1 Multiple Objective Linear Program (MOLP)

An important special case of a vector linear program (VLP) is a multiple objective linear program (MOLP), that is, a linear program (LP) with multiple linear objective functions. BENSOLVE assumes the following formulation of a MOLP:

$$\text{minimize (or maximize)} \quad \begin{pmatrix} P_{11}x_1 + P_{12}x_2 + \cdots + P_{1n}x_n \\ P_{21}x_1 + P_{22}x_2 + \cdots + P_{2n}x_n \\ \cdots \\ P_{q1}x_1 + P_{q2}x_2 + \cdots + P_{qn}x_n \end{pmatrix} \tag{1}$$

subject to the constraints

$$\begin{array}{rcccl} a_1 & \leq & B_{11}x_1 + B_{12}x_2 + \cdots + B_{1n}x_n & \leq & b_1 \\ a_2 & \leq & B_{21}x_1 + B_{22}x_2 + \cdots + B_{2n}x_n & \leq & b_2 \\ & & \cdots & & \\ a_m & \leq & B_{m1}x_1 + B_{m2}x_2 + \cdots + B_{mn}x_n & \leq & b_m \end{array} \tag{2}$$

$$\begin{array}{rcccl} l_1 & \leq & x_1 & \leq & s_1 \\ l_2 & \leq & x_2 & \leq & s_2 \\ & & \cdots & & \\ l_n & \leq & x_n & \leq & s_n. \end{array} \tag{3}$$

2 Multiple Objective Mixed Integer Program (MOMIP)

This problem is defined likewise the (MOLP) but some of the variables $x_1, \ldots, x_n$ (or even all of them) can assumed to be integer or binary.

3 Vector Linear Program

In this more general setting, the minimization (or maximization) in (1) is defined with respect to a partial ordering $\leq_C$ induced by a polyhedral cone $C \subseteq \mathbb{R}^q$, that is

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_q \end{pmatrix} \leq_C \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_q \end{pmatrix} \iff \begin{pmatrix} z_1 - y_1 \\ z_2 - y_2 \\ \vdots \\ z_q - y_q \end{pmatrix} \in C. \tag{4}$$

BENSOLVE assumes that C has a non-empty interior and contains no lines. The polyhedral cone C is assumed to be given by one of the following two representations:

The `CONE` representation is given by a matrix Y with q rows and o columns. A vector $(y_1, \ldots, y_q)$ belongs to C if and only if there are nonnegative real numbers $v_1, v_2, \ldots, v_o \geq 0$

such that

$$\begin{array}{rcl} y_1 & = & Y_{11}v_1 + Y_{12}v_2 + \cdots + Y_{1o}v_o \\ y_2 & = & Y_{21}v_1 + Y_{22}v_2 + \cdots + Y_{2o}v_o \\ & & \cdots \\ y_q & = & Y_{q1}v_1 + Y_{q2}v_2 + \cdots + Y_{qo}v_o. \end{array} \tag{5}$$

The columns of the matrix Y are generating vectors of the polyhedral cone C.

The `DUALCONE` representation is given by a matrix Z with q rows and p columns. A vector $(y_1, \ldots, y_q)$ belongs to C if and only if the following inequalities are satisfied:

$$\begin{array}{rcl} Z_{11}y_1 + Z_{21}y_2 + \cdots + Z_{q1}y_q & \geq & 0 \\ Z_{12}y_1 + Z_{22}y_2 + \cdots + Z_{q2}y_q & \geq & 0 \\ & \cdots & \\ Z_{1p}y_1 + Z_{2p}y_2 + \cdots + Z_{qp}y_q & \geq & 0. \end{array} \tag{6}$$

The columns of the matrix Z are generating vectors of the dual cone of the polyhedral cone C.

4 Vector Mixed Integer Program

This problem is defined likewise the (VLP) but some of the variables $x_1, \ldots, x_n$ (or even all of them) can assumed to be integer or binary.

5 VLP input format

The VLP input format is an extension of the GLPK LP/MIP format to the case of multiple objective linear programs (MOLP) / multiple objective mixed integer programs (MOMIP) and vector mixed integer programs (VMIP). It is a DIMACS-like format (see `http://dimacs.rutgers.edu/Challenges/`). A problem instance in VLP format is stored as a plain ASCII text file containing lines of several types. Lines are terminated by the end-of-line character. Each line begins with a one-character designator to identify the line type. Valid line designators are:

`c`	comment line
`p`	program line
`i`	row descriptor line
`j`	column descriptor line
`a`	constraint coefficient descriptor line
`o`	objective coefficent descriptor line
`k`	cone generator coefficient descriptor line (for VLP) or duality parameter descriptor line (for VLP)
`e`	end of file

The line designator is followed by several fields which are separated by at least one blank space.

A **comment line** begins with the lower-case character c:

```
c This is a comment line
```

The first line which is not a comment line must be the **program line**. In case of a MOLP, it begins with the lower-case character p followed by 7 fields:

```
p CLASS DIR ROWS COLS NZ OBJ OBJNZ
```

The `CLASS` field defines the problem class and must contain either the keyword `vlp` or `vmip`. In case of the keyword `vlp` we assume that all variables are continuous, whereas in the case of `vmip` integer and binary variables can be specified. The `DIR` field contains the optimization direction and must contain either `min` (for minimization) or `max` (for maximization). The `ROWS` and `COLS` fields represent the number of rows and columns, that is, the integers m and n in (2), respectively. The `NZ` field contains the number of non-zero constraint coefficients, that is, the number of non-zero entries of the coefficient matrix B in (2). The `OBJ` field contains the number of objectives, that is, the integer q in (1). The field `OBJNZ` contains the number of non-zero objective coefficients, that is, the number of non-zero entries of the $q \times n$-matrix P in (1).

A **row descriptor line** specifies the type of a constraint in (2). Such a line has one of the following formats:

```
i ROW f
i ROW l VAL1
i ROW u VAL1
i ROW d VAL1 VAL2
i ROW s VAL1
```

`ROW` contains the index of the constraint, which is an integer between 1 and m. The next character (`f`, `l`, `u`, `d`, `s`) specifies the type of the constraint. `VAL1` and `VAL2` contain the floating-point constraint values. For a constraint in (2), say the k-th constraint

$$a_k \leq B_{k1}x_1 + B_{k2}x_2 + \cdots + B_{kn}x_n \leq b_k,$$

the following types are possible:

`f`	no bound	$a_k = -\infty$	$b_k = +\infty$
`l`	lower bound	$a_k =$ `VAL1`	$b_k = +\infty$
`u`	upper bound	$a_k = -\infty$	$b_k =$ `VAL1`
`d`	double-sided bound	$a_k =$ `VAL1`	$b_k =$ `VAL2`
`s`	equation	$a_k =$ `VAL1`	$b_k =$ `VAL1`

A line of the form

```
i ROW s 0
```

is a default row descriptor line and can be omitted.

A **column descriptor line** specifies the type of a variable in (3). In case of the problem class `vlp`, such a line has one of the following formats:

```
j COL f
j COL l VAL1
j COL u VAL1
j COL d VAL1 VAL2
j COL s VAL1
```

Fo the problem class `vmip`, there is an additional field. Such a line has one of the following formats:

```
j COL KIND f
j COL KIND l VAL1
j COL KIND u VAL1
j COL KIND d VAL1 VAL2
j COL KIND s VAL1
```

`COL` contains the index of the variable, which is an integer between 1 and n. `KIND` specifies the type of the variable in case of problem class `vmip` as follows:

`c` - continuous variable

`i` - integer variable

`b` - binary variable (in this case all remaining fields must be omitted)

The next character (`f`, `l`, `u`, `d`, `s`) specifies the constraint type of the variable. `VAL1` and `VAL2` contain the floating-point values of the variable bounds. For the variable x_k in (3), we have $l_k \leq x_k \leq u_k$. The following variable types are possible:

`f`	free variable	$l_k = -\infty$	$s_k = +\infty$
`l`	variable with lower bound	$l_k = \texttt{VAL1}$	$s_k = +\infty$
`u`	variable with upper bound	$l_k = -\infty$	$s_k = \texttt{VAL1}$
`d`	double-bounded variable	$l_k = \texttt{VAL1}$	$s_k = \texttt{VAL2}$
`s`	fixed variable	$l_k = \texttt{VAL1}$	$s_k = \texttt{VAL1}$

If for some column its descriptor line does not appear in the file, by default that column is assumed to be non-negative (in case of LP class) or binary (in case of MIP class).

The default column descriptor lines

```
j COL l 0
```

for the case of the class `vlp` and

```
j COL b
```

for the case of the class `vmip` can be omitted.

A **constraint coefficient descriptor line** has the format:

```
a ROW COL VAL
```

For every non-zero constraint coefficient B_{ij} in (2), exactly one coefficient descriptor line must be specified. `ROW` and `COL` contain the row number i and the column number j of the coefficient B_{ij}, respectively. `VAL` contains the floating-point coefficient B_{ij}. A coefficient descriptor line where `VAL` equals zero is allowed. The number of constraint coefficient descriptor lines must be exactly the same as specified in the field `NZ` of the problem line.

An **objective coefficient descriptor line** has the format:

```
o ROW COL VAL
```

For every non-zero objective coefficient P_{ij} in (1), exactly one coefficient descriptor line must be specified. `ROW` and `COL` contain the row and column numbers i and j of P_{ij} and `VAL` contains the floating-point value of P_{ij}. A coefficient descriptor line where `VAL` equals zero is allowed. The number of constraint coefficient descriptor lines must be exactly the same as specified in the field `OBJNZ` of the problem line.

In case of a vector linear program (beyond the special case of MOLP), the program line has 10 fields (i.e. three additional fields):

```
p CLASS DIR ROWS COLS NZ OBJ OBJNZ CTYPE GEN GENNZ
```

The `CTYPE` field specifies the type of cone representation, compare Section 4. Valid entries are `cone` and `dualcone`. If `CTYPE` is specified, the fields `GEN` and `GENNZ` must be specified, too. The `GEN` field is the number of generating vectors, i.e., either o in (5) or p in (6). The `GENNZ` field contains the number of non-zero cone coefficients, that is, the number of non-zero entries of either the matrix Y or the matrix Z, dependent of which one is given.

A **cone coefficient descriptor line** has the format:

```
k ROW COL VAL
```

For every non-zero cone coefficient Y_{ij} in (5), respectively, Z_{ij} in (6), exactly one cone descriptor line must be specified. If `CTYPE`=`cone`, `ROW` and `COL` contain the row and column numbers $1 \leq i \leq q$ and $1 \leq j \leq o$ and `VAL` contains the floating-point value of Y_{ij}. If CTYPE=`dualcone`, `ROW` and `COL` contain the row and column numbers $1 \leq i \leq q$ and $1 \leq j \leq p$ and `VAL` contains the floating-point value of Z_{ij}. A cone descriptor line where `VAL` equals zero is allowed. The number of cone coefficient descriptor lines must be exactly the same as specified in the field `GENNZ` of the problem line.

A **duality parameter descriptor line** has the format:

```
k ROW 0 VAL
```

Do not count duality parameter descriptor lines in the field `GENNZ` of the problem line.