Generalized Fractional Programming With User Interaction

S¸. I. Birbil, J. B. G. Frenk and S. Zhang

ERIM REPORT SERIES *RESEARCH IN MANAGEMENT*	
ERIM Report Series reference number	ERS-2004-033-LIS
Publication	May 2004
Number of pages	27
Email address corresponding author	sibirbil@few.eur.nl
Address	Erasmus Research Institute of Management (ERIM) Rotterdam School of Management / Rotterdam School of Economics Erasmus Universiteit Rotterdam P.O. Box 1738 3000 DR Rotterdam, The Netherlands Phone: +31 10 408 1182 Fax: +31 10 408 9640 Email: info@erim.eur.nl Internet: www.erim.eur.nl

Bibliographic data and classifications of all the ERIM reports are also available on the ERIM website:
www.erim.eur.nl

ERASMUS RESEARCH INSTITUTE OF MANAGEMENT

REPORT SERIES
RESEARCH IN MANAGEMENT

BIBLIOGRAPHIC DATA AND CLASSIFICATIONS		
Abstract	The present paper proposes a new approach to solve generalized fractional programming problems through user interaction. Capitalizing on two alternatives, we review the Dinkelbach-type methods and set forth the main difficulty in applying these methods. In order to cope with this difficulty, we propose an approximation approach that can be controlled by a predetermined parameter. The proposed approach is promising particularly when a decision maker is involved in the solution process and agrees upon finding an effective but nearoptimal value in an efficient manner. The decision maker is asked to decide the parameter and our analysis shows how good is the value found by the approximation corresponding to this parameter. In addition, we present several observations that may be suitable for boosting up the performance of the proposed approach. Finally, we support our discussion through extensive numerical experiments.	
Library of Congress Classification (LCC)	5001-6182	Business
	5201-5982	Business Science
	HB 143	Mathematical Programming
Journal of Economic Literature (JEL)	M	Business Administration and Business Economics
	M 11 R 4	Production Management Transportation Systems
	C 61	Optimization Techniques, programming
European Business Schools Library Group (EBSLG)	85 A	Business General
	260 K 240 B	Logistics Information Systems Management
	255 B	Programming
Gemeenschappelijke Onderwerpsontsluiting (GOO)		
Classification GOO	85.00	Bedrijfskunde, Organisatiekunde: algemeen
	85.34 85.20	Logistiek management Bestuurlijke informatie, informatieverzorging
	31.80	Toepassingen van de wiskunde
Keywords GOO	Bedrijfskunde / Bedrijfseconomie	
	Bedrijfsprocessen, logistiek, management informatiesystemen	
	Wiskundige programmering, gebruikersinterfaces, optimalisatie	
Free keywords	Generalized fractional programming, user interaction, approximation approach, error analysis, performance improvement	

GENERALIZED FRACTIONAL PROGRAMMING WITH USER INTERACTION

Ş. İ. Birbil[†], J. B. G. Frenk[‡] and S. Zhang[♮*]

[†]*Erasmus Research Institute of Management, Erasmus University Rotterdam*
Postbus 1738, 3000 DR Rotterdam, The Netherlands.

[‡]*Econometrics Institute, Erasmus University Rotterdam*
Postbus 1738, 3000 DR Rotterdam, The Netherlands.

[♮]*Systems Engineering and Engineering Management*
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong.

ABSTRACT. The present paper proposes a new approach to solve generalized fractional programming problems through user interaction. Capitalizing on two alternatives, we review the Dinkelbach-type methods and set forth the main difficulty in applying these methods. In order to cope with this difficulty, we propose an approximation approach that can be controlled by a predetermined parameter. The proposed approach is promising particularly when a decision maker is involved in the solution process and agrees upon finding an effective but near-optimal value in an efficient manner. The decision maker is asked to decide the parameter and our analysis shows how good is the value found by the approximation corresponding to this parameter. In addition, we present several observations that may be suitable for boosting up the performance of the proposed approach. Finally, we support our discussion through extensive numerical experiments.

Keywords. Generalized fractional programming, user interaction, approximation approach, error analysis, performance improvement

1 Introduction

Decision makers in different application areas demand effective and easy-to-use solution methods for their optimization problems. In many circumstances, they are ready to welcome a new approach even though this approach may yield a value *close-enough*

[*]Research supported by Hong Kong RGC Earmarked Grants CUHK4233/01E and CUHK4174/03E.

to the actual (or theoretical) optimal value. One way of satisfying this demand is incorporating approximation approaches into well-known solution methods. To our advantage, the precision of an approximation approach depends on predefined parameters. Therefore through user interaction, the tradeoff between the quality of the results and the simplification of the problem can be controlled.

In this paper, we focus on solving a generalized fractional programming (GFP) problem by an approximation approach. GFP problems arise in many real-life applications. One of the earliest example dates back to the time of von Neumann when he used GFP in modeling an expanding economy [von Neumann, J., 1945]. Another important application is within the field of multi-objective programming. In this case, the main goal is to minimize the maximum deviation of the fractional objectives, such as input-output ratios, from their target values [Gugat, M, 1996b, Kornbluth, J.S. and R.E. Steuer, 1981, Jagannathan, R, 1985, Ignizio, J.P., 1976]. Also in numerical analysis, GFP methods are used for solving rational Chebyshev approximations [Barrodale, I., Powell, M.J.D. and F.D.K. Roberts, 1972, Cheney, E.W. and H.L. Loeb, 1961, Gugat, M, 1997, 1996a]. In order to study the theoretical properties of GFP problems, researchers made use of the tools from duality theory, nonlinear programming and numerical analysis. Naturally, the study of the theoretical properties led to the development of various solution methods [Crouzèix, J.P., Ferland, J.A. and S. Schaible, 1985, Crouzèix, J.P., Ferland, J.A. and S. Schaible, 1986, Crouzèix, J.P. and J.A. Ferland, 1991, Barros, A.I., Frenk, J.B.G., Schaible, S. and S. Zhang, 1996b,a, Barros, A, 1998, Gugat, M, 1996b, 1998, Roubi, A, 2000]. The majority of these methods are the variations of the approach suggested in the pioneering work of Crouzèix *et. al.* [1985]. In the literature, these methods are also known as Dinkelbach-type methods [Dinkelbach, W., 1967], and although not immediately clear, it can be shown that these methods are actually cutting plane methods [Barros, A.I. and J.B.G. Frenk, 1995]. Dinkelbach-type methods are based on the idea of solving a sequence of auxiliary problems so that the solutions of the auxiliary problems converge to the solution of the GFP problem. These methods are very powerful, however their performances heavily depend on the effective solution of the auxiliary problems. In general the auxiliary problems are nonsmooth and nonlinear. Therefore, they might pose a major difficulty in solving the GFP problems effectively and easily.

The main idea of this paper is replacing the difficult auxiliary problems in Dinkelbach-type methods with relatively simple approximations. Approximations of these sort yield ε-optimal (close-enough) values to the auxiliary problems. Consequently, we will analyze the error committed by replacing the auxiliary problems with their approximated counterparts. Especially for large size problems, the task of speeding up the algorithms plays an important role. Therefore, we will also propose several ap-

proaches that may lead to a performance improvement. During our analysis, we will work with a generic approximation function since the choice of the approximation function does not alter our results. In order to illustrate our idea, we will then give two particular approximation approaches that will be also used in an extensive numerical results section. In terms of our main contribution, we believe that our work provides a useful reference for a decision maker who is willing to solve a GFP problem effectively at the cost of finding inexact but controlled solutions.

2 Generalized Fractional Programming and The Dinkelbach Method

In this section we give an overview on generalized fractional programming and the most common solution approach, namely, the Dinkelbach method. Before proceeding to our subsequent analysis, let us first introduce the generalized fractional programming problem

$$\lambda^* := \inf_{x \in X} \max_{i \in I} \left\{ \frac{f_i(x)}{g_i(x)} \right\} \tag{P}$$

where $X \subseteq \mathbb{R}^n$ is the feasible region, $I := \{1, 2, \cdots, m\}$ a finite index set and for all $i \in I$, $f_i : \mathbb{R}^n \to \mathbb{R}$ and $g_i : \mathbb{R}^n \to \mathbb{R}$ are the problem functions. In addition, if we define the function $p : X \to (-\infty, \infty]$ by

$$p(x) := \max_{i \in I} \left\{ \frac{f_i(x)}{g_i(x)} \right\}, \tag{2.1}$$

then the optimization problem (P) can be written as

$$\lambda^* := \inf_{x \in X} p(x). \tag{2.2}$$

In the sequel we denote the optimal solution of problem (P) by x^*.

Introducing now the functions $g_{\min} : \mathbb{R}^n \to \mathbb{R}$ by

$$g_{\min}(x) := \min_{i \in I} g_i(x), \tag{2.3}$$

it is necessary to impose the following assumption to avoid pathological instances of problem (P).

Assumption 2.1

$$g_{\min}(x) > 0 \text{ for every } x \in X. \tag{2.4}$$

The main idea of the Dinkelbach method is to provide a solution to the difficult optimization problem (P) by solving a sequence of relatively simple parameterized problems. In this overview section, we focus on two main Dinkelbach-type approaches. The first approach is based on solving constrained subproblems, whereas the second approach aims at reformulating the parameterized subproblems as unconstrained nonlinear optimization problems.

We start with the first approach by defining for every $i \in I$, the functions $\phi_i : \mathbb{R}^{n+1} \to \mathbb{R}$ given by

$$\phi_i(\lambda, x) := f_i(x) - \lambda g_i(x) \tag{2.5}$$

and for every $(\lambda, x) \in \mathbb{R}^{n+1}$, the function

$$\phi(\lambda, x) := \max_{i \in I} \ \phi_i(\lambda, x). \tag{2.6}$$

This leads to the associated parametric *constrained* optimization problem

$$\phi(\lambda) := \inf_{x \in X} \ \phi(\lambda, x). \tag{P_λ}$$

After minor modifications of the proofs in [Crouzèix, J.P., Ferland, J.A. and S. Schaible, 1985], one can show the following important results.

Lemma 2.1 *$\phi(\lambda) \geq 0$ if and only if $-\infty < \lambda \leq \lambda^*$.*

Lemma 2.2 *The following conditions are equivalent.*

1. *The optimization problem (P) has an optimal solution.*
2. *The set $\mathcal{V}$ defined by*

$$\mathcal{V} := \{\lambda \in \mathbb{R} : \text{ Problem } (P_\lambda) \text{ has an optimal solution and } \phi(\lambda) = 0\}$$

   is nonempty.

3. *The set $\mathcal{V}$ is a singleton with $\mathcal{V} = \{\lambda^*\}$.*

We now discuss the unconstrained approach. In this case the feasible region X is given by

$$X := \{x \in \mathbb{R}^n : h_j(x) \leq 0, j \in J\}, \tag{2.7}$$

where $J := \{1, 2, \cdots, p\}$ is a finite index set and $h_j : \mathbb{R}^n \to \mathbb{R}$ for all $j \in J$ with $h(x) := (h_1(x), \cdots, h_p(x))^T$. Clearly, this form of the feasible set is a commonly used form for numerical purposes.

In addition to Assumption 2.1, we also assume that the following condition holds.

Assumption 2.2 *The function $x \to h_j(x)$, $j \in J$ are convex and the set X has a nonempty interior,* i.e.*, there exist some $x_0 \in \mathbb{R}^n$ such that*

$$\max_{j \in J} \; h_j(x_0) < 0. \tag{2.8}$$

As before, we start by introducing the parametric function $\psi_i : \mathbb{R}^{n+1} \to \mathbb{R}$ given by

$$\psi_i(\lambda, x) := \begin{cases} f_i(x) - \lambda g_i(x), & \text{for } i \in I \\ h_i(x), & \text{for } i \in J \end{cases} \tag{2.9}$$

and consider for $(\lambda, x) \in \mathbb{R}^{n+1}$, the function

$$\psi(\lambda, x) := \max_{i \in I \cup J} \psi_i(\lambda, x) \tag{2.10}$$

with the parametric *unconstrained* optimization problem

$$\psi(\lambda) := \inf_{x \in \mathbb{R}^n} \psi(\lambda, x). \tag{$\bar{P}_\lambda$}$$

It is shown in [Roubi, A, 2000, Addou, A. and A. Roubi] that the following versions of the previous two lemmas hold.

Lemma 2.3 *$\psi(\lambda) \geq 0$ if and only if $-\infty < \lambda \leq \lambda^*$.*

Lemma 2.4 *The following conditions are equivalent.*

1. *The optimization problem (P) has an optimal solution.*
2. *The set $\mathcal{W}$ defined by*

   $$\mathcal{W} := \{\lambda \in \mathbb{R} : \text{ Problem } (\bar{P}_\lambda) \text{ has an optimal solution and } \psi(\lambda) = 0\}$$

   is nonempty.
3. *The set $\mathcal{W}$ is a singleton with $\mathcal{W} = \{\lambda^*\}$.*

Using the above results, we can discuss the computational procedures for both constrained and unconstrained cases. In order to apply the Dinkelbach-type approach corresponding to the constrained case, we always assume that problem (P) and for any $\lambda \geq \lambda^*$, problems (P_λ) are solvable. The general scheme of the constrained case has now the form given in Algorithm 2.1.

Algorithm 2.1 Constrained Case

1. Select $x_0 \in X$ and set $k := 1$.
2. Set $\lambda_k := p(x_{k-1})$.
3. Determine an element x_k of the set $\{x \in X : \phi(\lambda_k, x) < 0\}$. If such an element does not exist, then return (λ_k, x_{k-1}), else set $k := k + 1$ and return to Step 2.

As in the constrained case, we assume that problem (P) and for any $\lambda \geq \lambda^*$, problems ($\bar{P}_\lambda$) are solvable. Observe that if $\psi(\lambda_k, x_k) < 0$, then it follows automatically that $h_j(x_k) < 0$ for every $j \in J$ and hence, $x_k \in X$. The algorithm for the unconstrained Dinkelbach-type approach has now the form given in Algorithm 2.2.

Algorithm 2.2 Unconstrained Case

1. Select $x_0 \in X$ and set $k := 1$.
2. Set $\lambda_k := p(x_{k-1})$.
3. Determine an element x_k of the set $\{x \in \mathbb{R}^n : \psi(\lambda_k, x) < 0\}$. If such an element does not exist, then return (λ_k, x_{k-1}), else set $k := k + 1$ and return to Step 2.

The geometrical interpretation of Algorithm 2.1 is illustrated in Figure 2. Clearly, for any $x_k \in \{x \in X : \phi(\lambda, x) < 0\}$ we have

$$\phi(\lambda) \leq \phi(\lambda, x_k). \tag{2.11}$$

Hence, at the k^{th} iteration of Algorithm 2.1, the nonconvex function ϕ is approximated from above by the polyhedral convex function $\lambda \to \phi(\lambda, x_k)$. As a direct consequence of Lemma 2.2, our main goal is to compute λ that satisfi es $\phi(\lambda) = 0$. Instead of this ambitious goal, we solve the approximating equation $\phi(\lambda, x_k) = 0$, which leads to

$$\lambda = \max_{i \in I} \left\{ \frac{f_i(x_k)}{g_i(x_k)} \right\}. \tag{2.12}$$

Notice that this solution is indeed the next iteration in Algorithm 2.1 (Step 2). A similar interpretation can be given for Algorithm 2.2, where the function ψ is approximated by the polyhedral convex function $\lambda \to \psi(\lambda, x_k)$.

In Step 3 of both algorithms we suggest to solve membership problems instead of solving the optimization problems (P_λ) and ($\bar{P}_\lambda$). The reason for this suggestion is given by the following observation. Consider the constrained approach and suppose at

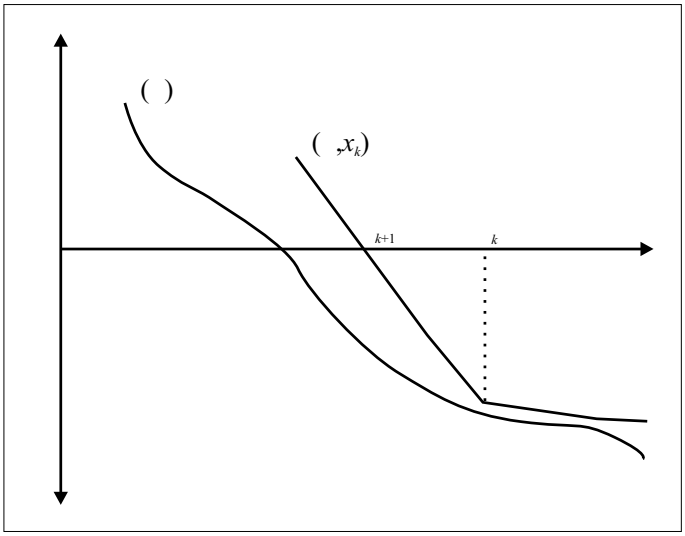

Figure 1: A typical iteration of Algorithm 2.1.

iteration k we compute the optimal solution x_k^* of problem (P_{λ_k}) and at the same time we choose a different vector x_k satisfying $\phi(\lambda_k, x_k) < 0$. It might now happen that the value $\lambda_{k+1} := p(x_k)$ is closer to λ^* than the value λ_{k+1} derived from x_k^*. An instance showing this behavior is given in Example 2.1. This suggests that investing time in solving problem (P_λ) to optimality might not always be the best strategy. On the other hand, to determine whether there exists an element of the set $\{x \in X : \phi(\lambda_k, x) < 0\}$ or decide whether this set is empty, it is most natural to solve problem (P_{λ_k}), and so x_k^* becomes a natural candidate.

Example 2.1 *Suppose we want to solve*

$$\lambda^* = \inf_{x\in[0,10]} \max\{\frac{-11x+1}{2x+2}, \frac{-7x+2}{4x+1}, \frac{3x-2}{16x+3}\}.$$

Let $x_0 = 1$ *then* $\lambda_1 = p(x_0) = 1/19$. *If we solve problem* (P_{λ_1})*, then we find the optimal solution* $x_1^* = 38/89$ *with the objective function value* $\phi(\lambda_1, x_1^*) = -1.21$. *Moreover, at the next iteration, we compute* $\lambda_2^* := p(x_1^*) = -0.068$. *Instead of the optimal solution, if we select* $x_1 = 30/89$ *yielding* $\phi(\lambda_1, x_1) = -0.48$*, then the corresponding value of* λ *at the next iteration becomes* $\lambda_2 := p(x_1) = -0.1178$*, which is less than* λ_2^*.

This observation may also become useful whenever the optimization routine for solving the auxiliary problems requires a considerable effort. In this case, one may stop as soon as the optimization routine finds a feasible, but not necessarily optimal, point x that solves the membership problem. As Lemma 2.1 and Lemma 2.3 show, from a practical point of view, we only need a decreasing sequence of λ_k values.

3 The Approximation Approach

Notice that in both Algorithm 2.1 and Algorithm 2.2, the membership problems at Step 3 can be solved by reformulating the problems as nonlinear optimization problems, *i.e.*, one needs to solve (approximately, see Example 2.1), in the constrained case, the auxiliary optimization problem

$$\min_{x\in X} \max\{f_1(x) - \lambda g_1(x), \cdots, f_m(x) - \lambda g_m(x)\} \tag{3.1}$$

and in the unconstrained case, the problem

$$\min_{x\in \mathbb{R}^n} \max\{f_1(x) - \lambda g_1(x), \cdots, f_m(x) - \lambda g_m(x), h_1(x), \cdots, h_p(x)\}. \tag{3.2}$$

Although the $\max$ function is a convex increasing function, unfortunately it is nonsmooth. Therefore, we still encounter a nonsmooth objective function in the auxiliary optimization problems. One promising solution to this obstacle is to approximate the $\max$ function by a smooth alternative. Consider now a smooth, convex and increasing function $\Pi_s : \mathbb{R}^s \to \mathbb{R}$ that satisfi es

$$\lim_{\varepsilon\downarrow 0} \varepsilon\Pi_s(\frac{y}{\varepsilon}) = \max\{y_1, \cdots, y_s\} \tag{3.3}$$

and

$$0 \leq \varepsilon\Pi_s(\frac{y}{\varepsilon}) - \max\{y_1, \cdots, y_s\} \leq \beta_s(\varepsilon) \tag{3.4}$$

for every $y \in \mathbb{R}^s$. As a direct consequence of the above two relations, the function $y \to \varepsilon\Pi_s(\frac{y}{\varepsilon})$ becomes a good candidate for approximating the nonsmooth objective functions of the auxiliary problems. Let us now give two examples that will also be used in the numerical results section.

Example 3.1 *It is easy to check that the entropy-type function*

$$\Pi_s(y) = \log\left(\sum_{i=1}^{s} e^{y_i}\right)$$

is convex, differentiable and increasing [Fang, S.-C., Rajasekera, J. R. and H.-S. J. Tsao, 197]. Moreover, it satisfies relation (3.3) and relation (3.4) with $\beta_s(\varepsilon) = \varepsilon \log s$ *[Ben-Tal, A. and M. Teboulle, 1989].*

Example 3.2 *In some problems, entropy-type function in Example 3.1 may create overflow problems because of the exponential function. Recently Birbil* et. al. *proposed a more stable approximation method that aims at extending two dimensional approximation functions to higher dimensions by recursion [Birbil, Ş.İ., Fang, S.-C., Frenk, J.B.G. and S. Zhang, 2002]. The main idea of the method is based on*