directions of  $\mathcal{P}$  belonging to  $C$  is called *cone compartment*.

### 3. Dual problem and dual solutions

Duality plays an important role for *Bensolve*. For the user of the software the following aspects are important:

- (i) The extended image  $\mathcal{D}$  of the dual problem contains information about the extended image  $\mathcal{P}$  of the primal problem: *Bensolve* outputs a V-representation of  $\mathcal{D}$ , which can be used to obtain an H-representation of  $\mathcal{P}$ . Likewise, an H-representation of  $\mathcal{D}$  can be obtained from the V-representation of  $\mathcal{P}$  computed by *Bensolve*.
- (ii) A dual algorithm can be chosen, which can be advantageous for certain problem instances. The dual algorithm constructs an outer and inner approximation of  $\mathcal{D}$  which corresponds, by duality, to an inner and outer approximation of  $\mathcal{P}$ .
- (iii) A duality parameter vector  $c \in \mathbb{R}^q$  can be chosen by the user. The dual problem and the dual solution (but not the primal problem and primal solution) depend on this vector. It has influence on numerical issues of both the primal and the dual algorithm.

To introduce the reader into the main ideas of duality for vector linear programming (in the sense of “geometric duality” established in [16]), we begin with the following special case of (MOLP):

$$\min Px \quad \text{subject to} \quad a \leq Bx. \quad (7)$$

The dual problem to (7) is the following vector linear program with ordering cone  $K := \{y \in \mathbb{R}^q \mid y_1 = 0, \dots, y_{q-1} = 0, y_q \geq 0\}$ :

$$K\text{-maximize} \quad D(u, w) \quad \text{s.t.} \quad B^T u = P^T w, \quad u \geq 0, \quad w \geq 0, \quad e^T w = 1, \quad (8)$$

where the objective function is defined as

$$D : \mathbb{R}^m \times \mathbb{R}^q \rightarrow \mathbb{R}^q, \quad D(u, w) := (w_1, w_2, \dots, w_{q-1}, a^T u)^T$$

and  $e = (1, \dots, 1)^T$  denotes the all-one vector in  $\mathbb{R}^q$ . In this special case, the duality parameter vector  $c \in \mathbb{R}^q$  has been chosen as  $c = e$ . Later on, the constraint  $e^T w = 1$  will be replaced by  $c^T w = 1$ . The feasible set is denoted by

$$T := \{(u, w) \in \mathbb{R}^m \times \mathbb{R}^q \mid B^T u = P^T w, \quad u \geq 0, \quad w \geq 0, \quad e^T w = 1\}.$$

Duality provides a relationship between the upper image  $\mathcal{P}$  of the primal problem (7) and the lower image  $\mathcal{D}$  of the dual problem (8), defined as

$$\mathcal{D} := D(T) - K.$$

In addition to weak and strong duality, for details see e.g. [6], there is a third type of duality relation, called *geometric duality*. It states that there is a one-to-one correspondence between the proper faces of the polyhedron  $\mathcal{P}$  and the non-vertical (i.e. the last component of the corresponding normal vector does not

vanish) proper faces of the polyhedron  $\mathcal{D}$ . The dimension of a proper face of  $\mathcal{P}$  and the dimension of the corresponding face of  $\mathcal{D}$  add up to  $q - 1$ . In particular, facets ( $(q - 1)$ -dimensional faces) correspond to vertices (0-dimensional faces).

In order to be able to formulate duality results we consider the following bi-affine *coupling function*, which was introduced in [16]:

$$\varphi(y, y^*) := \sum_{i=1}^{q-1} y_i y_i^* + y_q \left( 1 - \sum_{i=1}^{q-1} y_i^* \right) - \xi(y) y_q^*, \quad (9)$$

where

$$\xi(y) = \begin{cases} 1 & \text{if } y \text{ is a point} \\ 0 & \text{if } y \text{ is a direction.} \end{cases}$$

The next theorem is a consequence of the geometric duality theorem [16, Theorem 3.1]. It points out the facts relevant for users of *Bensolve* and does not aim to cover the complete idea of geometric duality. For simplicity, we assume that  $\mathcal{P}$  has a vertex, which corresponds exactly to the setting of the recent version of *Bensolve* [17].

**Theorem 10.** *If  $\mathcal{P}$  has a vertex, then the following statements hold true:*

- (i) *A finite set  $\bar{Y}$  of points and directions in  $\mathbb{R}^q$  is an irredundant V-representation of  $\mathcal{P}$  if and only if*

$$\varphi(y, y^*) \geq 0, \quad y \in \bar{Y}$$

*is an irredundant H-representation of  $\mathcal{D}$ .*

- (ii) *A finite set  $\bar{W}$  of points combined with the direction  $(0, \dots, 0, -1)^T$  in  $\mathbb{R}^q$  forms an irredundant V-representation of  $\mathcal{D}$  if and only if*

$$\varphi(y, y^*) \geq 0, \quad y^* \in \bar{W}$$

*is an irredundant H-representation of  $\mathcal{P}$ .*

*Proof.* Statement (i) follows from Corollary 3.3 in [16] and Theorem 4.62 in [6]. One has to take into account the following facts: (a) If a polyhedron  $P$  has a vertex, then an irredundant V-representation of  $P$  consists exactly of all vertices and all extreme directions of  $P$ , see e.g. [18]; (b) Every vertex of  $\mathcal{P}$  is weakly minimal (and even minimal), see e.g. [6, Corollary 4.67]; (c) The vertical facets are exactly the ones which are not  $K$ -maximal, see e.g. [6, Lemma 4.60].

Statement (ii) follows from Corollary 3.2 in [16] and the following facts: (d)  $\mathcal{D}$  has a vertex, see [16, Lemma 5.2], hence an irredundant V-representation of  $\mathcal{D}$  consists of its vertices and extreme directions; (e) every proper face and hence any facet of  $\mathcal{P}$  is weakly minimal, see e.g. [16, Lemma 5.6].  $\square$

Let us now consider the general case of (VLP). Given a (fixed) duality parameter  $c \in \mathbb{R}^q$  with  $c_q \neq 0$ , the dual problem of (VLP) is the following vector linear program with ordering cone  $K := \{y \in \mathbb{R}^q \mid y_1 = 0, \dots, y_{q-1} = 0, y_q \geq 0\}$ :

$$K\text{-maximize } D(u, w, v) \quad \text{s.t.} \quad B^T u = P^T w + v, \quad Yw \geq 0, \quad c^T w = 1 \quad (10)$$

with objective function

$$D(u, w, v) = \left( \frac{c_q}{|c_q|} w_1, \frac{c_q}{|c_q|} w_2, \dots, \frac{c_q}{|c_q|} w_{q-1}, d(u, v) \right)^T. \quad (11)$$

The function  $d : \mathbb{R}^m \times \mathbb{R}^n \rightarrow \mathbb{R}$  is defined as

$$d(u, v) = a^T u^+ - b^T u^- + l^T v^- - s^T v^+, \quad (12)$$

where we define  $\pm\infty \cdot 0 = 0$  and

$$\alpha^+ := \max(0, \alpha), \quad \alpha^- := \max(-\alpha, 0),$$

for  $\alpha \in \mathbb{R}$ , and

$$(\alpha_1, \dots, \alpha_m)^+ := (\alpha_1^+, \dots, \alpha_m^+), \quad (\alpha_1, \dots, \alpha_m)^- := (\alpha_1^-, \dots, \alpha_m^-)$$

for  $(\alpha_1, \dots, \alpha_m) \in \mathbb{R}^m$ .

**Theorem 11.** *Let  $c \in \text{int } C$  such that  $c_q \neq 0$ . Then Theorem 10 also holds for the general case of problem (VLP) and the dual problem (10) if the following generalized coupling function is used:*

$$\varphi(y, y^*) := c_q \sum_{i=1}^{q-1} y_i y_i^* + y_q \left( \frac{|c_q|}{c_q} - \sum_{i=1}^{q-1} c_i y_i^* \right) - \xi(y) |c_q| y_q^*. \quad (13)$$

*Proof.* The generalization of geometric duality to the case of a polyhedral convex ordering cone  $C$  that does not contain lines and has nonempty interior and a duality parameter vector  $c \in \text{int } C$  with  $c_q = 1$  can be found in [8]. In this setting the dual problem is

$$K\text{-maximize } D(u, w) \quad \text{s.t.} \quad B^T u = P^T w, \quad u \geq 0, \quad Y^T w \geq 0, \quad c^T w = 1$$

with objective function  $D(u, w) := (w_1, w_2, \dots, w_{q-1}, a^T u)^T$ . The coupling function in this setting is

$$\varphi(y, y^*) := \sum_{i=1}^{q-1} y_i y_i^* + y_q \left( 1 - \sum_{i=1}^{q-1} c_i y_i^* \right) - \xi(y) y_q^*. \quad (14)$$

Now let us relax the assumption  $c_q = 1$  by  $c_q > 0$ . Introducing new coordinates in the image space of the primal problem by replacing the last component  $y_q \rightarrow c_q^{-1} y_q$  (which effects the data  $Y$  and  $P$  of the primal problem as well as the duality parameter vector  $c$ ), we obtain the dual problem

$$K\text{-maximize } D(u, \bar{w}) \quad \text{s.t.} \quad B^T u = P^T \bar{w}, \quad u \geq 0, \quad Y^T \bar{w} \geq 0, \quad c^T \bar{w} = 1, \quad (15)$$

with objective function  $D(u, \bar{w}) := (\bar{w}_1, \dots, \bar{w}_{q-1}, a^T u)^T$ , where the dual variable  $w \in \mathbb{R}^q$  has been replaced by  $\bar{w}$  with  $\bar{w}_i := w_i$  for  $i = 1, \dots, q-1$  and  $\bar{w}_q := c_q^{-1} w_q$ . The coupling function is

$$\varphi(y, y^*) := \sum_{i=1}^{q-1} y_i y_i^* + c_q^{-1} y_q \left( 1 - \sum_{i=1}^{q-1} c_i y_i^* \right) - \xi(y) y_q^*. \quad (16)$$

Of course, the statement of Theorem 10 remains valid if the coupling function is replaced by (13) and the objective function is defined by (11) for the special case  $d(u, v) = a^T u$ .

Let us now consider the case  $c_q < 0$ . We perform a coordinate transformation  $y \rightarrow -y$  which results in a primal problem with data  $\bar{P} := -P$ ,  $\bar{Y} := -Y$  and a duality parameter  $\bar{c} := -c$ . Since  $\bar{c}_q > 0$ , the result is known for this case from the first part of the proof. The dual problem is

$$K\text{-maximize } D(u, \bar{w}) \quad \text{s.t. } B^T u = \bar{P}^T \bar{w}, \quad u \geq 0, \quad \bar{Y}^T \bar{w} \geq 0, \quad \bar{c}^T \bar{w} = 1$$

with objective function  $D(u, \bar{w}) := (\bar{w}_1, \dots, \bar{w}_{q-1}, a^T u)^T$ . Substitution of the dual variable  $\bar{w}$  by  $-w$  leads to

$$K\text{-maximize } D(u, w) \quad \text{s.t. } B^T u = P^T w, \quad u \geq 0, \quad Y^T w \geq 0, \quad c^T w = 1,$$

with objective function  $D(u, w) := (-w_1, -w_2, \dots, -w_{q-1}, a^T u)^T$ . The coordinate transformation ( $y \rightarrow -y$ ,  $c \rightarrow -c$ ) transforms the coupling function (13) for the case  $c_q > 0$  to (13) for the case  $c_q < 0$ .

The constraints of the general case (VLP) can be expressed as

$$\begin{pmatrix} B \\ -B \\ I \\ -I \end{pmatrix} x \geq \begin{pmatrix} a \\ -b \\ l \\ -s \end{pmatrix}, \quad (17)$$

where  $I$  denotes the  $n \times n$  unit matrix. This leads to the dual constraints

$$B^T u' - B^T u'' = P^T w + v' - v'', \quad u', u'', v', v'' \geq 0, \quad Y^T w \geq 0, \quad c^T w = 1,$$

and the last component of the objective function  $D$  is

$$d(u, v) = a^T u' - b^T u'' + l^T v'' - s^T v'.$$

If some components of the right-hand side in (17) are  $-\infty$ , the corresponding dual variables must vanish (i.e. they do not occur in the dual program). This is taken into account by setting  $\pm\infty \cdot 0 = 0$ .

Finally, in the constraints we set  $u := u' - u''$  and  $v := v' - v''$ , and in the objective function we choose  $u' := u^+$ ,  $u'' := u^-$ ,  $v'' := v^-$ ,  $v' := v^+$ . Since  $a \leq b$  and  $l \leq s$  imply that  $a^T u' - b^T u'' + l^T v'' - s^T v' \leq a^T u^+ - b^T u^- + l^T v^- - s^T v^+$ , this specification does not influence the optimum in case of maximization.  $\square$

We close this section by a short consideration of maximization problems:

$$\max_C P x \quad \text{s.t. } a \leq Bx \leq b, \quad l \leq x \leq s. \quad (\text{VLP}_{\max})$$

In this case we deal with a lower image  $\mathcal{P} := P[S] - C$  of the primal problem and an upper image  $\mathcal{D} := D[T] + K$  of the dual problem, which can be stated as

$$K\text{-maximize } D(u, w, v) \quad \text{s.t. } B^T u = P^T w + v, \quad Yw \geq 0, \quad c^T w = 1 \quad (18)$$

with objective function

$$D(u, v, w) = \left( \frac{c_q}{|c_q|} w_1, \dots, \frac{c_q}{|c_q|} w_{q-1}, \bar{d}(u, v) \right)^T, \quad (19)$$

where

$$\bar{d}(u, v) = b^T u^+ - a^T u^- + s^T v^- - l^T v^+. \quad (20)$$

The ordering cone is again  $K := \{y \in \mathbb{R}^q \mid y_1 = 0, \dots, y_{q-1} = 0, y_q \geq 0\}$ .

**Theorem 12.** *Let  $c \in \text{int } C$  such that  $c_q \neq 0$ . Then Theorem 10 holds for  $(\text{VLP}_{\max})$  and for the dual problem (18) if the following coupling function is used:*

$$\varphi(y, y^*) := -c_q \sum_{i=1}^{q-1} y_i y_i^* - y_q \left( \frac{|c_q|}{c_q} - \sum_{i=1}^{q-1} c_i y_i^* \right) + \xi(y) |c_q| y_q^*. \quad (21)$$

*Proof.* This follows from Theorem 11. We first replace maximization with respect to  $C$  by minimization with respect to  $-C$ . This requires the transformations  $Y \rightarrow -Y$  and  $c \rightarrow -c$ . In the dual problem we substitute  $(u, w, v) \rightarrow -(u, w, v)$ . The resulting maximization problem with respect to  $K$  is finally expressed as a minimization problem with respect to  $K$ , which refers to a coordinate transformation  $y_q^* \rightarrow -y_q^*$ .  $\square$

#### 4. A few remarks on the algorithms

*Bensolve* is an implementation of primal and dual Benson-type algorithms. The origin of these algorithms is discussed in Section 1. The implementation is closely related to the presentation in [8], therefore, we do not present too much details here. *Bensolve* can handle problems which are generalized in comparison to [8] with respect to the following two aspects:

- (i) The assumption  $c_q = 1$  can be replaced by  $c_q \neq 0$ , which allows to use arbitrary solid and line-free polyhedral ordering cones.
- (ii) Maximization is covered in addition to minimization.

These generalizations can be easily realized by the transformations and the dual programs discussed in the previous section. A modification of the algorithms is not necessary.

#### 5. Numerical results

The VLP-solver *Bensolve* is a C-implementation of the algorithms discussed above. In this section, we investigate and compare numerical properties of two applications of multiobjective optimization problems.

The first example (Example 13) is used to show the improvements that could be made in comparison to prior implementations. The second problem class (Example 14) with image space dimension equal to ten shows that *Bensolve* is well suited for problems with high image space dimensions.

The numerical examples were run on a computer with 8GB memory and an Intel® Core™ i5-4200 CPU with 1.60GHz clock. The source code of *Bensolve*<sup>1</sup> was compiled with the GNU Compiler Collection `gcc 5.2.1` and linked against the GNU Linear Programming Kit library `libglp 4.55`.

*Example 13.* In their paper [7], Shao and Ehrgott use a variant of Benson’s algorithm to solve MOLPs originating from a model for optimizing radiotherapy treatment planning. The clinical example (PL) from this paper has three objectives, 1211 constraints and 595 variables (which are additionally nonnegative).

---

<sup>1</sup>The source code is available at <http://bensolve.org> along with documentation and example problems.

The constraint matrix is sparse with 153936 nonzeros. Solving this problem exactly is not tractable. Thus, an approximation variant of Benson’s algorithm has been introduced in [7]: The algorithm stops as soon as the translation of an approximation  $\mathcal{P}^\varepsilon$  of the upper image  $\mathcal{P}$  by the duality parameter  $c$  is contained in the upper image:

$$\mathcal{P}^\varepsilon + \varepsilon \cdot c \subseteq \mathcal{P}.$$

This problem instance is also used for numerical tests in [8] with a Matlab®-implementation preceding the current version of *Bensolve*. The implementation used in [8] is similar to *Bensolve v1.2*<sup>2</sup>. The major difference is the use of a certain warm start heuristic in [8]. Approximations for various error-levels  $\varepsilon$  were computed with both the primal and dual variant of *Bensolve*<sup>3</sup>, where we used the primal simplex as LP-solver in both cases. The run times are listed and compared with those of the implementation from [8] in Table 1. It can be seen that *Bensolve* performs superior to the preceding implementation with regard to the run times. This might be explained partially by the speed-up gained through the transition from a Matlab® to a C-based implementation or by refinements in the algorithm itself. It should be noted that even better results are expected as soon as a warm start heuristic similar to the one used in [8] is implemented. Currently, a kind of “indirect” warm start technique is used, which is imminent in the LP-solver<sup>4</sup> we use for our implementation: between subsequent calls to the solver, the basis factorization of the solution found last remains in memory, hence speeding up the computation time due to the similar structure of the LP’s.

A further explanation for the improved run times is the utilization of a variant of the double description method (see e.g. [19]), adapted specifically for the vertex enumeration part of *Bensolve*: In each iteration we are given an H-representation and a V-representation of a polyhedron  $P$ . We intend to compute the intersection  $P \cap H$  of  $P$  with an affine halfspace  $H$ . We assume that only a few vertices of  $P$  do not belong to  $H$ . The classical variant requires a classification of vertices  $K_1$  of  $P$  belonging to  $H$  and vertices  $K_2$  of  $P$  not belonging to  $H$ . The adjacency relation for every pair  $(v_1, v_2)$ , where  $v_1 \in K_1$  and  $v_2 \in K_2$ , needs to be checked in order to compute the new V-representation. The new variant avoids this classification. One starts with some  $v_1 \in K_1$  and checks every neighboring vertex  $v_2$  whether it belongs to  $K_2$  or not. If not, the procedure is repeated with  $v_2$ .

While approximations with an error threshold below  $5 \cdot 10^{-3}$  were hardly possible with the version used in [8], our new implementation is capable of finding approximations with error level  $1 \cdot 10^{-5}$  in a reasonable amount of time. Additionally, an improvement with respect to the approximation quality can be observed: both the primal and dual variants of *Bensolve* are able to achieve approximations of the upper image with the same approximation error  $\varepsilon$  with fewer vertices (compare Table 2) and by solving less linear programs (Table 3). This behavior can be explained by a new implementation of the vertex enumeration: If two vertices are very close to each other, they are replaced by a single one, while maintaining the outer approximation property.

---

<sup>2</sup>*Bensolve v1.2* is available in the download section at <http://bensolve.org>

<sup>3</sup>The authors thank Dr. Lizhen Shao for supplying the problem data.

<sup>4</sup>GNU Linear Programming Kit, <http://www.gnu.org/software/glpk/glpk.html>

Table 1: Run time comparison

error $\varepsilon$	Run times in seconds		
	[8]	primal	dual
0.3	47	8	4
0.1	91	15	6
$5 \cdot 10^{-2}$	144	22	9
$5 \cdot 10^{-3}$	1411	87	36
$5 \cdot 10^{-4}$		355	153
$5 \cdot 10^{-5}$		1318	692
$1 \cdot 10^{-5}$		3376	2185

Table 2: Number of vertices computed.

error $\varepsilon$	# vertices of $\mathcal{P}$			# vertices of $\mathcal{D}$		
	[8]	primal	dual	[8]	primal	dual
0.3	46	41	21	29	29	40
0.1	104	91	54	61	60	88
$5 \cdot 10^{-2}$	176	156	75	94	94	146
$5 \cdot 10^{-3}$	1456	1097	548	597	595	1078
$5 \cdot 10^{-4}$		7887	3944		4072	7829
$5 \cdot 10^{-5}$		47293	23877		24190	47310
$1 \cdot 10^{-5}$		117981	65235		65354	118668

Table 3: Number of LP's solved

error $\varepsilon$	#LP's solved		
	[8]	primal	dual
0.3	75	72	66
0.1	165	157	145
$5 \cdot 10^{-2}$	270	258	227
$5 \cdot 10^{-3}$	2053	1772	1710
$5 \cdot 10^{-4}$		12549	12348
$5 \cdot 10^{-5}$		75656	75419
$1 \cdot 10^{-5}$		194893	195458

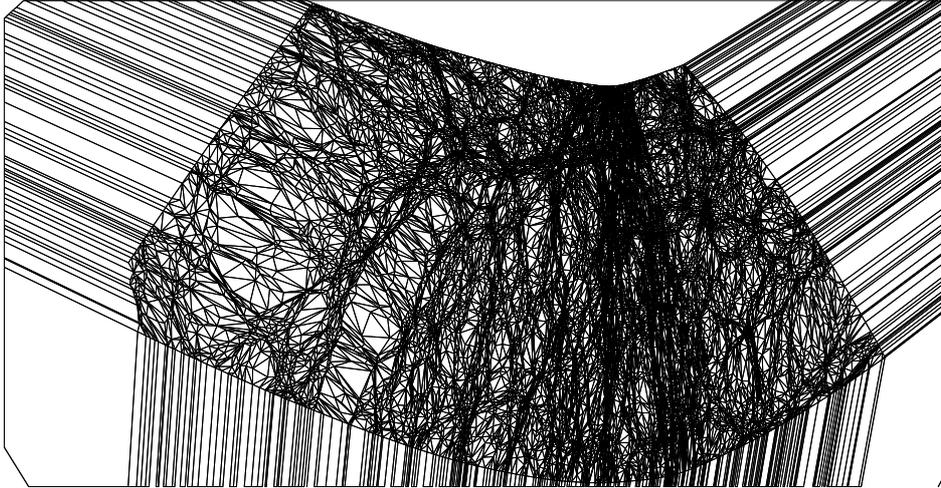


Figure 1: Approximation of the upper image of the problem instance (PL) from [7] with  $\varepsilon = 10^{-4}$

*Example 14.* In [9] Csirmaz presents a rather fascinating application of Benson’s algorithm: Exploring the entropy region formed by the entropies of the nonempty subsets of four random variables, which consists basically in finding a V-representation of a 10-dimensional projection of a high dimensional polytope. Csirmaz used a revised version of Benson’s algorithm to solve several instances of vector linear programs, which are generated by a procedure involving so called “copy steps”. The problem instances have been solved with *Bensolve*<sup>5</sup> and we compared the run times with those of [9] (compare table 4). Here we used the default options of *Bensolve*, the primal algorithm was used with an approximation error of  $\varepsilon = 10^{-8}$ . Although we used a machine with slightly higher specifications than the one used in [9], our implementation seems to be much faster, especially for large problem instances. One explanation might be the adapted vertex enumeration method used in *Bensolve*: In [9] it is stated that due to the huge number of vertices of intermediate polytopes the vertex enumeration (the double description method) becomes the bottleneck of the algorithm. It should be noted that the author of [9] put a lot of effort in reducing numerical errors to a minimum. Therefore, we also list the number of vertices and facets computed by the different implementations in Table 5. While the number of primal vertices concur for all problem instances, the number of dual vertices (which corresponds to the number of facets of the upper image) differs slightly for different instances. This may be the result of numerical errors or imprecision due to the approximating character of Benson’s algorithm.

---

<sup>5</sup>The authors are indebted to Prof. László Csirmaz for providing the `perl`-script that generates the problem instances.

Table 4: Run time comparison of different instances of entropy region mapping problems

problem instance (copy string)	Run times in <i>hh:mm:ss</i>	
	result of [9]	<i>Bensolve</i>
$r=c:ab; s=r:ac; t=r:ad$	00:00:01	00:00:01
$rs=cd:ab; t=r:ad; u=s:adt$	00:06:19	00:00:07
$rs=cd:ab; t=a:bc; u=(cs):abrt$	00:06:51	00:00:08
$rs=cd:ab; t=a:bc; u=b:adst$	00:17:40	00:00:14
$rs=cd:ab; t=a:bc; u=t:acr$	00:18:27	00:00:09
$rs=cd:ab; t=(cr):ab; u=t:acs$	00:22:58	00:00:12
$r=c:ab; st=cd:abr; u=a:bcrt$	00:29:18	00:00:21
$rs=cd:ab; t=a:bc; u=c:abrt$	01:04:32	00:00:49
$rs=cd:ab; t=a:bc; u=s:abcdt$	01:07:01	00:00:43
$rs=cd:ab; t=a:bc; u=(at):bc$	01:39:30	00:01:04
$rs=cd:ab; t=a:bc; u=a:bcst$	04:30:26	00:04:51
$rs=cd:ab; t=a:bc; ua:bdrt$	05:11:25	00:47:44
$rs=cd:ab; tu=cr:ab; v=(cs):abtu$	01:10:10	00:01:27
$rs=ad:bc; tu=ar:bc; v=r:abst$	03:24:37	00:01:25
$rs=cd:ab; t=(cr):ab; uv=cs:abt$	03:34:31	00:01:59
$rs=cd:ab; tu=cr:ab; v=t:adr$	09:20:19	00:02:21
$rs=cd:ab; tu=dr:ab; v=b:adsu$	13:20:08	00:03:01
$rs=cd:ab; tv=dr:ab; u=a:bcrt$	14:34:42	00:02:44
$rs=cd:ab; tu=cs:ab; v=a:bcrt$	22:02:39	00:02:42
$rs=cd:ab; t=a:bc; uv=bt:acr$	37:15:33	00:08:27
$rs=cd:ab; tu=cr:ab; v=a:bcstu$	427:43:30	18:53:29

Table 5: Computed vertices and facets of entropy region mapping problems

problem instance (copy string)	#vertices/#facets	
	result of [9]	<i>Bensolve</i>
$r=c:ab; s=r:ac; t=r:ad$	5/20	5/20
$rs=cd:ab; t=r:ad; u=s:adt$	40/132	40/133
$rs=cd:ab; t=a:bc; u=(cs):abrt$	47/76	47/77
$rs=cd:ab; t=a:bc; u=b:adst$	177/261	177/263
$rs=cd:ab; t=a:bc; u=t:acr$	85/134	85/136
$rs=cd:ab; t=(cr):ab; u=t:acs$	181/245	181/247
$r=c:ab; st=cd:abr; u=a:bcr$	209/436	209/438
$rs=cd:ab; t=a:bc; u=c:abrt$	363/599	363/601
$rs=cd:ab; t=a:bc; u=s:abcdt$	355/591	355/593
$rs=cd:ab; t=a:bc; u=(at):bc$	484/676	484/677
$rs=cd:ab; t=a:bc; u=a:bcst$	880/1238	880/1239
$rs=cd:ab; t=a:bc; ua:bdrt$	2506/2708	2506/2710
$rs=cd:ab; tu=cr:ab; v=(cs):abtu$	19/58	19/58
$rs=ad:bc; tu=ar:bc; v=r:abst$	40/103	40/101
$rs=cd:ab; t=(cr):ab; uv=cs:abt$	30/102	30/102
$rs=cd:ab; tu=cr:ab; v=t:adr$	167/235	167/235
$rs=cd:ab; tu=dr:ab; v=b:adsu$	318/356	318/356
$rs=cd:ab; tv=dr:ab; u=a:bcr$	318/356	318/356
$rs=cd:ab; tu=cs:ab; v=a:bcr$	297/648	297/650
$rs=cd:ab; t=a:bc; uv=bt:acr$	779/1269	779/1271
$rs=cd:ab; tu=cr:ab; v=a:bcstu$	4510/7966	4510/7972

## Bibliography

### References

- [1] J. P. Dauer, Analysis of the objective space in multiple objective linear programming, *J. Math. Anal. Appl.* 126 (2) (1987) 579–593.
- [2] J. P. Dauer, Y.-H. Liu, Solving multiple objective linear programs in objective space, *European J. Oper. Res.* 46 (3) (1990) 350–357.
- [3] H. Benson, Further analysis of an outcome set-based algorithm for multiple-objective linear programming, *Journal of Optimization Theory and Applications* 97 (1) (1998) 1–10.
- [4] H. Benson, 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 (1998) 1–24.
- [5] M. Ehrgott, A. Löhne, L. Shao, A dual variant of Benson’s outer approximation algorithm, *J. Glob. Optim.* 52 (4) (2012) 757–778.
- [6] A. Löhne, *Vector Optimization with Infimum and Supremum*, Vector Optimization, Springer, Berlin, 2011.
- [7] L. Shao, M. Ehrgott, Approximately solving multiobjective linear programmes in objective space and an application in radiotherapy treatment planning, *Math. Methods Oper. Res.* 68 (2) (2008) 257–276.
- [8] A. H. Hamel, A. Löhne, B. Rudloff, Benson type algorithms for linear vector optimization and applications, *J. Global Optim.* 59 (4) (2014) 811–836.
- [9] L. Csirmaz, Using multiobjective optimization to map the entropy region, *Computational Optimization and Applications* (2015) 1–23.
- [10] F. Bökler, P. Mutzel, Output-sensitive algorithms for enumerating the extreme nondominated points of multiobjective combinatorial optimization problems, in: N. Bansal, I. Finocchi (Eds.), *Algorithms – ESA 2015*, Vol. 9294 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2015, pp. 288–299.
- [11] A. Löhne, B. Weißing, Equivalence between polyhedral projection, multiple objective linear programming and vector linear programming, *arXiv:1507.00228* (2015).
- [12] F. Heyde, A. Löhne, Solution concepts in vector optimization: a fresh look at an old story, *Optimization* 60 (12) (2011) 1421–1440.
- [13] J. Evans, R. Steuer, A revised simplex method for linear multiple objective programs., *Math. Program.* 5 (1973) 54–72.
- [14] R. Steuer, *Adbase multiple objective linear programming package*, Tech. rep., University of Georgia, Athens, Georgia (1989).
- [15] M. E. Dyer, The complexity of vertex enumeration methods, *Math. Oper. Res.* 8 (3) (1983) 381–402.

- [16] F. Heyde, A. Löhne, Geometric duality in multiple objective linear programming., *SIAM J. Optim.* 19 (2) (2008) 836–845.
- [17] A. Löhne, B. Weißing, Bensolve - VLP solver, version 2.0.1.  
URL <http://bensolve.org>
- [18] A. Schrijver, Theory of linear and integer programming, Wiley-Interscience Series in Discrete Mathematics, John Wiley & Sons, Ltd., Chichester, 1986, a Wiley-Interscience Publication.
- [19] K. Fukuda, A. Prodon, Double description method revisited, in: *Combinatorics and computer science (Brest, 1995)*, Vol. 1120 of Lecture Notes in Comput. Sci., Springer, Berlin, 1996, pp. 91–111.