OPSEARCH, Vol. 45, No.2, 2008 0030-3887/00 $ 5.00+0.00

Revisiting Dinkelbach-Type Algorithms for Generalized Fractional Programs

Jean-Pierre Crouzeix,

LIMOS, CNRS-UMR 6158, Université Blaise Pascal, Clermont-Ferrand, France,

Jacques A. Ferland,

DIRO, Université de Montréal, Montréal, Canada,

Van Hien Nguyen,

Département de mathématiques, Facultés Universitaires Notre- Dame-de-la-Paix, √

October 2004

First Revision April 2006

Second Revision August 2006

Abstract

In this paper we introduce a new Dinkelbach-type algorithm where the new iterate is determined using the information given by previous iterates and not only by the last one. The new algorithm is compared numerically with previous algorithms for generalized fractional programs.

Key words

Fractional programming, Dinkelbach algorithms, interval methods, parameterized problem.

AMS Classification

90C32, 90C26.

* This research was supported by NSERC grant (OGP 0008312) from Canada

Paper received in November 2004

1. Introduction

A generalized fractional program consists in finding an optimal solution x* of the problem

$$\lambda^* = \min_{x} \left[\max_{1\le i\le p} \frac{f_i(x)}{g_i(x)} : x \in C \right] \qquad \text{(GFP)}$$

Where $\Phi \neq C \subset R^n$, $f_i : C \to R$, and $g_i : C \to (0, +\infty)$ for $i = 1, \cdots, p$. To simplify the presentation, we assume that (GPF) has an optimal solution.
When p = 1, one speaks of a fractional program, and the term generalized fractional program stands for the case p > 1. If C is convex, if the functions f_i are convex, and if the functions g_i are concave, then the problem is called a generalized convex fractional program. Similarly, if C is a polyhedral convex set and the functions f_i and g_i are affine, then the problem is called a generalized linear fractional program. Charnes and Cooper [12] introduce a variable transformation that can be used to reduce a convex (linear) fractional program to a convex (linear) program.

The case p=1, Fractional Programming

Dinkelbach [11] introduced his original algorithm designed for the case p = 1. It relies on the family of problems parameterized by $\lambda \in R$,

$$F(\lambda) = \inf_{x} \, [f(x) - \lambda g(x) : x \in C] \qquad (P_\lambda)$$

Note that to reduce notation, we omit the subscript "1" in presenting this case, and thus f and *g* stand for f_1 and g_1. By construction, F is concave, non increasing and upper semi continuous. Then, $F(\lambda^*) = 0$, $F(\lambda) < 0$ if $\lambda > \lambda^*$, and (GFP) and (P_{λ^*}) share the same set of optimal solutions. If in addition, (P_λ) has an optimal solution for some $\lambda < \lambda^*$, then $F(\lambda) > 0$ for all $\lambda < \lambda^*$. Thus the problem of finding λ^* consists in searching for the root of the equation of one variable $F(\lambda) = 0$.
It is clear that if x_λ is an optimal solution of (P_λ), then for all μ

$$F(\mu) \le f(x_\lambda) - \mu g(x_\lambda) = F(\lambda) - (\mu - \lambda) g(x_\lambda). \qquad (1)$$

Hence $-g(x_\lambda)$ belongs to the supdifferential of F at λ. In particular, since $F(\lambda^*) = 0$,

$$0 \le F(\lambda) - (\lambda^* - \lambda) g(x_\lambda) \qquad (2)$$

and

$$\lambda^* \le \lambda - \frac{F(\lambda)}{-g(x_\lambda)} = \frac{f(x_\lambda)}{g(x_\lambda)}. \tag{3}$$

Note that $F'(\lambda) = -g(x_\lambda)$ if F is differentiable at λ. The Dinkelbach algorithm is the following straightforward adaptation of the Newton method to a non differentiable context: the supgradient $-g(x_\lambda)$ replaces $F'(\lambda)$. The algorithm is as follows:

The one-ratio Dinkelbach algorithm

- **Initialization.**
  - Start with some $x_0 \in C$.
  - Take $\lambda_0 := \dfrac{f(x_0)}{g(x_0)}$, and k = 1.
- **Step k.**
  - Solve the problem

    $$t_k = \min_x [f(x) - \lambda_{k-1} g(x) : x \in C]. \qquad (P_\lambda)$$

    Let x_k be an optimal solution of this problem.
  - *Stopping rule.* If $t_k = 0$, then STOP: $\lambda^* = \lambda_{k-1}$ and x_k is an optimal solution of (GFP).
  - Otherwise, let $\lambda_k := \dfrac{f(x_k)}{g(x_k)}$. Let $k := k+1$, and go back to Step k.

By construction, the algorithm generates a sequence $\{x_k\} \subset C$ and a strictly decreasing sequence $\{\lambda_k\}$. It inherits a high speed of convergence from the Newton method, but because *F* is not differentiable in general, the convergence in the worst case is only superlinear (see Benadada et al. [2] for instance). Still, due to the concavity of F, the speed of convergence is better than the one of other methods finding the root of an equation of one real variable, like interval methods (dichotomy or secant methods for instances).

Generalized Fractional Program, the case p > 1

The Dinkelbach approach is generalized to the multi-ratio case in Crouzeix et al. [6] and [5]. Given a weight vector w > 0, the function F is replaced by

$$F_w(\lambda) = \min_x \left[\max_{1 \le i \le p} \frac{f_i(x) - \lambda g_i(x)}{w_i} : x \in C \right], \tag{4}$$

$$= \min_{x,t} [t : f_i(x) - \lambda g_i(x) - tw_i \le 0,\ 1 \le i \le p : x \in C, t \in R]. \tag{5}$$

Here again, F_w is non increasing and upper semi continuous, but not concave anymore. If (GFP) has an optimal solution, then $F_w(\lambda^*) = 0$ and $F_w(\lambda) < 0$ if $\lambda > \lambda^*$. If in addition problem (4) (or equivalently (5)) has an optimal solution for some $\lambda < \lambda^*$, then $F_w(\lambda) > 0$ for all $\lambda < \lambda^*$. These results are proved in details by Crouzeix et al. [6] for the case where $w_i = 1$ for all $1 \le i \le p$. The extension for any vector w > 0 is immediate (see Crouzeix et al. [5]).

Before going any further, it is worth to note the nice properties of the problems (4) and (5). Indeed, if C is convex, if the functions f_i are convex and if the functions g_i are concave, then the problems (4) and (5) are convex whenever $\lambda \ge 0$. Furthermore if C is a polyhedral convex set and if the functions f_i and g_i are affine, then (4) and (5) are linear programs.

As in the one-ratio case, the problem reduces to searching for the root of one equation of one real variable. The multi-ratio Dinkelbach algorithm is as follows (Crouzeix et al. [6]):

The multi-ratio Dinkelbach algorithm

- **Initialization.**
  - Start with some $x_0 \in C$.
  - Take $\lambda_0 := \max_{1 \le i \le p} \frac{f_i(x_0)}{g_i(x_0)}$. Choose $w^0 > 0$. Let k = 1.
- **Step k.**
  - Solve the problem

$$\min_{x,t} [t : f_i(x) - \lambda_{k-1} g_i(x) - tw_i^{k-1} \le 0, 1 \le i \le p : x \in C, t \in R]. \tag{P_k}$$

Let (x_k, t_k) be an optimal solution of this problem.

- *Stopping rule.* If $t_k = 0$, then STOP: $\lambda^* = \lambda_{k-1}$ and x_k is an optimal solution of (GFP).
- Otherwise, let $\lambda_k := \max_{1 \le i \le p} \frac{f_i(x_k)}{g_i(x_k)}$. Choose $w^k > 0$. Let $k := k+1$, and go back to Step k.

From the definition of λ_{k-1}, it follows that $\max_{1 \le i \le p} [f_i(x_{k-1}) - \lambda_{k-1} g_i(x_{k-1})] = 0$. Hence, since x_{k-1} is a feasible solution of (P_k), then $t_k \le 0$ and $\lambda^* \le \lambda_k \le \lambda_{k-1}$.

Crouzeix et al. [6] introduce the first version of this algorithm referred here as MAX, where $w_i^k = 1$ for all i and k. The convergence is then only linear. A more efficient version introduced by Crouzeix et al. [5], referred as MAXMOD, consists in updating w at step k by taking $w_i^k = g_i(x_k)$ for all i. Then, the convergence becomes superlinear under suitable assumptions (see Borde and Crouzeix [4] for instance). The correction algorithm for discrete fractional l_∞ − approximation (see Barrodale et al. [10], and Flachs [8]) is related to MAXMOD. A combination of MAXMOD with the proximal-method is introduced by Gugat [1].

Because F_w is not concave in general, the MAX and MAXMOD algorithms do not enjoy the high speed of the one-ratio Dinkelbach algorithm for the case p = 1. As a consequence, interval methods searching for the root of equation $F_w(\lambda) = 0$ become competitive. Such methods have been introduced by Bernard and Ferland [3] and Ferland and Potvin [7]. We shall compare numerically these methods with the new one introduced in the next section.

2. A new variant of Dinkelbach algorithm

In the new multi-ratio Dinkelbach algorithm, a sequence of values μ_k of λ is generated according to an approach similar to the one used to generate the sequence of λ_k in the multi-ratio Dinkelbach algorithm, but using the information given by several previous iterates rather than given by only the last one. The procedure is also initialized with some $x_0 \in C$, and

$$\mu_0 := \max_{1 \le i \le p} \frac{f_i(x_0)}{g_i(x_0)}.$$

At Step k of the procedure, we solve the parameterized problem $(\overline{P}_k)$

$$\min_{x,t} [t : f_i(x) - \mu_{k-1} g_i(x) - tw_i^{k-1} \leq 0, 1 \leq i \leq p : x \in C, t \in R]. \qquad (\overline{P}_k)$$

Let (x_k, t_k) be an optimal solution of this problem. As long as $t_k \leq 0$, the procedure evolves as the multi-ratio Dinkelbach algorithm. But whenever $t_k > 0$, then the procedure is restarted using x_{k-1} or x_k (see below) as a new initial solution x_0.

A lower bound lb is also introduced to reduce computational effort by eliminating the resolution of $(\overline{P}_k)$ in some cases. The value of lb is initialised at a value small enough to have lb $\leq \lambda^*$. For instances, lb can take a very small value close to $-\infty$, or a better value like 0 if $f_i \geq 0$, for $i = 1, \cdots, p$. The value of lb is updated using the different values of μ_{k-1} generated on the left-hand side of λ^* whenever $t_k > 0$.

Before we go any further, we summarize the new procedure, and further justifications are given afterward.

The new multi-ratio Dinkelbach algorithm

- **Initialization.**
  - Start with some $x_0 \in C$ and some $lb < \lambda^*$.
- **Step 0.**
  - Take $\mu_0 := \max_{1 \leq i \leq p} \frac{f_i(x_0)}{g_i(x_0)}$. Let $k = 1$.
- **Step k.**
  - Solve the problem

    $$\min_{x,t} [t : f_i(x) - \mu_{k-1} g_i(x) - tw_i^{k-1} \leq 0, 1 \leq i \leq p : x \in C, t \in R]. \qquad (\overline{P}_k)$$

    Let (x_k, t_k) be an optimal solution of this problem, and

    $$\overline{\lambda}_k := \max_{1 \leq i \leq p} \frac{f_i(x_k)}{g_i(x_k)}.$$

  - *Stopping rule.* If $t_k = 0$, then STOP: $\lambda^* = \mu_{k-1}$ and x_k is an optimal solution of (GFP).
  - IF $t_k > 0$ (then $lb < \mu_{k-1} < \lambda^*$):
    - Let $lb := \mu_{k-1}$.
    - IF $\overline{\lambda}_k < \overline{\lambda}_{k-1}$, then $x_0 := x_k$ and go to Step 0.
    - IF $\overline{\lambda}_k \geq \overline{\lambda}_{k-1}$, then $x_0 := x_{k-1}$ and go to Step 0.

- IF $t_k < 0$ (then $\lambda^* < \mu_{k-1} < \bar{\lambda}_{k-1}$):
  - Let $\mu_k := \max_{1\le i\le p} \min_{0\le j\le k} \frac{f_i(x_j)}{g_i(x_j)}$
  - IF $\mu_k \le lb$, then $x_0 := x_k$ and go to Step 0.
  - Otherwise, choose $w^k > 0$. Let $k := k+1$, and go back to Step k.

Now we analyze more closely the general Step k. First, it is interesting to note the similarity between $\bar{\lambda}_k := \max_{1\le i\le p} \frac{f_i(x_k)}{g_i(x_k)}$ and the value λ_k generated in the multi-ratio Dinkelbach algorithm using the parameterized problem (P_k). Assume that during the last (k – 1) iterations, $t_j < 0$, $1 \le j \le (k-1)$, and suppose that $t_k < 0$. Let

$$\mu_k := \max_{1\le i\le p} \min_{0\le j\le k} \frac{f_i(x_j)}{g_i(x_j)}.$$

Since
$$\max_{1\le i\le p} \min_{0\le j\le k} \frac{f_i(x_j)}{g_i(x_j)} \le \min_{0\le j\le k} \max_{1\le i\le p} \frac{f_i(x_j)}{g_i(x_j)},$$

it follows that
$$\mu_k = \max_{1\le i\le p} \min_{0\le j\le k} \frac{f_i(x_j)}{g_i(x_j)} \le \min_{0\le j\le k} \max_{1\le i\le p} \frac{f_i(x_j)}{g_i(x_j)} \le \bar{\lambda}_k.$$

Furthermore, since we assume that $t_j < 0$, $1 \le j \le (k-1)$, then

$$\mu_j \le \bar{\lambda}_j \qquad 1 \le j \le k.$$

Since we suppose that $t_k < 0$, then referring to problem $(\bar{P}_k)$, it follows that

$$f_i(x_k) - \mu_{k-1} g_i(x_k) < 0, \qquad 1 \le i \le p.$$

Thus
$$\mu_{k-1} > \frac{f_i(x_k)}{g_i(x_k)}, \qquad 1 \le i \le p,$$

and hence $$\mu_{k-1} > \max_{1\le i\le p} \frac{f_i(x_k)}{g_i(x_k)} = \bar{\lambda}_k .$$

Then we can conclude that as long as $t_j < 0$, $1 \le j \le k$,

$$\lambda^* \le \mu_k \le \bar{\lambda}_k < \mu_{k-1} \le \bar{\lambda}_{k-1} < \mu_{k-2} \le \cdots \le \bar{\lambda}_1 < \mu_0 .$$

Thus μ_k remains on the right - hand side of λ^*, and it is "closer" to λ^* than $\bar{\lambda}_k$.

Unfortunately, nothing prevents t_k to be positive. In such a situation, $\mu_{k-1} < \lambda^*$, and μ_{k-1} is a lower bound on λ^*. Hence the value μ_{k-1} can be used to update the best lower bound lb found so far during the execution of the procedure. Furthermore, whenever $t_k > 0$ or $\mu_k <$ lb, the procedure is restarted with a new initial solution x_0 selected differently according to the following cases:

Case 1: If $\mu_k <$ lb, then we do not solve $(\bar{P}_{k+1})$ since we know that its optimal value is positive. In this case, $x_0 := x_k$.

Case 2: If $\mu_{k-1} \ge$ lb but $t_k > 0$, then lb $< \mu_{k-1} < \lambda^*$. Hence $\text{lb} := \mu_{k-1}$. To determine x_0 we consider the solution x_{k-1} and x_k, and the corresponding values $\bar{\lambda}_{k-1}$ and $\bar{\lambda}_k$. If $\bar{\lambda}_{k-1} > \bar{\lambda}_k$ (i.e., $\bar{\lambda}_k$ is closer to λ^* than $\bar{\lambda}_{k-1}$),then $x_0 := x_k$. Otherwise $x_0 := x_{k-1}$.

It is easy to see that reinitializing the procedure according to this strategy induces a decreasing sequence of values for λ converging to λ^* like in the original multi-ratio Dinkelbach algorithm.

As for the previous algorithm, we consider two choices for updating the weight vector w. In the REST algorithm, $w_i^k = 1$ for all k and i, and in the RESTMOD algorithm, $w_i^k = g_i(x_k)$ for all k and i.

3.Generalized linear fractional programming

A generalized linear fractional programming problem is of the form:

$$\lambda^* = \inf_x \max_{1\le i\le p} \left[\frac{a_{i.}^t x + \alpha_i}{b_{i.}^t x + \beta_i} : Ex \le \gamma,\ x \ge 0 \right] \qquad \text{(GLFP)}$$

where $a_{i.}$ and $b_{i.}$ denote rows i of the $p \times n$ matrices A and B, respecively, α and β belong to R^p, E is a $m \times n$ matrix, and $\gamma \in R^m$.

We make the following assumptions:

(H1) **Feasibility assumption:** there exists $\bar{x} \geq 0$ such that $E\bar{x} \leq \gamma$.

(H2) **Positivity assumption:** $B > 0$ and $\beta > 0$.

(H3) **Optimality assumption:** (GLFP) has optimal solutions.

Now we introduce a specific version of the new multi-ratio Dinkelbach algorithm where we take advantage of the fact that the functions are linear to update the lower bound lb on λ^* not only when the optimal value t_k of $(\bar{P}_k)$ is positive , but even when t_k is negative. In order to do this, we introduce the following analysis where for any weight vector $w > 0$, the parameterized problems in $\lambda \in R$ are denoted

$$F_w(\lambda) = \inf_{x,t} \left[t : -Ex \geq -\gamma,\ (\lambda B - A)x + tw \geq \alpha - \lambda\beta : x \geq 0, t \in R \right]. \qquad (\mathrm{Pr}_\lambda)$$

These problems are linear, and their duals are as follows:

$$G_w(\lambda) = \sup\left[\ (\alpha - \lambda\beta)^t y - \gamma^t z : w^t y = 1,\ (\lambda B^t - A^t)y - E^t z \leq 0,\ y \geq 0,\ z \geq 0\ \right]. \quad (\mathrm{Du}_\lambda)$$

If both problems (Pr_λ) and (Du_λ) are feasible, then both have optimal solutions and the equality $F_w(\lambda) = G_w(\lambda)$ holds. In view of (H3),

$$F_w(\lambda^*) = G_w(\lambda^*) = 0 \quad \text{and} \quad \lambda^* = \max\left[\lambda : G_w(\lambda) \geq 0\right].$$

Since $G_w(\lambda)$ is non negative if and only if there exists (y, z) such that

$$y \geq 0,\ z \geq 0,\ w^t y = 1$$

$$\frac{\alpha^t y - \gamma^t z}{\beta^t y} \geq \lambda \quad \text{and} \quad A^t y + E^t z \geq \lambda B^t y,$$

it follows that, according to Borde and Crouzeix [4] and Crouzeix et al. [9],

$$\lambda^* = \sup_{y,z}\left(\min\left[\frac{\alpha^t y-\gamma^t z}{\beta^t y},\ \min_{1\le l\le n}\frac{a_{\cdot l}^t y+e_{\cdot l}^t z}{b_{\cdot l}^t y}\right]: w^t y=1,\ y\ge 0,\ z\ge 0\right) \quad (d_w)$$

where $a_{\cdot l}$, $b_{\cdot l}$, and $e_{\cdot l}$ denote columns l of matrices A, B, and E, respectively. It is interesting to note that (d_w) is also a generalized linear fractional program.

Now assume that (x_λ, t_λ) is an optimal solution of (Pr_λ) and that (y_λ, z_λ) is an optimal solution of (Du_λ). Then,

$$t_\lambda = F_w(\lambda) = G_w(\lambda) = (\alpha-\lambda\beta)^t y_\lambda - \gamma^t z_\lambda, \qquad A^t y_\lambda + E^t z_\lambda \ge \lambda B^t y_\lambda.$$

Hence,

$$\frac{\alpha^t y_\lambda-\gamma^t z_\lambda}{\beta^t y_\lambda}-\frac{t_\lambda}{\beta^t y_\lambda} = \lambda \le \min_{1\le l\le n}\frac{a_{\cdot l}^t y_\lambda+e_{\cdot l}^t z_\lambda}{b_{\cdot l}^t y_\lambda}. \quad (6)$$

Note also that (y_λ, z_λ) is a feasible solution for (d_w) and $\beta^t y_\lambda > 0$. Consider the two following cases :

- $t_\lambda < 0$.

  Then, $\lambda^* < \lambda$ and $\frac{\alpha^t y_\lambda-\gamma^t z_\lambda}{\beta^t y_\lambda} < \lambda \le \min_{1\le l\le n}\frac{a_{\cdot l}^t y_\lambda+e_{\cdot l}^t z_\lambda}{b_{\cdot l}^t y_\lambda}$.

  It follows from (6) that

$$\lambda+\frac{t_\lambda}{\beta^t y_\lambda} < \lambda^* < \lambda. \quad (7)$$

- $t_\mu > 0$.

  Then it follows from (6) that

$$\lambda \le \min\left(\lambda + \frac{t_\lambda}{\beta^t y_\lambda},\ \min_{1\le l\le n} \frac{a_{\cdot l}^t y_\lambda + e_{\cdot l}^t z_\lambda}{b_{\cdot l}^t y_\lambda} \right) \le \lambda^*. \tag{8}$$

In both cases, we obtain a lower bound for λ^* that can be used to update lb. Furthermore, since the simplex algorithm applied to problem (Pr_λ) (or (Du_λ)) generates at the same time, an optimal solution for (Pr_λ) and also for (Du_λ), then the required information to evaluate this lower bound is available. Therefore, we modify the algorithm of the last section as follows.

The new linear multi-ratio Dinkelbach algorithm

- **Initialization.**
  - Start with some $x_0 \in \{x : Ex \le \gamma,\ x \ge 0\}$ and $lb < \lambda^*$.
- **Step 0.**
  - Take $\mu_0 := \max_{1\le i\le p} \dfrac{f_i(x_0)}{g_i(x_0)}$. Choose $w^0 > 0$. Let k = 1.
- **Step k.**
  - Solve the problem $(Pr_{\mu_{k-1}})$.

    Let (x_k, t_k) be an optimal solution of this problem, and (y_k, z_k) an optimal solution of its dual $(Du_{\mu_{k-1}})$. Let

    $$\bar\lambda_k := \max_{1\le i\le p} \frac{f_i(x_k)}{g_i(x_k)} .$$
  - *Stopping rule.* If $t_k = 0$, then STOP: $\lambda^* = \mu_{k-1}$ and x_k is an optimal solution of (GLFP).
  - IF $t_k > 0$ (then $lb < \mu_{k-1} < \lambda^*$):
    - Let

      $$lb := \min\left(\mu_{k-1} + \frac{t_k}{\beta^t y_k},\ \min_{1\le l\le n} \frac{a_{\cdot l}^t y_k + e_{\cdot l}^t z_k}{b_{\cdot l}^t y_k} \right).$$
    - IF $\bar\lambda_k < \bar\lambda_{k-1}$, then $x_0 := x_k$ and go to Step 0.
    - IF $\bar\lambda_k \ge \bar\lambda_{k-1}$, then $x_0 := x_{k-1}$ and go to Step 0.
  - IF $t_k < 0$ (then $\lambda^* < \mu_{k-1} < \bar\lambda_{k-1}$):
    - Let

$$\mu_k := \max_{1\le i\le p} \min_{0\le j\le k} \frac{f_i(x_j)}{g_i(x_j)} \quad \text{and} \quad lb := \max\left(lb,\ \mu_{k-1} + \frac{t_k}{\beta^t y_k} \right).$$

- IF $\mu_k \le lb$, then $x_0 := x_k$ and go to Step 0.
- Otherwise, choose $w^k > 0$. Let $k := k+1$, and go back to Step k.

Finally, it is worthy of noting that the lower bound lb specified in (6) has an important practical impact since when any Dinkelbach-type algorithm is implemented numerically, the stopping rule

$$\text{"If } t_k = 0 \ldots\text{"}$$

is in fact replaced by

$$\text{"If} \left| t_k \right| < \varepsilon \ldots\text{"} \quad .$$

Thus the value of the lower bound lb specified in (8) allows evaluating the distance of

μ_{k-1} (or λ_{k-1}) from λ^* since

$$\mu_{k-1} - \lambda^* \le \mu - lb.$$

Furthermore, the value of lb specified in (9) justifies a posteriori the stopping rule $\text{"If} \left| t_k \right| < \varepsilon \ldots\text{"}$. Indeed, if $\left| t_k \right| < \varepsilon$, then

$$\mu_{k-1} - \lambda^* \le \mu_{k-1} - lb \le \mu_{k-1} - \mu_{k-1} - \frac{t_k}{\beta^t y_k}$$

and

$$\mu_{k-1} - \lambda^* \le \frac{\varepsilon}{\beta^t y_k}.$$

4. Numerical experiments

The new algorithms REST and RESTMODM are compared numerically with the algorithms MAX and MAXMODM and with two interval-type algorithms NEWMAX and NEWMODM introduced in Bernard and Ferland [3] and Ferland and Potvin [7]. The test problems used in the numerical tests are generalized linear

fractional programs similar to those used in Ferland and Potvin [7]. Hence the subproblems (P_k) are linear programs that are solved with the software CPLEX.
Problems with n = 10, 20, 30, 50, 70, and 100 variables and p = 10 and 20 ratios are generated randomly, The elements a_{ij} and α_i are random numbers in [−50, 50], and elements b_{ij} and β_i are random numbers in [0, 50] and [0.01, 50], respectively. Five different types of feasible domains are considered:

$$X1 = \left\{ x : \sum_{j=1}^{n} x_j = 1 \ ; \ x_j \geq 0, \ 1 \leq j \leq n \right\}$$

$$X2 = \left\{ x : \sum_{j=1}^{n} x_j \geq 2 \ ; \ \sum_{j=1}^{n} x_j \leq 5 \ ; \ x_j \geq 0, \ 1 \leq j \leq n \right\}$$

$$X3 = \left\{ x : \sum_{j=1}^{n} x_j \geq 10 \ ; \ \sum_{j=1}^{n} x_j \leq 30 \ ; \ 0 \leq x_j \leq 20, \ 1 \leq j \leq n \right\}$$

$$X4 = \left\{ x : \sum_{j=1}^{n} x_j \geq 35 \ ; \ 0 \leq x_j \leq 30, \ 1 \leq j \leq n \right\}$$

$$X5 = \left\{ x : 0 \leq x_j \leq 10, \ 1 \leq j \leq n \right\}.$$

The theoretical stopping rule t = 0 is changed to $t < \varepsilon$, and numerical tests are completed using three different values for $\varepsilon : \varepsilon_1 = 0.01, \varepsilon_2 = 0.0001$, and $\varepsilon_3 = 0.000005$. Thus the algorithms are compared on 180 problems (one for each feasible domain, and each value of n, p and ε).
The results are summarized in Table 5.1 and 5.2 where each figure is the average number of iterations required by the technique associated with the row to solve the set of problems of the corresponding column. As expected, the variant where $w_i^k = g_i(x_k)$ for all k and i is more efficient than the corresponding variant where $w_i^k \equiv 1$ for all k and i. Furthermore the new algorithms REST and RESTMODM are more efficient than MAX and MAXMODM, respectively but the interval-type algorithms remain more efficient. The overall ranking of the techniques in decreasing order of efficiency is as follows; NEWMODM, RESTMODM, MAXMODM, NEWMAX, REST, MAX.

Algorithm	n						m	
	10	20	30	50	70	100	10	20
MAX	12.80	10.20	10.73	12.80	13.30	11.47	11.33	12.37
NEWMAX	9.87	9.57	7.63	8.00	8.97	10.23	10.63	7.40
REST	10.17	8.83	10.00	11.80	12.93	10.97	10.83	10.70
MAXMODM	7.27	6.50	7.27	7.40	7.77	6.83	7.13	7.10
NEWMODM	5.93	5.13	5.53	5.57	6.00	5.40	5.60	5.53
RESTMODM	6.73	6.10	6.93	7.03	7.10	6.50	6.63	6.77

Table 5.1 Average number of iterations for the subsets of problems associated with the columns

Algorithm	Domain types					Epsilon			Overall
	X1	X2	X3	X4	X5	0.01	0.0001	0.000005	average
MAX	10.73	11.80	11.77	12.97	11.97	9.70	12.60	13.20	11.80
NEWMAX	8.57	10.37	8.83	9.37	7.87	7.10	9.60	10.30	9.00
REST	8.63	8.87	11.47	12.07	12.67	8.30	11.10	12.90	10.70
MAXMODM	5.87	6.27	7.23	7.83	8.43	6.70	7.20	7.50	7.10
NEWMODM	5.07	5.00	5.43	6.03	6.23	5.00	5.60	6.00	5.50
RESTMODM	5.73	5.80	6.70	7.07	8.13	6.20	6.90	7.10	6.60

Table 5.2 Average number of iterations for the subsets of problems associated with the columns.

5. Conclusion

In this paper, we introduce a new Dinkelbach-type algorithm improving the choice of λ_k at each iteration. Numerical results indicate that for generalized linear fractional programs, these new algorithms REST and RESTMODM are more efficient than MAX and MAXMODM, respectively, but that they remain less efficient than interval-type algorithms.

6. References

(1) Gugat, M. (1998), "Prox-regularization Methods for Generalized Fractional Programming", Journal of Optimization Theory and Applications, 99, 691–722.

(2) Benadada, Y., Crouzeix, J.- P., Ferland, J.A. (1993), "Rate of Convergence of a Generalization of Newton's Method", Journal of Optimization Theory and Applications, 78, 599–604.

(3) Bernard, J.C., Ferland, J.A. (1989), "Convergence of Interval-Type Algorithms for Generalized Fractional Programming", Mathematical Programming, 43, 349–363.

(4) Borde, J., Crouzeix, J.- P. (1987), “Convergence of a Dinkelbach-type Algorithm in Generalized Fractional Programming”, Zeitschrift für Operations Research 31, 3–54.

(5) Crouzeix, J.- P., Ferland, J.A., Schaible, S. (1986), “A Note on an Algorithm for Generalized Fractional Programs”, Journal of Optimization Theory and Applications, 50, 183–187.

(6) Crouzeix, J.- P., Ferland, J.A., Schaible, S. (1985), “An Algorithm for Generalized Fractional Programs”, Journal of Optimization Theory and Applications, 47, 35–49.

(7) Ferland, J.A., Potvin,J.Y. (1985), “Generalized Fractional Programming : Algorithm and Numerical Experimentation”, European Journal of Operational Research, 20, 92–101.

(8) Flachs, J. (1985), “Generalized Cheney-Loeb-Dinkelbach-Type Algorithms”, Mathematics of Operations Research, 10, 674–687.

(9) Crouzeix, J.- P., Ferland, J.A., Schaible, S. (1983), “Duality in Generalized Linear Fractional Programming”, Mathematical Programming, 27, 342–354.

(10) Barrodale, I., Powell, M.J.D., Roberts, F.D.K. (1972), “The Differential Correction Algorithm for Fractional l_∞ Approximation”, SIAM Journal Numerical Analysis, 9, 493–504.

(11) Dinkelbach, W. (1967), “On Nonlinear Fractional Programming”, Management Science, 13, 492–498.

(12) Charnes, A., Cooper, W.W. (1962), Programming with Linear Fractional Functionals”, Naval Research Logistics Quarterly, 9, 181–186.