To appear in *Optimization*
Vol. 00, No. 00, Month 20XX, 1–16

A unifying approach to solve a class of rank-three programs involving linear and quadratic functions

Riccardo Cambini* and Claudio Sodini

Department of Economics and Management, University of Pisa
Via Cosimo Ridolfi 10, 56124 Pisa, ITALY;

(*Received 00 Month 20XX; accepted 00 Month 20XX*)

The aim of this paper is twofold. First, the so called "optimal level solutions" method is described in a new unifying framework with the aim to provide an algorithmic scheme able to approach various different classes of problems. Then, the "optimal level solutions" method is used to solve a class of low-rank programs involving linear and quadratic functions and having a polyhedral feasible region. In particular, the considered class of programs covers, among all, rank-three d.c., multiplicative and fractional programs. Some optimality conditions are used to improve the performance of the proposed algorithm.

Keywords: Fractional programming; Multiplicative programming; d.c. programming; Optimal level solutions; Global optimization.

AMS Subject Classification: AMS - 2000 Math. Subj. Class. 90C05, 90C26, 90C31.
JEL - 1999 Class. Syst. C61, C63.

1. Introduction

In this paper the following class of low-rank nonconvex problems involving linear and quadratic functions is studied from both a theoretical, an algorithmic and a computational point of view:

$$\inf_{x\in X} f(x) = \frac{1}{2}x^TQx + q^Tx + (c^Tx + c_0)\phi(d^Tx + d_0),$$

where $X \subset \Re^n$ is a nonempty polyhedron, $Q \in \Re^{n\times n}$ is symmetric and positive definite, $c, d, q \in \Re^n$, $c \neq 0$, $d \neq 0$, c and d linearly independent, $c_0, d_0 \in \Re$. The scalar function $\phi(\xi)$ is assumed to be continuous and to be defined for all the values $d^Tx + d_0$ with $x \in X$. Moreover, function $\phi(\xi)$ is assumed to eventually change its increasing /decreasing behavior just a finite number of times (for example, $\phi(\xi)$ may be monotone, convex/concave, or a polynomial).

Notice that the class of problems described in P covers both multiplicative and fractional rank-three programs (in the sense that function f actually depends on the three variables $y_1 = \frac{1}{2}x^TQx+q^Tx$, $y_2 = c^Tx+c_0$ and $y_3 = d^Tx+d_0$). Specifically

*Corresponding author. Email: riccardo.cambini@unipi.it

speaking, the following functions are examples of objectives for problem P:

$$f(x) = \frac{1}{2}x^T Qx + q^T x + (c^T x + c_0)(d^T x + d_0)^k \quad \text{where } k > 0 \text{ integer;}$$

$$f(x) = \frac{1}{2}x^T Qx + q^T x + (c^T x + c_0)\mathbf{e}^{d^T x + d_0};$$

$$f(x) = \frac{1}{2}x^T Qx + q^T x + \frac{c^T x + c_0}{(d^T x + d_0)^k} \quad \text{where } d^T x + d_0 > 0 \text{ and } k > 0;$$

$$f(x) = \frac{1}{2}x^T Qx + q^T x + \frac{c^T x + c_0}{\log(d^T x + d_0)} \quad \text{where } d^T x + d_0 > 1.$$

These examples point out the wideness of the class of problems P which, at the best of our knowledge, have never been studied in the literature by means of a unifying framework. Notice also that even in the particular case $\phi(\xi) = \xi$ the problems considered in this paper differ from the rank-two ones studied in [8–11, 15].

The solution method proposed to solve this class of problems is based on the so called "optimal level solutions" method (see [6, 9, 13–17, 26]). It is known that this is a parametric method, which finds the optimum of the problem by determining the minima of particular subproblems. In particular, the optimal solutions of these subproblems are obtained by means of a sensitivity analysis which maintains the optimality conditions. Notice that this method solves in an unifying approach both multiplicative and fractional rank-three problems. In the literature various methods have been proposed to solve either multiplicative or fractional problems, but none of them (at the best of our knowledge) solves in an unifying way all of the problems described by P (see for all [23, 24]).

In Section 2 the Optimal Level Solutions method is described in details, with the aim to provide a new unifying algorithmic framework to approach several of the different classes of problems studied so far by means of such a method. In Section 3 the optimal level solutions method is applied to the class of rank-three programs previously described, and an underestimation function is also studied to improve the performance of the algorithm. Finally, some computational results are summarized in Section 4.

2. The Optimal Level Solutions Approach

The aim of this section is twofold, that is to provide a clear general description of the *Optimal Level Solution Method* and to give a new unifying algorithmic approach for the different classes of functions studied so far by means of such a method. In this light, and for the sake of clearness and completeness, a brief survey is provided in the followings.

The optimal level solution method has been first proposed by A. Cambini, L. Martein and C. Sodini in [1, 2] and some first algorithmic results have been given in [3, 5, 20, 21, 28]. Such an approach has been applied to study various classes of problems in [4, 7, 22, 25–27]. Later, the optimal level solution method has been applied to study more general classes of problems, providing also computational results obtained by extensive numerical experiences [8–11, 13–19].

2.1. *The basics*

In order to describe the basics of this parametric approach let us consider the following class of problems:

$$P : \begin{cases} \inf \ f(x) = g(x, d^T x + d_0) \\ x \in X \subset \Re^n \end{cases},$$

where the feasible region $X \neq \emptyset$ is a polyhedron, that is a set given by linear equalities and/or inequalities, $f : X \to \Re$ and $g : (X \times \Lambda) \to \Re$ are continuous functions, where $\Lambda = \left\{\xi \in \Re : \ \xi = d^T x + d_0, \ x \in X\right\}$. The values $\xi \in \Lambda$ are said to be *feasible levels*. Notice that since X is convex the set of feasible levels Λ is convex too, that is to say that it is an interval. In this light, the following further notations can be introduced for the boundaries of Λ:

$$\xi_{min} = d_0 + \inf_{x \in X} d^T x \quad \text{and} \quad \xi_{max} = d_0 + \sup_{x \in X} d^T x.$$

Clearly, if X is a compact set then ξ_{min} and ξ_{max} are finite and the set Λ of feasible levels is compact too. For each feasible level $\xi \in \Lambda$ we can define:

$$P_\xi : \begin{cases} \min \ g_\xi(x) = g(x, \xi) \\ x \in X_\xi = \{x \in X : \ d^T x + d_0 = \xi\} \end{cases},$$

where for all $\xi \in \Lambda$ functions $g_\xi : X_\xi \to \Re$ are assumed to be quasiconvex and such that $\arg\min_{x \in X_\xi} g_\xi(x) \neq \emptyset$. Given a feasible level $\xi \in \Lambda$, the optimal solutions of the parametric subproblems P_ξ are called *optimal level solutions*; in this light the following further notations can be introduced:

$$z(\xi) = \min_{x \in X_\xi} g_\xi(x) \quad , \quad S_\xi = \arg\min_{x \in X_\xi} g_\xi(x) \quad , \quad S = \bigcup_{\xi \in \Lambda} S_\xi.$$

Notice that:

- Since $g_\xi(x)$ is quasiconvex then S_ξ is a convex set for all feasible levels $\xi \in \Lambda$
- If $g_\xi(x)$ is strictly quasiconvex then S_ξ is a singleton for all feasible levels $\xi \in \Lambda$
- The optimal global solution of P can be found just by analyzing the set of optimal level solutions S for all the feasible levels $\xi \in \Lambda$, that is to say:

$$\inf_{x \in X} f(x) = \inf_{x \in S} f(x) = \inf_{\xi \in \Lambda} z(\xi).$$

Obviously, an optimal solution of problem P (if it exists) is also an optimal level solution and, in particular, it is the optimal level solution with the smallest value. The idea of this approach is then to scan all the feasible levels, studying the corresponding optimal level solutions, until either the minimizer of the problem is reached or the infimum is obtained along an halfline. The optimal level solutions method will be algorithmically described by means of the following procedures:

- "*Main*()" : this procedure initializes the algorithm by determining the set of feasible levels and by computing a starting incumbent solution, then the feasible region is partitioned and the various partitions are analyzed by means of procedure "*Visit*()";

- "*Visit*()" : this procedure scans iteratively the given set of feasible levels, from the smallest level up to the biggest one, providing the best visited optimal level solution (or the infimum reached along an halfline);
- "*LevelSolutions*()" : this procedure is specifically dependent to the particular class of functions to be optimized; basically, by means of a sort of sensitivity analysis on Karush-Kuhn-Tucker conditions, this procedure determines the set of optimal level solutions corresponding to a specific basis of the polyhedron.

2.2. *Parametric search of the optimal level solutions*

From an algorithmic point of view, the idea of this method is to iteratively visit the feasible levels by analyzing the corresponding optimal level solutions. In this light, starting from a specific optimal level solution x' it is necessary to determine the optimal level solutions on the polyhedron X sharing with x' the same basis. Notice that this part of the algorithm is very dependent to the particular class of objective function of the problem and, generally speaking, it is based on a parametric sensitivity analysis on the Karush-Kuhn-Tucker optimality conditions.

With this aim, some further assumptions on problem P are usually assumed in order to let the Karush-Kuhn-Tucker conditions become both necessary and sufficient. Specifically speaking, functions $g_\xi(x)$ can be assumed to be pseudoconvex or, following the lines proposed in [10], can be transformed to equivalent quadratic convex functions $\gamma_\xi(x)$ such that $\arg\min_{x\in X_\xi} g_\xi(x) = \arg\min_{x\in X_\xi} \gamma_\xi(x)$ (in this last case problem P has been said to be "parametrically-convexifiable").

In this light, the optimal level solutions of $\overline{P}_\xi$ can be determined by verifying the Karush-Kuhn-Tucker conditions for problems $\overline{P}_\xi$.

The sensitivity analysis on such Karush-Kuhn-Tucker conditions allows to determine a set of points $\hat{x}(\xi)$ which result to be feasible for levels $\xi \in [\xi', \xi_F]$ and to be optimal for levels $\xi \in [\xi', \xi_O]$, so that they result to be optimal level solutions for levels $\xi \in [\xi', \xi_m]$, with $\xi_m := \min\{\xi_F, \xi_O\}$. This can be done in a procedure "*LevelSolutions*()" which is dependent to the particular class of considered problems.

Procedure **LevelSolutions**(*inputs*: ξ', x'; *outputs*: $\hat{x}(\xi)$, ξ_F, ξ_O)

$\vdots$

end proc.

Clearly, in order to have a working algorithm some properties need to be verified by the objective function of the problem, that is to say properties which guarantee the existence of a basis providing a value $\xi_m > \xi'$. See for example [8–11, 13–18] for some particular classes of problems studied with this parametric approach.

2.3. *Iterative visit*

The iterative visit of the feasible region is left to procedure "*Visit*()", which analyzes the feasible levels of the given problem P in increasing order from the level ξ_{start} up to the level ξ_{end}. In each iteration of the while cycle the procedure starts from the feasible level ξ' and its optimal level solution x', then procedure "*LevelSolutions*()" is invoked in order to determine a set of optimal level solutions belonging to the same basis of the polyhedron and corresponding to the feasible levels $[\xi', \xi_m]$, with $\xi' < \xi_m$. By means of subprocedure "*MinRestriction*()" the infimum/minimum of the continuous single valued function $z(\xi)$ in the interval $[\xi', \xi_m]$

***Procedure* Visit**(*inputs*: P, ξ_{start}, ξ_{end}, x'_{start}, $\overline{x}$, UB; *outputs*: $\overline{x}$, UB)
$\xi' := \xi_{start}$; $x' := x'_{start}$;
Optional : *if* $\xi' < \xi_{end}$ and $\psi(\xi') > UB$ *then*
$[\xi', x'] := Skip(\xi', x', \xi_{end}, UB)$;
end if;
while $\xi' < \xi_{end}$
set $[\hat{x}(\xi), \xi_F, \xi_O] := LevelSolutions(\xi', x')$;
let $\xi_m := \min\{\xi_F, \xi_O\}$ and $z(\xi) = f(\hat{x}(\xi))$;
set $[\overline{\xi}, z_{inf}] := MinRestriction(z(\xi), [\xi', \xi_m])$;
if $z_{inf} = -\infty$ *then*
$\overline{x} := []$; $UB := -\infty$; $\xi' := +\infty$
else
if $z_{inf} < UB$ *then*
$UB := z_{inf}$;
if $\overline{\xi} = +\infty$ *then* $\overline{x} := []$ *else* $\overline{x} := \hat{x}(\overline{\xi})$ *end if*;
end if;
set $\xi' := \xi_m$; *if* $\xi' < \xi_{end}$ *then* $x' := \hat{x}(\xi')$ *end if*;
end if;
# Optional : *if* $\xi' < \min\{\xi_O, \xi_{end}\}$ *then*
$[\xi', x'] := Jump(\xi', x', \xi_O, \xi_{end}, UB)$;
end if;
# Optional : *if* $\xi' < \xi_{end}$ and $\psi(\xi') > UB$ *then*
$[\xi', x'] := Skip(\xi', x', \xi_{end}, UB)$;
end if;
end while;
end proc.

is determined; the obtained value is used to to improve the incumbent optimal solution $\overline{x}$ and its value UB. At the end of the iteration the starting level ξ' is moved ahead and its optimal level solution x' is updated. Notice that procedure "*MinRestriction*()" can be implemented numerically, and eventually improved for specific functions $f(x)$ (see [9, 26]).

The computing performance of procedure "*Visit*()" can be improved by means of the two optional subprocedures "*Skip*()" and "*Jump*()" which will be described in details in the next subsection.

Procedure "*Visit*()" points out the reason the optimal level solution approach is said to be a "simplex-like" method. In the very particular case $g(x, d^T x + d_0) = d^T x + d_0$ problem P is nothing but a linear programming problem, moreover the while cycle is nothing but the simplex iterative visit from vertex to vertex of the polyhedron. In other words, the optimal level solution approach can be seen as a generalization of the simplex algorithm and hence it has the same algorithmic complexity of the simplex algorithm (it si very well known that in the worst case all of the vertices need to be visited).

2.4. *Implicit visit and algorithm improvements*

As it has been shown in [8–11, 13–17], it is possible to greatly improve the performance of the solution algorithm by implicitly visiting some of the feasible levels. Specifically speaking, it is possible to avoid the visit of the feasible levels which are not able to improve the incumbent optimal level solution. The main idea (see for example [10]), is the use of a so called *underestimation function*, that is a function

$\psi(\xi)$ which verifies the following property for all the feasible levels $\xi \in \Lambda$:

$$\psi(\xi) \leq \min_{x \in X_\xi} f(x) = z(\xi).$$

Clearly, such an underestimation function is specifically dependent to the particular class of functions to be optimized (see for example [9, 10, 13, 15, 17] where some underestimation functions have been studied). In this light, some feasible levels can be implicitly visited by means of the following two optional subprocedures:

- "*Skip*()" : evaluates the opportunity of avoiding the explicit visit of some feasible levels by means of the underestimation function $\psi(\xi)$;
- "*Jump*()" : tries to implicitly visit some feasible levels in the case $\xi_O > \xi_F$ by means of $z(\xi) = f(\hat{x}(\xi))$ (which in the interval $[\xi_F, \xi_O]$ results to be an underestimation function too).

Procedure **Skip**(*inputs*: ξ', x', ξ_{end}, UB; *outputs*: ξ', x')
  let $\mathcal{L} := [\xi', \xi_{end}] \cap \{\xi \in \Re : \psi(\xi) \leq UB\}$;
  if $\mathcal{L} = \emptyset$ *then* $\xi' := \xi_{end}$ *else*
    $\xi' := \min\{\mathcal{L}\}$;
    if $\xi' < \xi_{end}$ *then* $x' := \arg\min\{P_{\xi'}\}$ *end if*;
  end if;
end proc.

Procedure "Skip()" starts from level ξ' and tries to avoid the explicit visit of the feasible levels such that $UB < \psi(\xi) \leq \min_{x \in X_\xi} f(x)$ since they will not provide an optimal level solution better than the incumbent one. The implementation of this procedure can be optimized for particular classes of functions; generally speaking a numerical approach is needed, while some classes of functions allow to analytically determine $\min\{\mathcal{L}\}$.

Procedure **Jump**(*inputs*: ξ', x', ξ_O , ξ_{end}, UB; *outputs*: ξ', x')
  $[\tilde{\xi}, \tilde{z}_{inf}] := MinRestriction(z(\xi), [\xi', \min\{\xi_O, \xi_{end}\}])$;
  if $\tilde{z}_{inf} >= UB$ *then*
    $\xi' := \xi_O$;
    if $\xi' < \xi_{end}$ *then* $x' := \arg\min\{P_{\xi'}\}$ *end if*;
  end if;
end proc.

Procedure "Jump()" looks for the minimum value of $z(\xi) = f(\hat{x}(\xi))$ in the interval $[\xi_F, \min\{\xi_O, \xi_{end}\}]$, if this is greater than or equal UB then there is no need to explicitly visit those feasible levels since they will not provide an optimal level solution better than the incumbent one.

Procedures "*Skip*()" and "*Jump*()" result to be as more effective as smaller is the value UB of the incumbent solution. For this very reason, in order to improve the algorithm performance it is important in procedure "Main()" to initialize the algorithm computing a "good" starting incumbent solution. Clearly, in the case procedures "*Skip*()" and "*Jump*()" are not used, then the whole set of feasible levels Λ is visited and all of the optimal level solutions are examined.

***Procedure* Main**(*inputs*: P; *outputs*: Opt, $OptVal$)
compute the values $\xi_{min} := d_0 + \inf_{x \in X} d^T x$ and $\xi_{max} := d_0 + \sup_{x \in X} d^T x$;
if $\xi_{min} = \xi_{max}$ *then* $Opt := \arg\min\{P_{\xi_{min}}\}$; $OptVal := f(Opt)$;
else let $\xi_{big} >> 0$ and set $\xi_1 := \max\{-\xi_{big}, \xi_{min}\}$; $\xi_2 := \min\{\xi_{big}, \xi_{max}\}$;
compute $x'_1 := \arg\min\{P_{\xi_1}\}$ and $x'_2 := \arg\min\{P_{\xi_2}\}$;
set $\xi_M := \frac{\xi_1 + \xi_2}{2}$ and compute $x'_M := \arg\min\{P_{\xi_M}\}$;
if $f(x'_1) < f(x'_2)$ *then* $\overline{x} := x'_1$ *else* $\overline{x} := x'_2$ *end if*;
if $f(x'_M) < f(\overline{x})$ *then* $\overline{x} := x'_M$ *end if*;
Set $UB := f(\overline{x})$;
if $\xi_2 < \xi_{max}$ *then* $[\overline{x}, UB] := Visit(P, \xi_2, \xi_{max}, x'_2, \overline{x}, UB)$ *end if*;
if $\xi_1 > \xi_{min}$ *then* $[\overline{x}, UB] := Visit(\tilde{P}, -\xi_1, -\xi_{min}, x'_1, \overline{x}, UB)$ *end if*;
$[\overline{x}, UB] := Visit(P, \xi_1, \xi_2, x'_1, \overline{x}, UB)$;
$Opt := \overline{x}$ and $OptVal := UB$;
end if;
end proc.

2.5. *Algorithm initialization*

It is now possible to provide the procedure to be launched in order to solve problem P, that is procedure "*Main*()". Such a procedure has to manage both the case of an unbounded region and the case of numerical problems given by feasible levels which are very big in absolute value. For this very reasons, it is better to partition the feasible region into at least three parts, a central compact one and two other parts eventually unbounded. In this light, a value $\xi_{big} >> 0$ is set and the set of feasible levels is partitioned in the intervals $[\xi_{min}, \xi_1]$, $[\xi_1, \xi_2]$ and $[\xi_2, \xi_{max}]$. Three optimal level solutions (x'_1, x'_2, x'_M) are computed in order to determine a starting "good" incumbent optimal solution $\overline{x}$ and a "good" incumbent optimal value "UB". The three intervals $[\xi_{min}, \xi_1]$, $[\xi_1, \xi_2]$ and $[\xi_2, \xi_{max}]$ are then visited by means of procedure "*Visit*()". In order to improve the performance of the algorithm (subprocedures "*Skip*()" and "*Jump*()" are as more effective as smaller is the value UB) the first partitions to be visited are $[\xi_{min}, \xi_1]$ and $[\xi_2, \xi_{max}]$ since they are the ones where the problem can reach the infimum in the case no mimimum exists. It is now worth noticing that, for the sake of convenience, procedure "*Visit*()" scans the feasible levels in increasing order. On the other hand, in the case $\xi_{min} = -\inf$ it is not possible from an algorithmic point of view to visit the feasible levels $[\xi_{min}, \xi_1]$ in increasing order. To manage this particular case the following problem equivalent to P can be actually studied:

$$P \equiv \tilde{P} : \begin{cases} \inf \ f(x) = \tilde{g}(x, \tilde{d}^T x + \tilde{d}_0) \\ x \in X \subset \Re^n \end{cases},$$

where $\tilde{g}(x, \xi) = g(x, -\xi)$, $\tilde{d} = -d$ and $\tilde{d}_0 = -d_0$. Notice that the feasible region is untouched while the objective function has been simply rewritten in an equivalent form. Since the new linear function $\tilde{d}^T x + \tilde{d}_0$ allow the visit of the feasible levels in increasing order, it correspond to the visit of the levels in decreasing order with respect to $d^T x + d_0$.

2.6. *Final notes*

The optimal level solutions approach is a simplex-like method, that is to say that in each iteration given a certain basis of the polyhedron X a set of optimal level

solutions is determined and visited until either the feasibility or the optimality is lost. At that time, a new basis is determined and a new iteration starts. Generally speaking, the correctness of these algorithms is given by the fact that the optimal solution of P (if it exists) is also an optimal level solution and that all of the optimal level solutions are visited. The convergence of these algorithms is given by some assumptions on the objective function, assumptions aimed to guarantee that $\xi_m > \xi'$ and that a basis is visited at most a finite number of times (for the problems studied in [8–11, 13–18] it is proved that a basis can be visited just once); in this light the convergence follows noticing that a polyhedron has just a finite number of basis. Clearly, since these are simplex-like algorithms, they have the same complexity of simplex algorithm, that is to say that in the worst case all of the vertices must be visited. Actually, the numerical experiences done so far provide very good time performances (see [8–11, 13–18]).

3. Solving a class of rank-three programs

Let us now consider the following class of low-rank nonconvex problems involving linear and quadratic functions:

$$P : \begin{cases} \inf \ f(x) = \frac{1}{2}x^T Q x + q^T x + (c^T x + c_0)\phi(d^T x + d_0) \\ x \in X = \{x \in \Re^n : \ Ax \underline{\leq} b\} \end{cases},$$

where $A \in \Re^{m\times n}$, $b \in \Re^m$, $Q \in \Re^{n\times n}$ is symmetric and positive definite, $c, d, q \in \Re^n$, $c \neq 0$, $d \neq 0$, c and d linearly independent, $c_0, d_0 \in \Re$ and $X \neq \emptyset$ (notice that if $c = \alpha d$, $\alpha \in \Re$, then problem P reduces to a rank-two problem belonging to the class of programs already studied in [15]). The scalar function $\phi(\xi)$ is assumed to be continuous and to be defined for all the values in $\Lambda = \left\{\xi \in \Re : \ \xi = d^T x + d_0, \ x \in X\right\}$. Moreover, function $\phi(\xi)$ is assumed to eventually change its increasing /decreasing behavior just a finite number of times (for example, $\phi(\xi)$ may be monotone, convex/concave, or a polynomial).

The aim of this section is to show how problem P can be solved by means of the *optimal level solutions approach*. Let us recall that in order to find a global minimum (assuming that one exists) it would be necessary to solve problems P_ξ for all the feasible levels $\xi \in \Lambda$. In this section we will show that this can be done by means of a finite number of iterations, taking into account the results and the detailed description provided in the previous section. In this light, we just need to determine for this specific problem the procedure "*LevelSolutions*()" and the underestimation function $\psi(\xi)$.

3.1. *Determining the optimal level solutions*

The following parametric subproblem can be obtained just by adding to problem P the constraint $d^T x + d_0 = \xi$:

$$P_\xi : \begin{cases} \inf \ f_\xi(x) \\ x \in X_\xi = \{x \in \Re^n : \ Ax \leq b, \ d^T x + d_0 = \xi\} \end{cases},$$

where:

$$f_\xi(x) = \frac{1}{2}x^T Q x + q^T x + (c^T x + c_0)\phi(\xi) = \frac{1}{2}x^T Q x + (q + \phi(\xi)c)^T x + \phi(\xi)c_0.$$

Subproblem P_ξ is strictly convex and quadratic, hence it always admits an unique minimum point, which is the so called *optimal level solution*.

Given a specific feasible level $\xi' \in \Lambda$ the corresponding optimal level solution x' is the optimal solution for the quadratic programming problem $P_{\xi'}$:

$$P_{\xi'} : \begin{cases} \min \ \frac{1}{2} x^T Q x + (q + \phi(\xi')c)^T x + \phi(\xi')c_0 \\ Ax \leq b \\ d^T x = \xi' - d_0 \end{cases},$$

Being $P_{\xi'}$ strictly convex the following necessary and sufficient Karush-Kuhn-Tucker optimality conditions hold:

$$\begin{cases} Qx + q + \phi(\xi')c = A^T \mu + d\lambda & \\ d^T x = \xi' - d_0 & \\ Ax \leqq b & \textit{feasibility} \\ \mu \leqq 0 & \textit{optimality} \\ \mu^T (Ax - b) = 0 & \textit{complementarity} \\ \mu \in \Re^m, \ \lambda \in \Re & \end{cases}. \tag{1}$$

These conditions admit at least one solution (x', μ', λ'), where x' is unique due to the strict convexity of $P_{\xi'}$. Let us denote with A_i, $i \in \{1, \ldots, m\}$, the i-th row of matrix A; we can define $B' = \{i \ : \ A_i x' = b_i, \ i = 1, \ldots, m\}$ as the set of binding constraints, hence it is possible to determine the largest subset $B \subseteq B'$ such that the rows A_i, $i \in B$, and the vector d are linearly independent. It is then possible to define $N = \{1, \ldots, m\} \setminus B$ and denote with A_B and A_N the submatrices of A made by the rows in B and N, respectively; analogously, the vectors b_B, b_N, μ'_B and μ'_N can be defined too.

The Karush-Kuhn-Tucker optimality conditions become:

$$\begin{cases} Qx' \ -A_B^T \mu'_B \ -d\lambda' = -q - \phi(\xi')c \\ A_B x' \qquad\qquad\quad = b_B \\ d^T x' \qquad\qquad\quad = \xi' - d_0 \end{cases}, \tag{2}$$

which can be rewritten in the following matrix form:

$$S \begin{pmatrix} x' \\ \mu'_B \\ \lambda' \end{pmatrix} = \begin{pmatrix} -q - \phi(\xi')c \\ b_B \\ \xi' - d_0 \end{pmatrix} \quad , \quad \text{with} \ \ S = \begin{bmatrix} Q & -A_B^T & -d \\ A_B & 0 & 0 \\ d^T & 0 & 0 \end{bmatrix}.$$

Notice that the rows of $\begin{bmatrix} A_B \\ d^T \end{bmatrix}$ are linearly independent so that matrix S results to be nonsingular. This allows to show that:

$$\begin{pmatrix} x' \\ \mu'_B \\ \lambda' \end{pmatrix} = S^{-1} \begin{pmatrix} -q \\ b_B \\ -d_0 \end{pmatrix} + \xi' S^{-1} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} + \phi(\xi') S^{-1} \begin{pmatrix} -c \\ 0 \\ 0 \end{pmatrix}.$$

For the sake of simplicity, it is worth introducing the following notations:

$$\begin{pmatrix} u_x \\ u_\mu \\ u_\lambda \end{pmatrix} = S^{-1} \begin{pmatrix} -q \\ b_B \\ -d_0 \end{pmatrix} \ , \ \begin{pmatrix} v_x \\ v_\mu \\ v_\lambda \end{pmatrix} = S^{-1} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \ , \ \begin{pmatrix} w_x \\ w_\mu \\ w_\lambda \end{pmatrix} = S^{-1} \begin{pmatrix} -c \\ 0 \\ 0 \end{pmatrix},$$

so that

$$\begin{pmatrix} x' \\ \mu'_B \\ \lambda' \end{pmatrix} = \begin{pmatrix} u_x \\ u_\mu \\ u_\lambda \end{pmatrix} + \xi' \begin{pmatrix} v_x \\ v_\mu \\ v_\lambda \end{pmatrix} + \phi(\xi') \begin{pmatrix} w_x \\ w_\mu \\ w_\lambda \end{pmatrix}.$$

Let us now perform a sensitivity analysis by increasing the feasible level from ξ' up to $\xi > \xi'$, with $(\xi - \xi') > 0$ small enough to guarantee that the optimal level solutions share the same basis B. Let us denote with $(\hat{x}(\xi), \mu'_B(\xi), \lambda'(\xi))$ be the solutions of the corresponding Karush-Kuhn-Tucker necessary and sufficient optimality conditions. It results:

$$\begin{pmatrix} \hat{x}(\xi) \\ \mu'_B(\xi) \\ \lambda'(\xi) \end{pmatrix} = \begin{pmatrix} x' \\ \mu'_B \\ \lambda' \end{pmatrix} + (\xi - \xi') \begin{pmatrix} v_x \\ v_\mu \\ v_\lambda \end{pmatrix} + \big(\phi(\xi) - \phi(\xi')\big) \begin{pmatrix} w_x \\ w_\mu \\ w_\lambda \end{pmatrix}. \tag{3}$$

It is worth noticing that from $\hat{x}(\xi) = x' + (\xi - \xi')v_x + (\phi(\xi) - \phi(\xi'))\, w_x$ it yields that the optimal level solutions lie in a straight segment in the two following cases:

- $w_x = 0$,
- $\phi(\xi)$ is a linear function,

otherwise a curve is produced.

From $S[v_x, v_\mu, v_\lambda]^T = [0, 0, 1]^T$ and $S[w_x, w_\mu, w_\lambda]^T = [-c, 0, 0]^T$ it yields:

$$Qv_x = A_B^T v_\mu + dv_\lambda \tag{4}$$
$$A_B v_x = 0 \tag{5}$$
$$d^T v_x = 1 \tag{6}$$
$$Qw_x = A_B^T w_\mu + dw_\lambda - c \tag{7}$$
$$A_B w_x = 0 \tag{8}$$
$$d^T w_x = 0 \tag{9}$$

By means of simple calculations, the following implications hold:

i) $d \neq 0$ and (6) imply $v_x \neq 0$
ii) in the case $w_x \neq 0$, (6), (9) and *i)* imply that v_x and w_x are linearly independent
iii) (4), (5), (6) and Q positive definite imply $v_\lambda = v_x^T Q v_x > 0$
iv) (4), (8) and (9) imply $w_x^T Q v_x = 0$
v) (5), (6), (7) and *iii)* imply $w_\lambda = v_x^T c$

In the case $card(B) = n-1$, that is to say that A_B has $n-1$ rows and hence $\begin{bmatrix} A_B \\ d^T \end{bmatrix}$ is a square nonsingular matrix, then (8) and (9) imply $w_x = 0$, so that the optimal level solutions sharing the same basis lie in a segment (segment which belongs to an edge of the polyhedron X).

Notice also that *i)* and *ii)* imply that for all $\xi > \xi'$, with $(\xi - \xi') > 0$ small enough, it is:

$$x' \neq \hat{x}(\xi) = x' + (\xi - \xi')v_x + \big(\phi(\xi) - \phi(\xi')\big)\, w_x.$$

Procedure **LevelSolutions**(*inputs*: ξ', x'; *outputs*: $\hat{x}(\xi)$, ξ_F, ξ_O)
let $\delta > 0$, be the step parameter; determine $x'_\delta := \arg\min\{P_{\xi'+\delta}\}$;
let $B' := \{i \ : \ A_i x' = b_i, \ A_i x'_\delta = b_i, \ i = 1, \ldots, m\}$;
determine $B \subseteq B'$ such that $rank \begin{bmatrix} A_B \\ d^T \end{bmatrix} = rank \begin{bmatrix} A_{B'} \\ d^T \end{bmatrix} = card(B) + 1$;
set $S := \begin{bmatrix} Q & -A_B^T & -d \\ A_B & 0 & 0 \\ d^T & 0 & 0 \end{bmatrix}$; compute S^{-1} and:
$\begin{bmatrix} u_x \\ u_\mu \\ u_\lambda \end{bmatrix} = S^{-1} \begin{bmatrix} -q \\ b_B \\ -d_0 \end{bmatrix}$, $\begin{bmatrix} v_x \\ v_\mu \\ v_\lambda \end{bmatrix} := S^{-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$, $\begin{bmatrix} w_x \\ w_\mu \\ w_\lambda \end{bmatrix} := S^{-1} \begin{bmatrix} -c \\ 0 \\ 0 \end{bmatrix}$;
set $\xi_F := \sup\{\xi > \xi' : A\,(u_x + \epsilon v_x + \phi(\epsilon) w_x) \leqq b \ \forall \epsilon \in (\xi', \xi)\}$;
if $B = \emptyset$ *then* set $\xi_O := +\inf$
else set $\xi_O := \sup\{\xi > \xi' : (u_\mu + \epsilon v_\mu + \phi(\epsilon) w_\mu) \leqq 0 \ \forall \epsilon \in (\xi', \xi)\}$;
end if;
set $\hat{x}(\xi) := (x' + (\xi - \xi') v_x + (\phi(\xi) - \phi(\xi'))\, w_x)$;
end proc.

Clearly, the Karush-Kuhn-Tucker conditions are verified for levels $\xi \geq \xi'$ such that:

$$\begin{aligned} &\textit{feasibility conditions} : A\hat{x}(\xi) \leqq b \ , \\ &\textit{optimality conditions} : \mu'_B(\xi) \leqq 0 \ . \end{aligned}$$

In this light, the following values can be defined:

$$\begin{aligned} \xi_F &= \sup\{\xi > \xi' : A\hat{x}(\epsilon) \leqq b \ \forall \epsilon \in (\xi', \xi)\} \leq \sup\{\xi \geq \xi' : A\hat{x}(\xi) \leqq b\}, \\ \xi_O &= \sup\{\xi > \xi' : \mu'_B(\epsilon) \leqq 0 \ \forall \epsilon \in (\xi', \xi)\} \leq \sup\{\xi \geq \xi' : \mu'_B(\xi) \leqq 0\}, \end{aligned}$$

as well as the value $\xi_m = \min\{\xi_F, \xi_O\}$. Obviously, in the case $B = \emptyset$, that is to say that the optimal level solutions are interior points, it is $\xi_O = +\inf$.

Notice that for $\xi \in [\xi', \theta_m]$ both the optimality and the feasibility of $\hat{x}(\xi)$ are guaranteed, so that $\hat{x}(\xi)$ represents a set of optimal level solutions. Starting from $\hat{x}(\xi_m)$ we can iterate the process determining a new set of optimal level solutions. The objective function can be evaluated over the optimal level solutions in order to determine the global optimal solution of problem P. This leads to the use of the following function:

$$z(\xi) = f\left(\hat{x}(\xi)\right).$$

By means of the results obtained in this subsection, a procedure "*LevelSolutions*()" can then be described.

In order to find a basis B guaranteeing a value $\xi_m > \xi'$, a numerical approach is used. Specifically speaking, at the beginning of procedure "*LevelSolutions*()" an optimal level solution x'_δ (belonging to the feasible level $\xi' + \delta$) is computed, with a step parameter $\delta > 0$ small enough to guarantee that both x' and x'_δ share the same basis. Notice also that $B \subseteq B'$ can be easily determined by looking for the biggest nonsingular square submatrix of $\begin{bmatrix} A_{B'} \\ d^T \end{bmatrix}$ or by selecting iteratively $rank\left(\begin{bmatrix} A_{B'} \\ d^T \end{bmatrix}\right) - 1$ rows of $A_{B'}$ such that $rank\left(\begin{bmatrix} A_B \\ d^T \end{bmatrix}\right) = card(B) + 1$.

Notice finally that in the case $\xi_F < \xi_O$ the function $z(\xi)$ for $\xi \in [\xi_F, \xi_O]$ represents an underestimation function for the objective function $f(x)$ over the region $\{x \in X : \xi_F \leq d^T x + d_0 \leq \xi_O\}$ (see Subsection 2.4).

3.2. *Correctness and convergence*

The correctness of the proposed algorithm follows since all the feasible levels are scanned (either explicitly or implicitly) and the optimal solution, if it exists, is also an optimal level solution.

It remains to discuss the convergence (finiteness) of the algorithm, that is to say that the procedure stops after a finite number of steps. In this light, first notice that in procedure "*Visit*()" at the beginning of every iteration of the while cycle the procedure "*LevelSolutions*()" determines a new basis (set of binding constraints) which results to be different with respect to the one of the previous iteration. In particular:

- if $\xi_F \leq \xi_O$ a new binding constraint enters the basis;
- if $\xi_O < \xi_F$ one of the binding constraints leaves the basis.

Recall also that the use of a step parameter guarantees that $\xi' < \xi_m = \min\{\xi_F, \xi_O\}$. Even if every feasible level is visited just once, the same basis of binding constraints may be obtained in different nonconsecutive iterations of the while cycle. In any case, the same basis can be obtained a number of times not bigger than the finite (recall the assumptions) number of times that function $\phi(\xi)$ changes its increasing /decreasing behavior. For this very reason, taking into account that in a polyhedron there is a finite number of possible basis, every function $\phi(\xi)$ which changes its increasing /decreasing behavior a finite number of times guarantees the convergence of the method. As particular cases, notice that a monotone, or a convex/concave, or a polynomial function $\phi(\xi)$ guarantees the convergence of the method.

Finally, recall that the optimal level solution method is a simplex-like one, hence from a computational point of view it has the same complexity of the simplex algorithm.

3.3. *Underestimation function*

A key role in the computational resolution of problem P, that is in procedure "*Skip*()", will be played by the use of a proper underestimation function, that is a function $\psi(\xi)$ which verifies the following property for all the feasible levels ξ:

$$\min_{x \in X_\xi} f(x) \geq \psi(\xi).$$

In order to determine such an underestimation function the following notations can be introduced:

$$\sigma_d = d^T Q^{-1} q - d_0 \quad , \quad \sigma_c = c^T Q^{-1} q - c_0.$$

In this light, it is worth studying the objective function $f(x)$ over $\Re^n$ with just the constraint $d^T x = \xi - d_0$, instead of the whole region X. This leads to the following

strictly convex problem:

$$\begin{cases} \min \ \frac{1}{2}x^TQx + (q + \phi(\xi)c)^Tx + \phi(\xi)c_0 \\ d^Tx = \xi - d_0 \\ x \in \Re^n \end{cases}.$$

The solutions can be determined by solving the corresponding necessary and sufficient Lagrange optimality conditions. Specifically speaking it results:

$$\begin{cases} Qx + q + \phi(\xi)c = \lambda d \\ d^Tx = \xi - d_0 \\ \lambda \in \Re \end{cases},$$

so that

$$\begin{cases} x = Q^{-1}\left(-q - \phi(\xi)c + \lambda d\right) \\ d^TQ^{-1}\left(-q - \phi(\xi)c + \lambda d\right) = \xi - d_0 \end{cases}.$$

By means of simple calculations we get

$$\hat{\lambda}(\xi) = \frac{\xi + \phi(\xi)d^TQ^{-1}c + \sigma_d}{d^TQ^{-1}d},$$

$$\hat{x}(\xi) = \frac{Q^{-1}}{d^TQ^{-1}d}\left[\xi d + \phi(\xi)\left(d^TQ^{-1}c \cdot d - d^TQ^{-1}d \cdot c\right) + (\sigma_d d - q)\right].$$

The following underestimation function can then be used for $\xi \in \Lambda$:

$$\psi(\xi) = f(\hat{x}(\xi)) = \frac{1}{2}\hat{\lambda}(\xi)^2 d^TQ^{-1}d - \frac{1}{2}\phi(\xi)^2 c^TQ^{-1}c - \sigma_c\phi(\xi) - \frac{1}{2}q^TQ^{-1}q,$$

that is

$$\psi(\xi) = \frac{1}{2}\frac{(\xi+\sigma_d)^2}{d^TQ^{-1}d} + \frac{1}{2}\phi(\xi)^2\left[\frac{(d^TQ^{-1}c)^2}{d^TQ^{-1}d} - c^TQ^{-1}c\right] + \phi(\xi)(\xi+\sigma_d)\frac{d^TQ^{-1}c}{d^TQ^{-1}d} - \sigma_c\phi(\xi) - \frac{1}{2}q^TQ^{-1}q.$$

Clearly, the points $\hat{x}(\xi)$ belonging to the feasible region X of problem P are optimal level solutions corresponding to levels ξ. This happens for the levels $\xi \in \Lambda$ such that $A\hat{x}(\xi) \underline{\leq} b$; these are in general nonlinear inequalities, that can be solved analytically just for particular cases (for example when function ϕ is linear or quadratic).

3.4. *A particular case*

In this section we aim to discuss the particular case of problems P having a linear function $\phi(\xi)$, that is $\phi(\xi) = \xi$. Clearly, if $Q + cd^T + dc^T$ is a positive semidefinite matrix then problem P results to be a convex quadratic problem and can be solved by means of any method looking for a local minimum. In the case matrix $Q + cd^T + dc^T$ is not positive semidefinite the problem can be solved by specializing the results obtained in the previous sections.

First of all notice that (3) reduces to:

$$\begin{pmatrix} \hat{x}(\xi) \\ \mu'_B(\xi) \\ \lambda'(\xi) \end{pmatrix} = \begin{pmatrix} x' \\ \mu'_B \\ \lambda' \end{pmatrix} + (\xi - \xi')\begin{pmatrix} v_x + w_x \\ v_\mu + w_\mu \\ v_\lambda + w_\lambda \end{pmatrix}.$$

This points out that in this particular case the set of optimal level solutions is given by the union of segments. Moreover, it is not possible in procedure "*Visit*()" to obtain the same basis of binding constraints in two different iterations of the while cycle, thus trivially guaranteeing the convergence of the method. Values ξ_F and ξ_O can be easily computed by means of the following linear inequalities:

$$\xi_F = \sup\{\xi \geq \xi' : (\xi - \xi')(v_x + w_x) \leqq b - Ax'\},$$
$$\xi_O = \sup\{\xi \geq \xi' : (\xi - \xi')(v_\mu + w_\mu) \leqq -\mu'_B\}.$$

Functions $z(\xi)$ and $\psi(\xi)$ result to be single valued quadratic functions, so that both the minimum of $z(\xi)$ over an interval in procedure "*LevelSolutions*()" and the set $\{\xi \in \Re : \psi(\xi) \leq UB\}$ in procedure "*Skip*()" can be very easily computed in an analytical way.

Finally, it is worth noticing that in this particular case problem P can be approached also by means of branch and bound techniques (see for example [12]). In this light, notice that branch and bound methods allow to solve "small" dimension problems (due to their exponential complexity) and that they determine an "ϵ-optimal" solution with respect to a tolerance parameter $\epsilon > 0$ (needed to guarantee the convergence in a reasonable time). On the other hand, the method proposed in this paper determines the exact global optimal solution.

4. Computational test

The previously described procedures have been fully implemented with the softwares MATLAB 9.1 R2016b and Gurobi 6.5.2 on a Mac OSX computer having 16 Gb RAM and one i7 quad core processor at 3,5 GHz. For the sake of convenience, in the computational test we considered the case $\phi(\xi) = \xi$. Problems with a number of variables from $n = 10$ to $n = 100$ and a number of inequality constraints equal to $m = \lceil \frac{7}{2}n \rceil$ have been considered. The problems have been randomly created by using the "randi()" MATLAB function (deeply speaking, random integer numbers in the interval [-10,10] generated with uniform distribution). In particular, matrix $Q \in \Re^{n \times n}$ has been determined by generating a random symmetric matrix and by letting it diagonal dominant, matrix $A \in \Re^{m \times n}$, vectors $c, d, q \in \Re^n$ and scalars $c_0, d_0 \in \Re$ have been generated directly with the "randi()" function, vector $b \in \Re^m$ has been generated taking into account to guarantee the feasibility of three random vectors (deeply speaking, given three random vectors v_1, v_2, v_3 the componentwise maximum $w = max\{Av_1, Av_2, Av_3\}$ is computed and b determined as $b = w + w_0$ where w_0 is another random nonnegative vector). Clearly, the randomly created polyhedrons results to be not necessarily compact. Within the procedures, the quadratic problems have been solved with the Gurobi engine. Various instances have been randomly generated and solved. In particular, for the values of n equal to 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, we solved a number of random problems equal to 1000, 1000, 600, 300, 160, 80, 60, 40, 20, 20, respectively. The average number of iterations (that is, number of solved subproblems) and the average CPU times spent by the algorithm to solve the instances are summarized in the following table and given as the results of the test.

n	Iterations	Seconds
10	29.674	1.0862
20	83.331	4.1502
30	153.25	11.158
40	234.35	24.368
50	325.24	50.111
60	434.37	93.824
70	539.39	156.69
80	658.08	251.53
90	762.97	374.42
100	828.05	501.92

5. Conclusions

The "optimal level solutions" method has been used in the literature to solve various classes of problems. In this paper a new unifying algorithmic scheme is provided to implement it and to apply it to various classes of programs. The "optimal level solutions" method is then used to solve a wide class of rank-three problems involving linear and quadratic functions. The correctness of the method guarantees that the global minimum is found even in the case of unbounded feasible regions.

Acknowledgments Careful reviews by the anonymous referees are gratefully acknowledged.

References

[1] Cambini A. (1981) An algorithm for a special class of generalized convex programs. In: Generalized Concavity in Optimization and Economics, Academic Press, pp.491-508.

[2] Cambini A., Martein L. and C. Sodini (1983) An algorithm for two particular nonlinear fractional programs. Methods of Operations Research, vol.45, pp.61-70.

[3] Cambini A. and L. Martein (1986) A modified version of Martos' algorithm. Methods of Operation Research, vol.53, pp.33-44.

[4] Cambini A., Martein L. and S. Schaible (1989) On maximizing a sum of ratios. Journal of Information & Optimization Sciences, vol.10, pp.65-79, doi: 10.1080/02522667.1989.10698952.

[5] Cambini A. and L. Martein (1992) Equivalence in linear fractional programming. Optimization, vol.23, pp.41-51, doi: 10.1080/02331939208843743.

[6] Cambini A. and L. Martein (2009) Generalized Convexity and Optimization: theory and applications, ISBN: 978-3-540-70875-9, Lecture Notes in Economics and Mathematical Systems, Springer, Berlin, vol.616, doi: 10.1007/978-3-540-70876-6.

[7] Cambini R. (1994) A class of non-linear programs: theoretical and algorithmic results. In: Generalized Convexity, S. Komlósi, T. Rapcsák and S. Schaible (eds), Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, Berlin, vol.405, pp.294-310, doi: 10.1007/978-3-642-46802-5.

[8] Cambini R. and C. Sodini (2002) A finite algorithm for a particular d.c. quadratic programming problem. Annals of Operations Research, vol.117, pp.33-49, doi: 10.1023/A:1021509220392.

[9] Cambini R. and C. Sodini (2003) A finite algorithm for a class of nonlinear multiplicative programs. Journal of Global Optimization, vol.26, pp.279-296, doi: 10.1023/A:1023279306921.

[10] Cambini R. and C. Sodini (2007) An unifying approach to solve a class of parametrically-convexifiable problems. In: Generalized Convexity and Related Topics, I.V. Konnov, D.T. Luc and A.M. Rubinov (eds), Lecture Notes in Economics and

Mathematical Systems, Springer, Berlin, vol.583, pp.149-166, doi: 10.1007/978-3-540-37007-9_8.

[11] Cambini R. and C. Sodini (2008), A sequential method for a class of box constrained quadratic programming problems. Mathematical Methods of Operations Research, vol.67, n.2, pp.223-243, doi: 10.1007/s00186-007-0173-x.

[12] Cambini R. and C. Sodini (2008) A computational comparison of some branch and bound methods for indefinite quadratic programs. Central European Journal of Operations Research, vol.16, n.2, pp.139-152, doi: 10.1007/s10100-007-0049-4.

[13] Cambini R. and C. Sodini (2010) Global optimization of a rank-two nonconvex program. Mathematical Methods of Operations Research, vol.71, n.1, pp.165-180, doi: 10.1007/s00186-009-0289-2.

[14] Cambini R. and C. Sodini (2010) A unifying approach to solve some classes of rank-three multiplicative and fractional programs involving linear functions. European Journal of Operational Research, vol.207, pp.25-29, doi: 10.1016/j.ejor.2010.03.047.

[15] Cambini R. and C. Sodini (2012) A parametric approach for solving a class of generalized quadratic-transformable rank-two nonconvex programs. Journal of Computational and Applied Mathematics, vol.236, n.10, pp.2685-2695, doi: 10.1016/j.cam.2012.01.004.

[16] Cambini R. and C. Sodini (2014) On the minimization of a class of generalized linear functions on a flow polytope. Optimization, vol.63, n.10, pp.1449-1464, doi: 10.1080/02331934.2013.852548.

[17] Cambini R. and C. Sodini (2014) A parametric solution algorithm for a class of rank-two nonconvex programs. Journal of Global Optimization, vol.60, n.4, pp.649-662, doi: 10.1007/s10898-013-0115-5.

[18] Cambini R., Carosi L., Martein L. and E. Valipour (OnLineFirst) Simplex-like sequential methods for a class of generalized fractional programs. Mathematical Method of Operations Research, ISSN 1432-2994, doi: 10.1007/s00186-016-0556-y.

[19] Carosi L. and L. Martein (2008) A sequential method for a class of pseudoconcave fractional problems. Central European Journal of Operations Research, ISSN: 1435-246X, vol.16, n.2, pp.153-164, doi: 10.1007/s10100-007-0050-y.

[20] Ellero A. and E. Moretti (1992) A computational comparison between algorthms for linear fractional programming. Journal of Information & Optimization Sciences, vol.13, pp.343-362, doi: 10.1080/02522667.1992.10699120.

[21] Ellero A. and E. Moretti (1993) A parametric simplex-like algorithm for a fractional programming problem. Rivista di matematica per le scienze economiche e sociali, vol.16, n.2, pp.77-88, doi: 10.1007/BF02095126.

[22] Ellero A. (1996) The optimal level solutions method. Journal of Information & Optimization Sciences, vol.17, pp.355-372, doi: 10.1080/02522667.1996.10699287.

[23] Frenk J.B.G. and S. Schaible (2005) Fractional programming. In: Handbook of Generalized Convexity and Generalized Monotonicity, Nonconvex Optimization and Its Applications, vol.76, pp.335-386, Springer, New York, doi: 10.1007/b101428.

[24] Konno H. and T. Kuno (1995) Multiplicative programming problems. In: Handbook of Global Optimization, R. Horst and P.M. Pardalos (eds), Nonconvex Optimization and Its Applications, Kluwer Academic Publishers, Dordrecht, vol.2, pp.369-405, doi: 10.1007/978-1-4615-2025-2.

[25] Merlone U. (2000) The optimal level solutions method applied to a non linear programming problem with exponential objective function. Journal of Information & Optimization Sciences, vol.21, pp.305-321, doi: 10.1080/02522667.2000.10699454.

[26] Schaible S. and C. Sodini (1995) Finite algorithm for generalized linear multiplicative programming. Journal of Optimization Theory and Applications, vol.87, n.2, pp.441-455, doi: 10.1007/BF02192573.

[27] Sodini C. (1986) Minimizing the sum of a linear function and the square root of a convex quadratic form. Methods of Operations Research, vol.53, pp.171-182.

[28] Sodini C. (1990) Equivalence and parametric analysis in linear fractional programming. In: Generalized Convexity and Fractional Programming with Applications, Lecture Notes in Economics and Mathematical Systems, vol.345, Springer, pp.143-154, doi: 10.1007/978-3-642-46709-7_11.