American Journal of Operations Research, 2018, 8, 266-280
http://www.scirp.org/journal/ajor
ISSN Online: 2160-8849
ISSN Print: 2160-8830

A Class of Continuous Separable Nonlinear Multidimensional Knapsack Problems

Bin Zhang[1], Zhe Lin[1], Yu Wang[2]

[1]Lingnan College, Sun Yat-sen University, Guangzhou, China
[2]School of Economic and Business Administration, Chongqing University, Chongqing, China
Email: zhangb38@mail.sysu.edu.cn

How to cite this paper: Zhang, B., Lin, Z. and Wang, Y. (2018) A Class of Continuous Separable Nonlinear Multidimensional Knapsack Problems. *American Journal of Operations Research*, **8**, 266-280.
https://doi.org/10.4236/ajor.2018.84015

Received: May 11, 2018
Accepted: June 29, 2018
Published: July 2, 2018

Abstract

The nonlinear multidimensional knapsack problem is defined as the minimization of a convex function with multiple linear constraints. The methods developed for nonlinear multidimensional programming problems are often applied to solve the nonlinear multidimensional knapsack problems, but they are inefficient or limited since most of them do not exploit the characteristics of the knapsack problems. In this paper, by establishing structural properties of the continuous separable nonlinear multidimensional knapsack problem, we develop a multi-tier binary solution method for solving the continuous nonlinear multidimensional knapsack problems with general structure. The computational complexity is polynomial in the number of variables. We presented two examples to illustrate the general application of our method and we used statistical results to show the effectiveness of our method.

Keywords

Nonlinear Programming, Convex Programming, Multidimensional Knapsack, Separable Knapsack, Lagrangian Relaxation

1. Introduction

The nonlinear multidimensional knapsack problem is defined as minimizing a convex function with multiple linear constraints. The nonlinear knapsack problem is a class of nonlinear programming, and some methods designed for nonlinear programming can be applied for solving the nonlinear multidimensional knapsack problems. The general nonlinear programming problems have been intensively studied in the last decades, and some different methods have been developed, such as Newton method [1] [2] [3], branch and bound method [4] [5], interior point method [6] [7], sequential quadratic programming method [8]

[9] and the filter method [10] [11]. These methods are designed for nonlinear programming problems, and some of them are inefficient or limited for solving the nonlinear knapsack problems since they do not consider the characteristics of the knapsack problems.

Generally, it is much faster and more reliable to solve knapsack problems with specialized methods than with standard methods [12]. Many researchers studied the solution methods for the nonlinear knapsack problems based on the specialized knapsack structures. Most of the research studied the problems with single constraint. Two basic specialized methods are mainly applied for solving the single-constraint nonlinear knapsack problem. One is the multiplier search method [13], and another is the pegging method [14] [15]. Recently, some new methods are proposed for efficiently solving the single-constraint nonlinear knapsack problem. Zhang and Hua developed a united method for solving a class of continuous separable nonlinear knapsack problems [16]. Kiwiel developed the breakpoint searching method for the continuous quadratic knapsack problem [17]. Sharkey *et al.* studied a general class of nonlinear non-separable continuous knapsack problem [18].

Most research of nonlinear knapsack problems studied the one-dimensional problems with continuous or integer variables, and the proposed methods cannot be directly extended for solving multi-dimensional problems. Some researchers attempted to solve multi-dimensional problems with integer-valued variables. Morin and Marsten firstly studied the nonlinear multidimensional knapsack problems and developed the imbedded state space approach [19]. Some researchers investigate the efficiency of other methods, such as smart greedy method [20], cut method [21] [22], branch and bound method [23] and branch and cut method [24]. Other research studied different applications of multidimensional knapsack, e.g., multi-product newsvendor problems with multiple constraints [10] [25] [26] [27]. The continuous separable nonlinear multidimensional knapsack problems with general structure have not been well studied due to its complexity, and the specialized methods are very limited.

This paper establishes some structural properties of the continuous separable nonlinear multidimensional knapsack problem, and develops a multi-tier binary solution method for solving a class of continuous nonlinear multidimensional knapsack problems with general structure. The computational complexity is polynomial in the number of variables. We presented two examples to illustrate the application of our method, and the statistical study with the randomly generated instances for different problem sizes are reported to show the effectiveness of our method.

The paper is organized as follows. In Section 2, the nonlinear multidimensional knapsack problem is described. Section 3 studies the structural properties of the problem, and develops the algorithm. Section 4 presents the illustrative examples and the statistical results. Finally, the concluding remarks are given in Section 5. All proofs are listed in **Appendix**.

2. Problem Formulation

The continuous separable nonlinear multidimensional knapsack problem studied in this paper is as follows (denoted as problem P):

$$\text{Min } \boldsymbol{f}(\boldsymbol{x}) = \sum_{i=1}^{N} f_i(x_i), \tag{1}$$

Subject to

$$\sum_{i=1}^{N} c_{i,j} x_i \le C_j, j = 1, \cdots, M, \tag{2}$$

$$l_i \le x_i \le u_i, i = 1, \cdots, N. \tag{3}$$

The notation used in this paper is listed in Table 1.

In problem P, all objective functions $f_i(x_i), i = 1, \cdots, n$ are convex and differentiable, the unit resource coefficient $c_{i,j} > 0$ for all $i = 1, \cdots, N, j = 1, \cdots, M$, the resource constraints $C_j > 0$ for all $j = 1, \cdots, M$, and the lower and upper bounds satisfy $0 \le l_i < u_i$ for all $i = 1, \cdots, N$.

Since the objective functions and the feasible domain in problem P are all convex, the optimality condition for problem P can be characterized using KKT conditions. Let $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_M)$, $\lambda_j \ge 0, j = 1, \cdots, M$, be the Lagrange multiplier vector for the constraints given in Equation (2), and $\boldsymbol{w} = (w_1, \cdots, w_N)$, $w_i \ge 0, i = 1, \cdots, N$, $\boldsymbol{v} = (v_1, \cdots, v_N), v_i \ge 0, i = 1, \cdots, N$ be the Lagrange multiplier vectors for the constraints in Equation (3). Thus, the Lagrange function for problem P can be written as:

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{v}) = \sum_{i=1}^{N} f_i(x_i) - \sum_{j=1}^{M} \lambda_j \left(C_j - \sum_{i=1}^{N} c_{i,j} x_i \right) - \sum_{i=1}^{N} w_i (x_i - l_i) + \sum_{i=1}^{N} v_i (x_i - u_i). \tag{4}$$

Table 1. Notation.

Notations	Definitions
N	total number of variables
M	total amount of resource
i	variable index
j	resource index
X	decision variable vector $\boldsymbol{x} = (x_1, \cdots, x_N)$
$f_i(x_i)$	the objective function related to variable x_i
$g_i(x_i)$	the derivative function of $f_i(x_i)$, $g_i(x_i) = \mathrm{d}f_i(x_i)/\mathrm{d}x_i$
$k_i(x_i)$	the derivative function of $g_i(x_i)$, $k_i(x_i) = \mathrm{d}g_i(x_i)/\mathrm{d}x_i$
$h_i(\cdot)$	the inverse function of $g_i(x_i)$, $h_i(\cdot) = g_i^{-1}(\cdot)$
$c_{i,j}$	coefficient of variable *i* of resource *j*
C_j	available amount of resource *j*
$\boldsymbol{\lambda}$	the Lagrange multiplier vector for the resource constraints
$\boldsymbol{w}$	the Lagrange multiplier vector for the variable constraints
$\boldsymbol{v}$	the Lagrange multiplier vector for the variable constraints
$\boldsymbol{f}(\cdot)$	The objective function vector

Let $g_i(x_i) = \mathrm{d}f_i(x_i)/\mathrm{d}x_i$, $i = 1,\cdots,N$. The KKT conditions for problem P can be summarized as the following proposition.

Proposition 1: The KKT conditions for problem P are:

$$g_i(x_i) + \sum_{j=1}^{M} \lambda_j c_{i,j} - w_i + v_i = 0, i = 1,\cdots,N, \tag{5}$$

$$\sum_{i=1}^{N} w_i(x_i - l_i) + \sum_{i=1}^{N} v_i(x_i - u_i) = 0, \tag{6}$$

$$\lambda_j\left(\sum_{i=1}^{N} c_{i,j}x_i - C_j\right) = 0, j = 1,\cdots,M. \tag{7}$$

Since $f_i(x_i)$ is convex in x_i, $g_i(x_i)$ is an increasing function of x_i. Let $\bar{x}_i$ be the point that satisfies $g_i(x_i) = 0$ if $g_i(0) \le 0$ and $\lim_{x_i\to+\infty} g_i(x_i) \ge 0$. If $g_i(0) > 0$, we let $\bar{x}_i = 0$. If $\lim_{x_i\to+\infty} g_i(x_i) < 0$, we set $\bar{x}_i = +\infty$. Then $\bar{x}_i$ is the optimal solution to the objective function in Equation (1) without any constraint. We summarize it as

$$\begin{aligned}\bar{x}_i &= \arg\min\{f_i(x_i), 0 \le x_i \le +\infty\} \\ &= \begin{cases} 0, & \text{if } g_i(0) > 0, \\ \arg\{x_i \mid g_i(x_i) = 0\}, & \text{if } g_i(0) \le 0 \text{ and } \lim_{x_i\to+\infty} g_i(x_i) \ge 0, \\ +\infty, & \text{if } \lim_{x_i\to+\infty} g_i(x_i) < 0. \end{cases}\end{aligned} \tag{8}$$

3. Structural Properties and Solution Method

In this section, we first investigate the structural properties of the optimal solution to problem P. Then we develop a solution method based on the structural properties for solving problem P.

3.1. Structural Properties

We denote by problem PR the knapsack relaxation problem from problem P, in which the constraints in Equation (2) are relaxed. This implies that we do not consider Equation (2) in problem PR. By analyzing the solution to problem PR, we can find the way to construct the solution to problem P. We let $\hat{x}_i$ ($i = 1,\cdots,N$) be the optimal solution to problem PR, then $\hat{x}_i$ ($i = 1,\cdots,N$) has the following property.

Proposition 2: The optimal solution to problem PR is $\hat{x}_i = \min\{\max\{\bar{x}_i, l_i\}, u_i\}$.

If $\sum_{i=1}^{N} c_{i,j}\hat{x}_i \le C_j$ holds for some $j = 1,\cdots,M$, then the corresponding constraints in problem P are inactive, which can be removed from problem P. In the following, without loss of generality, we assume that $\sum_{i=1}^{N} c_{i,j}\hat{x}_i > C_j$ for all $j = 1,\cdots,M$. The KKT conditions in Equation (7) are met at either $\lambda_j = 0$, or $\sum_{i=1}^{N} c_{i,j}x_i = C_j$. The condition $\lambda_j = 0$ implies that there is enough resource *j* at the optimal solution, and hence the *j*-th constraint is inactive. $\sum_{i=1}^{N} c_{i,j}x_i = C_j$ means that the *j*-th constraint is active, and knapsack space of the *j*-th constraint must be fully utilized at the optimal solution.

We denote by $\boldsymbol{x}^*$ the optimal solution to problem P and $\boldsymbol{\lambda}^*$ the corresponding Lagrange multiplier vector. Let $x_i(\boldsymbol{\lambda})$ be a solution of the KKT conditions in Equation (5) and Equation (6). We denote by $h_i(\cdot) = g_i^{-1}(\cdot)$, then we have the following proposition.

Proposition 3. (a) $x_i(\boldsymbol{\lambda}) = \min\left\{\max\left\{h_i\left(-\sum_{j=1}^{M}\lambda_j c_{i,j}\right), l_i\right\}, u_i\right\}$, $i = 1,\cdots,N$.

(b) If $(x(\boldsymbol{\lambda}),\boldsymbol{\lambda})$ satisfies $\lambda_j = 0$ or $\sum_{i=1}^{N} c_{i,j}x_i = C_j$, $j = 1,\cdots,M$, then we have $\boldsymbol{x}^* = \boldsymbol{x}(\boldsymbol{\lambda})$.

For any given $\lambda_M \ge 0$, we let $\boldsymbol{x}(\lambda_M)$ and $\lambda_1,\cdots,\lambda_{M-1}$ be the optimal solution of Equations (5) and (6) and $\lambda_j\left(\sum_{i=1}^{N} c_{i,j}x_i - C_j\right) = 0$, $j = 1,\cdots,M-1$. For ease of exposition, we denote problem P as $P(\boldsymbol{f}, M)$, where $\boldsymbol{f} = (f_i,\cdots,f_N)$ is the objective function vector. Problem $P(\hat{\boldsymbol{f}}(\lambda_M), M-1)$ with $\hat{f}_i(\lambda_M) = f_i + \lambda_M c_{i,M}x_i$, $i = 1,\cdots,N$, is an $M-1$ constraint problem with the objective function $\hat{f}_i(\lambda_M)$ and the first $M-1$ knapsack constraints of problem P.

By analyzing the structural properties of $\boldsymbol{x}(\lambda_M)$ and $P(\hat{\boldsymbol{f}}(\lambda_M), M-1)$, we can prove the following proposition.

Proposition 4. (a) If $(\boldsymbol{x}(\lambda_M), \lambda_M)$ satisfies $\lambda_M = 0$ or $\sum_{i=1}^{N} c_{i,M}x_i(\lambda_M) = C_M$, then we have $\boldsymbol{x}^* = \boldsymbol{x}(\lambda_M)$.

(b) $\boldsymbol{x}(\lambda_M)$ is the optimal solution to problem $P(\hat{\boldsymbol{f}}(\lambda_M), M-1)$ with $\hat{f}_i(\lambda_M) = f_i + \lambda_M c_{i,M}x_i$, $i = 1,\cdots,N$.

From Proposition 4(a), we know that the optimal solution to problem $P(\hat{\boldsymbol{f}}(\lambda_M), M-1)$ is obtained in two possible cases: 1) $\lambda_M = 0$, which means that the constraint $\sum_{i=1}^{N} c_{i,M}x_i(\lambda_M) \le C_M$ is not binding and it can be removed from problem $P(\boldsymbol{f}, M)$. Therefore, $\boldsymbol{x}^*$ can be obtained by solving problem $P(\boldsymbol{f}, M-1)$, which has the same structure as problem $P(\boldsymbol{f}, M)$; 2) $\sum_{i=1}^{N} c_{i,M}x_i(\lambda_M) = C_M$, which implies that $\sum_{i=1}^{N} c_{i,M}x_i(\lambda_M) \le C_M$ is an active constraint, and the optimal solution must be obtained at $\sum_{i=1}^{N} c_{i,M}x_i(\lambda_M) = C_M$ with $\lambda_M > 0$.

Since problem $P(\boldsymbol{f}, M)$ can be solved by solving problem $P(\boldsymbol{f}, M-1)$ in the case of $\lambda_M = 0$. In the following, we study the case of $\lambda_M > 0$. Proposition 4(b) indicates that problem $P(\hat{\boldsymbol{f}}(\lambda_M), M-1)$ determines the optimal values of $\boldsymbol{x}(\lambda_M)$ and λ_j, $j = 1,\cdots,M-1$. For any $\lambda_M > 0$, the $M-1$ resource constraints could be active or inactive, and the N decision variables could take bound values or non-bound values.

If $\lambda_j > 0$, $j = 1,\cdots,M-1$, constraint j will be active, thus we denote by $J(\lambda_M) = \{ j \mid \lambda_j > 0, j = 1,\cdots,M \}$ the active constraint set for the given λ_M. Note that $J(\lambda_M)$ includes at least one active constraint for the case of $\lambda_M > 0$.

From Equation (5), we know $x_i(\lambda_M) > l_i$ if $-g_i(l_i) - \sum_{j=1}^{M} \lambda_j c_{i,j} > 0$, $i = 1,\cdots,N$, and $x_i(\lambda_M) < u_i$ if $-g_i(u_i) - \sum_{j=1}^{M} \lambda_j c_{i,j} > 0$, $i = 1,\cdots,N$. For the given λ_M, we define the non-bound variable set $I(\lambda_M)$, and lower and upper bound variable sets $I_L(\lambda_M)$ and $I_U(\lambda_M)$ as

$$I(\lambda_M) = \left\{ i \mid g_i(l_i) < -\sum_{j=1}^{M} \lambda_j c_{i,j} < g_i(u_i), i = 1,\cdots,N \right\}, \tag{9}$$

$$I_L(\lambda_M) = \left\{ i \mid -\sum_{j=1}^{M} \lambda_j c_{i,j} \le g_i(l_i), i = 1,\cdots,N \right\}, \tag{10}$$

$$I_U(\lambda_M) = \left\{ i \mid -\sum_{j=1}^{M} \lambda_j c_{i,j} \ge g_i(u_i), i = 1,\cdots,N \right\}. \tag{11}$$

Let $m = |J(\lambda_M)|$, $n = |I(\lambda_M)|$, $n_L = |I_L(\lambda_M)|$, and $n_U = |I_U(\lambda_M)|$. For the given $\lambda_M > 0$, without changing the orders of indices *j* and *i*, we re-index the constraints in the active constraint set $J(\lambda_M)$ as $j = 1,\cdots,m$, and we re-index the variables in the non-bound variable set $I(\lambda_M)$ as $i = 1,\cdots,n$, and re-index the variables in $I_L(\lambda_M)$ and $I_U(\lambda_M)$ as $i = 1,\cdots,n_L$, and $i = 1,\cdots,n_U$, respectively. As a result, constraint *M* in the original problem is re-indexed as constraint *m*, and λ_M is also restated as λ_m.

We define $G_j(\lambda_1,\cdots,\lambda_m) \equiv \sum_{i=1}^{N} c_{i,j} x_i(\boldsymbol{\lambda}) - C_j = 0$, $j = 1,\cdots,m-1$, and substitute

$$x_i(\boldsymbol{\lambda}) = \min\left\{ \max\left\{ h_i\left(-\sum_{j=1}^{M} \lambda_j c_{i,j} \right), l_i \right\}, u_i \right\}$$

into $G_j(\lambda_1,\cdots,\lambda_m)$, then we have

$$G_j(\lambda_1,\cdots,\lambda_m) \equiv \sum_{i=1}^{n} c_{i,j} h_i\left(-\sum_{s=1}^{m} \lambda_s c_{i,s} \right) - \left(C_j - \sum_{i=1}^{n_L} c_{i,j} l_i - \sum_{i=1}^{n_U} c_{i,j} u_i \right) = 0, \tag{12}$$

Taking the derivative of Equation (12), we get

$$\begin{aligned} \frac{\mathrm{d}G_j(\lambda_1,\cdots,\lambda_m)}{\mathrm{d}\lambda_m} &= -\left[\sum_{i=1}^{n} \frac{c_{i,j}}{k_i(x_i(\lambda_1,\cdots,\lambda_m))} \sum_{s=1}^{m} \frac{\mathrm{d}\lambda_s}{\mathrm{d}\lambda_m} c_{i,s} \right] \\ &= -\sum_{s=1}^{m} \sum_{i=1}^{n} \frac{c_{i,j} c_{i,s}}{k_i(x_i(\lambda_1,\cdots,\lambda_m))} \frac{\mathrm{d}\lambda_s}{\mathrm{d}\lambda_m} \\ &= 0, \ j = 1,\cdots,m-1 \end{aligned}, \tag{13}$$

where $k_i(x_i) = \mathrm{d}g_i(x_i)/\mathrm{d}x_i$.

Since $f_i(x_i)$, $i = 1,\cdots,n$ are differentiable convex, we know $g_i(x_i)$ is increasing and $k_i(x_i(\lambda_1,\cdots,\lambda_m)) > 0$. Note that $\hat{f}_i(\lambda_M) = f_i + \lambda_M c_{i,M} x_i$ has the same structure as $f_i(x_i)$. So we define

$$\rho_i = \frac{1}{k_i\left(x_i\left(\lambda_1,\cdots,\lambda_m\right)\right)} > 0,$$

$i = 1,\cdots,n$, and $a_{js} = \sum_{i=1}^{n} \rho_i c_{i,j} c_{i,s}$, $j,s = 1,\cdots,m$, then Equation (13) can be rewritten in matrix form:

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{(m-1)1} & a_{(m-1)2} & \cdots & a_{(m-1)m} \end{pmatrix} \begin{pmatrix} \mathrm{d}\lambda_1/\mathrm{d}\lambda_m \\ \vdots \\ \mathrm{d}\lambda_{m-1}/\mathrm{d}\lambda_m \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{14}$$

In order to solve $\frac{\mathrm{d}\lambda_j}{\mathrm{d}\lambda_m}$, $j = 1,\cdots,m-1$, from Equation (14), we further define

$$H_m = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} \end{vmatrix}, \tag{15}$$

and denote by $H_{j(m-1)}$, $j = 1,\cdots,m-1$ the m-1 dimensional determinant in which the j column of H_{m-1} is replaced by $\left(a_{1m}, a_{2m}, \cdots, a_{(m-1)m}\right)^{\mathrm{T}}$. We have the following formula from Equation (14) and Equation (15):

$$\frac{\mathrm{d}\lambda_j}{\mathrm{d}\lambda_m} = -\frac{H_{j(m-1)}}{H_{m-1}},\ j = 1,\cdots,m-1,\ m > 1. \tag{16}$$

Notice that the above results have similar structures as the results in Zhang [27]. Using the similar way, we can prove that

$$\begin{aligned} \frac{\mathrm{d}\sum_{i=1}^{n} c_{i,m} x_i\left(\lambda_m\right)}{\mathrm{d}\lambda_m} &= -\sum_{i=1}^{n} \rho_i c_{i,m}\left(\sum_{j=1}^{m-1} c_{i,j}\frac{\mathrm{d}\lambda_j}{\mathrm{d}\lambda_m} + c_{i,m}\right) \\ = -\sum_{i=1}^{n} \rho_i c_{i,m}\left(-\sum_{j=1}^{m-1} c_{i,j}\frac{H_{j(m-1)}}{H_{m-1}} + c_{i,m}\right) &= -\frac{H_m}{H_{m-1}} < 0 \end{aligned}. \tag{17}$$

Since constraint M in the original problem is re-indexed as constraint m, and λ_M is also restated as λ_m, then $\sum_{i=1}^{n} c_{i,m} x_i\left(\lambda_m\right)$ is equivalent to $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right)$ in problem P with the original index, thus we know that $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right)$ is a decreasing in λ_M.

Therefore, there are three possible cases: 1) When $\lambda_M = 0$, we get the optimal solution to problem $P\left(\boldsymbol{f}, M\right)$ by solving problem $P\left(\boldsymbol{f}, M-1\right)$; 2) If $\lambda_M > 0$ and $m = 1$, we obtain the optimal solution to problem $P\left(\boldsymbol{f}, M\right)$ by setting $x_i\left(\lambda_M\right) = \min\left\{\max\left\{h_i\left(-\lambda_M c_{i,M}\right), l_i\right\}, u_i\right\}$; 3) When $\lambda_M > 0$ and $m > 1$, we can solve problem $P\left(\boldsymbol{f}, M\right)$ by studying problem $P\left(\hat{\boldsymbol{f}}\left(\lambda_M\right), M-1\right)$, with $\hat{f}_i\left(\lambda_M\right) = f_i + \lambda_M c_{i,M} x_i$.

3.2. Solution Method

According to Proposition 2, we can solve $\boldsymbol{x}^*$ by searching the optimal value of $\boldsymbol{\lambda}$. Before presenting the solution method, we first study the bounds for $\boldsymbol{\lambda}$. The lower bound for λ is 0, and the upper bound for λ is given in the following proposition.

Proposition 5. *The upper bound of* λ_i is $\max\left(0, \max_{i=1,\cdots,N}\left\{-g_i\left(l_i\right)/c_{i,M}\right\}\right)$.

From Proposition 4, we get the optimal value of $\boldsymbol{x}^*$ if the optimal solution $\boldsymbol{x}\left(\lambda_M\right)$ to problem $P\left(\hat{\boldsymbol{f}}\left(\lambda_M\right), M-1\right)$ satisfies

$$\lambda_M\left(\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right) - C_M\right) = 0 .$$

Since $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right)$ is decreasing in λ_M, the optimal solution can be found by applying the binary search over $\left[0, \max\left(0, \max_{i=1,\cdots,N}\left\{-g_i\left(l_i\right)/c_{i,M}\right\}\right)\right]$. Since Problem $P\left(\hat{\boldsymbol{f}}\left(\lambda_M\right), M-1\right)$ has the same structure as problem $P\left(\boldsymbol{f}, M\right)$, we can use a multi-tier binary search method to solve problem P. Main steps of the multi-tier binary search method are given in Algorithm 1.

```
Algorithm 1: SloveP(f, M)
Step 1: If $M = 0$, then let $x_i^* = \min\left\{\max\left\{\arg\left\{g_i\left(x_i\right)=0\right\}, l_i\right\}, u_i\right\}$, stop;
Step 2: Let $\lambda_M^L = 0$, $\lambda_M^U = \max\left(0, \max_{i=1,\cdots,N}\left\{-g_i\left(l_i\right)/c_{i,M}\right\}\right)$;
Step 3: Let $\lambda_M = \left(\lambda_M^L + \lambda_M^U\right)/2$;
Step 4: If $\lambda_M = 0$, then let $x_i^* = \mathrm{SolveP}\left(\boldsymbol{f}, M-1\right)$ and $\lambda_M^* = 0$, stop;
Step 5: If $M = 1$, then let $x_i\left(\lambda_M\right) = \min\left\{\max\left\{h_i\left(-\lambda_M c_{i,M}\right), l_i\right\}, u_i\right\}$;
        If $M > 1$, then let $x_i\left(\lambda_M\right) = \mathrm{SolveP}\left(\hat{\boldsymbol{f}}\left(\lambda_M\right), M-1\right)$
Step 6: If $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right) > C_M$, then let $\lambda_M^L = \lambda_M$, go to Step 3;
        If $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right) < C_M$, then let $\lambda_M^U = \lambda_M$, go to Step 3;
Step 7: Let $x_i^* = x_i\left(\lambda_M\right)$ and $\lambda_M^* = \lambda_M$, stop.
```

In the algorithm, we first solve the unconstrained problem with bounded variables (Step 1) to obtain $\boldsymbol{x}^*$. If the constraints are active, we apply the binary search procedure (Step 2 - 7) over interval $\left[\lambda_M^L, \lambda_M^U\right]$ to determine λ_M^*. If either $\lambda_M = 0$ or $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right) = C_M$, the binary search procedure terminates. If $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right) \le C_M$ is not binding, then the iterating process will end in Step 4 with $\lambda_M = 0$. Therefore, we can get the optimal solution x_i^* by solving problem $P\left(\boldsymbol{f}, M-1\right)$. If the constraint $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right) \le C_M$ is active, the solution procedure will stop at Step 7 with $\sum_{i=1}^{N} c_{i,M} x_i\left(\lambda_M\right) = C_M$. Step 5 derives $x_i\left(\lambda_M\right)$ by solving problem $P\left(\hat{\boldsymbol{f}}\left(\lambda_M\right), M-1\right)$ with $\hat{f}_i\left(\lambda_M\right) = f_i + \lambda_M c_{i,M} x_i$ for the given $\lambda_M > 0$. If $M = 1$, problem $P\left(\hat{\boldsymbol{f}}\left(\lambda_M\right), M-1\right)$ has no knapsack constraint, and hence we have $x_i\left(\lambda_M\right) = \min\left\{\max\left\{h_i\left(-\lambda_M c_{i,M}\right), l_i\right\}, u_i\right\}$. If $M > 1$, we can solve the problem recursively. Problem $P\left(\hat{\boldsymbol{f}}\left(\lambda_M\right), M-1\right)$ has the same structure as problem $P\left(\boldsymbol{f}, M\right)$, and hence the algorithm can call itself recursively to solve the problem $P\left(\hat{\boldsymbol{f}}\left(\lambda_M\right), M-1\right)$.

The algorithm is a recursive algorithm with *M* tiers of binary search loop. The computational complexity of *M*-tier binary search procedure is $O\left(\left(\log_2\left(1/\varepsilon\right)\right)^M\right)$, where ε is the error target for the binary search. The computational complexity of the last recursive step is $O\left(N\right)$. Therefore, the proposed algorithm has the computational complexity $O\left(\left(\log_2\left(1/\varepsilon\right)\right)^M N\right)$, which is polynomial in the number of decision variables *N*.

4. Numerical Study

The solution method developed in this paper can be used for solving the continuous nonlinear multidimensional knapsack problems with general structure, so many application problems with different objective functions summarized in Zhang and Hua with multiple constraints can be used to show the application of our method [16].

In our numerical study, we first show the application of our method using two examples: quadratic multidimensional knapsack problem (QMK) and the production planning problem presented in Bretthauer and Shetty [12]. Then we use the statistical study to show the efficiency of our method. All computational experiments are conducted on a laptop (dual processor 2.00 GHz, memory 2.96G) with Matlab R2011a.

4.1. The Illustrative Examples

The first illustrative example is a separable quadratic knapsack problem. We set the objective function as $f_i(x_i) = a_i(x_i - b_i)^2$, $a_i > 0, i = 1, \cdots, N$. It has two resource constraints: C_1 = 12,000 and C_2 = 10,000. **Table 2** gives the relevant information for this example. x_i^* is the optimal solution obtained by applying our algorithm. To show the efficiency of our method, we plot the values of λ_M^L, λ_M^U and λ_M in the iteration process for solving the example in **Figure 1**. **Figure 1** shows our algorithm can solve the problem within very limited iterations.

In the second example, we solve the production planning problem in Bretthauer and Shetty [12]. The objective function was set as

$$f_i(x_i) = \min \sum_{i=1}^{n} \left(h_i + d_i x_i + \frac{e_i}{x_i} \right),$$

$i = 1, \cdots, N$. There are three resource constraints: C_1 = 200, C_2 = 300, and C_3 = 500. We use the same parameters used in Bretthauer and Shetty [12]. The relevant

Table 2. Parameters and solution for the first example.

i	a_i	b_i	$c_{i,1}$	$c_{i,2}$	l_i	u_i	x_i^*
1	12	20	50	100	6.7	10	10.0000
2	15	18	50	80	1	20	13.4020
3	20	8	50	100	2	30	3.6894
4	10	28	150	100	2.5	40	19.3787
5	10	10	100	80	5	5.6	5.0000
6	20	30	100	80	3	20	20.0000
7	18	25	100	100	8	25	20.2104
8	15	30	100	88	3	20	20.0000
λ^*			0.0092	1.7243			
f^*							6795

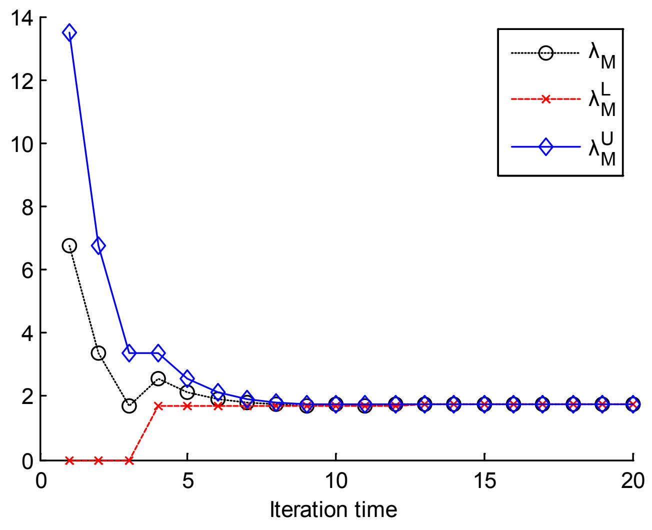

Figure 1. $\lambda_M^L, \lambda_M^U, \lambda_M$ in the iteration process for solving the first example.

information for this example is listed in **Table 3**. x_i^* is the optimal solution obtained by applying our algorithm.

4.2. The Statistical Results

In this subsection, we present two numerical experiments to show the effectiveness of our method for solving problems with different scale and objective functions. In the first experiment, parameters of the QMK problems are all randomly generated. We use the notation $z \sim U(\alpha, \beta)$ to denote that z is uniformly generated over $[\alpha, \beta]$. The parameters of QMK instances are generated as follows: $a_i \sim U(1,2)$, $b_i \sim U(5,10)$, $c_{i,j} \sim U(1,10)$, $l_i \sim U(5,15)$, $u_i \sim U(20,30)$ and $C_j \sim N \times U(100000, 200000)$, for $i = 1, \cdots, N; j = 1, \cdots, M$.

In this experiment, we set problems with different sizes, respectively with M = 4 and N = 10, M = 2 and N = 100, M = 3 and N = 100, M = 2 and N = 1000. For each problem size, 50 test instances are randomly generated. The statistical results on number of iterations and computation time (in seconds) are reported in **Table 4**.

In the Second experiment, we solve the production planning problem with randomly generated parameters. The parameters of the instances are generated as follows: $d_i \sim U(30,50)$, $e_i \sim U(100,200)$, $c_{i,j} \sim U(10,50)$, $l_i \sim U(1,5)$, $u_i \sim U(20,30)$ and $C_j \sim N \times U(100000, 200000)$, for $i = 1, \cdots, N; j = 1, \cdots, M$.

In this experiment, we set problems with different sizes, respectively with M = 4 and N = 10, M = 2 and N = 100, M = 3 and N = 100, M = 2 and N = 1000. For each problem size, we randomly generated 50 test instances. The statistical results on number of iterations and computation time (in seconds) are presented in **Table 5**.

From **Table 4** and **Table 5**, we observe that the standard deviations of number of iterations and computation times are quite low. It implies that our method is quite effective with different objective functions. We also observe that the

Table 3. Parameters and solutions for the second example.

i	h_i	d_i	e_i	$c_{i,1}$	$c_{i,2}$	$c_{i,3}$	l_i	u_i	x_i^*
1	10	30.2	83	10	1	11	1	20	1.6578
2	20	5	15	12	2	2	5	20	5.0000
3	14	42.5	63	1	2	4	2	25	2.0000
4	13	48	81	5	5	5	4.4	22	4.4000
5	4	42	65	3	1	6	2.3	25	2.3000
6	5	36	75	8	2	3	2.2	24	2.2000
7	13	41.4	94	5	2	2	1	24	1.5068
8	27	22.5	20	1	3	3	3.5	22	3.5000
9	40	31.6	12	2	5	5	1.6	30	1.6000
10	23	44	55.5	3	8	8	1.9	32	1.9000
λ^*				0.0051	0.0064	0.0064			
f^*									1261.5

Table 4. Statistical results for randomly generated QMK problems.

		# of iterations				Computation time			
N, M		10, 4	100, 2	100, 3	1000, 2	10, 4	100, 2	100, 3	1000, 2
Mean		15.30	16.82	16.92	17.86	1.1713	0.0073	0.1044	0.0225
Std.dev.		1.0926	0.7475	0.8041	0.3505	0.2811	0.0026	0.0365	0.0083
95% C.I.	Lower	13	15	15	17	0.6012	0.0028	0.0405	0.0082
	Upper	17	18	18	18	1.7729	0.0118	0.1575	0.0381

Table 5. Statistical results for randomly generated production planning problems.

		# of iterations				Computation time			
N, M		10, 4	100, 2	100, 3	1000, 2	10, 4	100, 2	100, 3	1000, 2
Mean		22.46	23.82	23.92	24.02	9.2210	0.0292	0.6762	0.1025
Std.dev.		1.4458	0.3881	0.2740	0.1414	2.3702	0.0120	0.2538	0.0456
95% C.I.	Lower	18	23	23	24	2.5809	0.0113	0.2713	0.0457
	Upper	24	24	24	25	13.5177	0.0456	1.0566	0.1750

computation time is more sensitive to the number of the resource constraints rather than the number of variables. Since the application problems often have much more variables than knapsack constraints, our algorithm is useful in practice.

5. Conclusions

In this paper, we study a class of continuous separable nonlinear multidimensional knapsack problems. By analyzing the structural properties of the optimal solution, we develop a multi-tier binary solution method. The proposed method has following advantages. 1) It is applicable for solving the nonlinear multidimensional knapsack problems with general structure. 2) It has computational complexity of polynomial in the number of variables.

This research can be further extended in several ways. One is to study non-separable multidimensional knapsack problems using the similar idea. Another way is to develop exact solution methods or heuristics for solving the integer multidimensional knapsack problems based on our method. Finally, the idea used in this study can be extended for investigating other complex optimization problems with multiple constraints.

Acknowledgements

This work is supported by National Natural Science Foundation of China (Grants No. 71672199).

References

[1] Bonnans, J.F. (1994) Local Analysis of Newton-Type Methods for Variational Inequalities and Nonlinear Programming. *Applied Mathematics and Optimization*, **29**, 161-186. https://doi.org/10.1007/BF01204181

[2] Coleman, T.F. and Li, Y. (1994) On the Convergence of Interior-Reflective Newton Methods for Nonlinear Minimization Subject to Bounds. *Mathematical Programming*, **67**, 189-224. https://doi.org/10.1007/BF01582221

[3] Van Hentenryck, P., Michel, L. and Benhamou, F. (1998) Constraint Programming over Nonlinear Constraints. *Science of Computer Programming*, **30**, 83-118. https://doi.org/10.1016/S0167-6423(97)00008-7

[4] Borchers, B. and Mitchell, J.E. (1994) An Improved Branch and Bound Algorithm for Mixed Integer Nonlinear Programs. *Computers & Operations Research*, **21**, 359-367. https://doi.org/10.1016/0305-0548(94)90024-8

[5] Leyffer, S. (2001) Integrating SQP and Branch-and-Bound for Mixed Integer Nonlinear Programming. *Computational Optimization and Applications*, **18**, 295-309. https://doi.org/10.1023/A:1011241421041

[6] Byrd, R.H., Hribar, M.E. and Nocedal, J. (1999) An Interior Point Algorithm for Large-Scale Nonlinear Programming. *SIAM Journal on Optimization*, **9**, 877-900. https://doi.org/10.1137/S1052623497325107

[7] Benson, H.Y., Vanderbei, R.J. and Shanno, D.F. (2002) Interior-Point Methods for Nonconvex Nonlinear Programming: Filter Methods and Merit Functions. *Computational Optimization and Applications*, **23**, 257-272. https://doi.org/10.1023/A:1020533003783

[8] Spellucci, P. (1998) An SQP Method for General Nonlinear Programs Using only Equality Constrained Subproblems. *Mathematical Programming*, **82**, 413-448. https://doi.org/10.1007/BF01580078

[9] Zhu, Z. and Zhang, K. (2004) A New SQP Method of Feasible Directions for Nonlinear Programming. *Applied Mathematics and Computation*, **148**, 121-134. https://doi.org/10.1016/S0096-3003(02)00832-9

[10] Nie, P.Y. and Ma, C.F. (2006) A Trust Region Filter Method for General Non-Linear Programming. *Applied Mathematics and Computation*, **172**, 1000-1017. https://doi.org/10.1016/j.amc.2005.03.004

[11] Nie, P.Y. (2007) Sequential Penalty Quadratic Programming Filter Methods for Nonlinear Programming. *Nonlinear Analysis: Real World Applications*, **8**, 118-129.

https://doi.org/10.1016/j.nonrwa.2005.06.003

[12] Bretthauer, K.M. and Shetty, B. (2002b) The Nonlinear Knapsack Problem-Algorithms and Applications. *European Journal of Operational Research*, **138**, 459-472. https://doi.org/10.1016/S0377-2217(01)00179-5

[13] Bretthauer, K.M. and Shetty, B. (1995) The Nonlinear Resource Allocation Problem. *Operations Research*, **43**, 670-683. https://doi.org/10.1287/opre.43.4.670

[14] Kodialam, M.S. and Luss, H. (1998) Algorithms for Separable Nonlinear Resource Allocation Problems. *Operations Research*, **46**, 272-284. https://doi.org/10.1287/opre.46.2.272

[15] Bretthauer, K.M. and Shetty, B. (2002a) A Pegging Algorithm for the Nonlinear Resource Allocation Problem. *Computers & Operations Research*, **29**, 505-527. https://doi.org/10.1016/S0305-0548(00)00089-7

[16] Zhang, B. and Hua, Z. (2008) A Unified Method for a Class of Convex Separable Nonlinear Knapsack Problems. *European Journal of Operational Research*, **191**, 1-6. https://doi.org/10.1016/j.ejor.2007.07.005

[17] Kiwiel, K.C. (2008) Breakpoint Searching Algorithms for the Continuous Quadratic Knapsack Problem. *Mathematical Programming*, **112**, 473-491. https://doi.org/10.1007/s10107-006-0050-z

[18] Sharkey, T.C., Romeijn, H.E. and Geunes, J. (2011) A Class of Nonlinear Nonseparable Continuous Knapsack and Multiple-Choice Knapsack Problems. *Mathematical Programming*, **126**, 69-96. https://doi.org/10.1007/s10107-009-0274-9

[19] Morin, T.L. and Marsten, R.E. (1976) An Algorithm for Nonlinear Knapsack Problems. *Management Science*, **22**, 1147-1158. https://doi.org/10.1287/mnsc.22.10.1147

[20] Ohtagaki, H., Iwasaki, A., Nakagawa, Y. and Narihisa, H. (2000) Smart Greedy Procedure for Solving a Multidimensional Nonlinear Knapsack Class of Reliability Optimization Problems. *Mathematical and Computer Modelling*, **31**, 283-288. https://doi.org/10.1016/S0895-7177(00)00097-2

[21] Li, D., Sun, X.L. and Wang, F.L. (2006) Convergent Lagrangian and Contour Cut Method for Nonlinear Integer Programming with a Quadratic Objective Function. *SIAM Journal on Optimization*, **17**, 372-400. https://doi.org/10.1137/040606193

[22] Li, D., Sun, X.L., Wang, J. and McKinnon, K.I. (2009) Convergent Lagrangian and Domain Cut Method for Nonlinear Knapsack Problems. *Computational Optimization and Applications*, **42**, 67-104. https://doi.org/10.1007/s10589-007-9113-1

[23] Quadri, D., Soutif, E. and Tolla, P. (2009) Exact Solution Method to Solve Large Scale Integer Quadratic Multidimensional Knapsack Problems. *Journal of Combinatorial Optimization*, **17**, 157-167. https://doi.org/10.1007/s10878-007-9105-1

[24] Wang, H., Kochenberger, G. and Glover, F. (2012) A Computational Study on the Quadratic Knapsack Problem with Multiple Constraints. *Computers & Operations Research*, **39**, 3-11. https://doi.org/10.1016/j.cor.2010.12.017

[25] Abdel-Malek, L.L. and Areeratchakul, N. (2007) A Quadratic Programming Approach to the Multi-Product Newsvendor Problem with Side Constraints. *European Journal of Operational Research*, **176**, 1607-1619. https://doi.org/10.1016/j.ejor.2005.11.002

[26] Abdel-Malek, L.L. and Otegbeye, M. (2013) Separable Programming/Duality Approach to Solving the Multi Constrained Newsboy/Gardener Problem. *Applied Mathematical Modelling*, **37**, 4497-4508. https://doi.org/10.1016/j.apm.2012.09.059

[27] Zhang, B. (2012) Multi-Tier Binary Solution Method for Multi-Product Newsvendor Problem with Multiple Constraints. *European Journal of Operational Research*, **218**, 426-434. https://doi.org/10.1016/j.ejor.2011.10.053

Appendix

A.1 Proof of Proposition 2

It is defined that $0 \le l_i < u_i$ for all $i = 1, \cdots, N$. The optimal solution to problem PR should satisfy Equation (1) and Equation (3). If $l_i \le \bar{x}_i \le u_i$, it means that the bound constraint is inactive. Therefore, we have $\hat{x}_i = \bar{x}_i$. Since $g_i(x_i)$ is increasing in x_i, and $g_i(\bar{x}_i) = 0$, we have $g_i(x_i) \ge 0$ if $x_i > \bar{x}_i$. If $\bar{x}_i < l_i < u_i$, then $g_i(x_i) \ge 0$ if $l_i \le x_i \le u_i$. Thus for any $x_i \in [l_i, u_i]$, we have $f_i(l_i) \le f_i(x_i)$, and $\hat{x}_i = l_i$. If $l_i < u_i < \bar{x}_i$, we have $\hat{x}_i = u_i$. It can be proved similar to the condition of $\bar{x}_i < l_i$.

A.2 Proof of Proposition 3

1) If $g_i(l_i) \le -\sum_{j=1}^{M} \lambda_j c_{i,j} \le g_i(u_i)$, then we have $l_i \le h_i\left(-\sum_{j=1}^{M} \lambda_j c_{i,j}\right) \le u_i$, and $w_i = v_i = 0$, which implies $x_i(\boldsymbol{\lambda}) = h_i\left(-\sum_{j=1}^{M} \lambda_j c_{i,j}\right)$. If $-\sum_{j=1}^{M} \lambda_j c_{i,j} < g_i(l_i)$, then we have $g_i(x_i) + \sum_{j=1}^{M} \lambda_j c_{i,j} \ge g_i(l_i) + \sum_{j=1}^{M} \lambda_j c_{i,j} > 0$, and hence $w_i > 0$, $x_i(\boldsymbol{\lambda}) = l_i$.

If $-\sum_{j=1}^{M} \lambda_j c_{i,j} > g_i(u_i)$, we have $g_i(x_i) + \sum_{j=1}^{M} \lambda_j c_{i,j} \le g_i(u_i) + \sum_{j=1}^{M} \lambda_j c_{i,j} < 0$, which means $v_i > 0$, and $x_i(\boldsymbol{\lambda}) = u_i$. Therefore, we have

$$x_i(\boldsymbol{\lambda}) = \begin{cases} l_i, & \text{if } -\sum_{j=1}^{M} \lambda_j c_{i,j} < g_i(l_i), \\ h_i\left(-\sum_{j=1}^{M} \lambda_j c_{i,j}\right), & \text{if } g_i(l_i) \le -\sum_{j=1}^{M} \lambda_j c_{i,j} \le g_i(u_i), \\ u_i, & \text{if } -\sum_{j=1}^{M} \lambda_j c_{i,j} > g_i(u_i). \end{cases} \tag{A1}$$

2) $\lambda_j = 0$ or $\sum_{i=1}^{N} c_{i,j} x_i(\boldsymbol{\lambda}) = C_j$ implies

$$\lambda_j \left(\sum_{i=1}^{N} c_{i,j} x_i(\boldsymbol{\lambda}) - C_j \right) = 0, j = 1, \cdots, M.$$

Because $x(\lambda)$ satisfies Equation (5) and Equation (6), $\boldsymbol{x}(\boldsymbol{\lambda})$ will satisfy all KKT conditions. Therefore, $\boldsymbol{x}^* = \boldsymbol{x}(\boldsymbol{\lambda})$ if $(\boldsymbol{x}(\boldsymbol{\lambda}), \boldsymbol{\lambda})$ satisfies $\lambda_j = 0$ or $\sum_{i=1}^{N} c_{i,j} x_i = C_j$, $j = 1, \cdots, M$.

A.3 Proof of Proposition 4

1) $\lambda_M = 0$ or $\sum_{i=1}^{N} c_{i,M} x_i(\lambda_M) = C_M$ implies $\lambda_M \left(\sum_{i=1}^{N} c_{i,M} x_i - C_M\right) = 0$. Since $(\boldsymbol{x}(\lambda_M), \lambda_M)$ satisfies Equation (5) and Equation (6), it will satisfy all KKT conditions. Therefore, $\boldsymbol{x}^* = \boldsymbol{x}(\lambda_M)$ if $(\boldsymbol{x}(\lambda_M), \lambda_M)$ satisfies $\lambda_M = 0$ or $\sum_{i=1}^{N} c_{i,M} x_i(\lambda_M) = C_M$.

2) KKT conditions for problem $P\left(\hat{\boldsymbol{f}}(\lambda_M), M-1\right)$ are

$$\frac{\mathrm{d}\hat{f}_i(x_i)}{\mathrm{d}x_i} + \sum_{j=1}^{M-1} \lambda_j c_{i,j} - w_i + v_i = 0, i = 1, \cdots, N, \tag{A2}$$

$$\sum_{i=1}^{N} w_i (x_i - l_i) + \sum_{i=1}^{N} v_i (x_i - u_i) = 0, \tag{A3}$$

$$\lambda_j\left(\sum_{i=1}^{N}c_{i,j}x_i-C_j\right)=0, j=1,\cdots,M-1. \tag{A4}$$

Notice that $\hat{f}_i(x_i)$ is a parameter-adjusted function of $f_i(x_i)$, with $\hat{f}_i(\lambda_M)=f_i+\lambda_M c_{i,M}x_i$. These conditions in Equations (A2)-(A3) are the same as KKT conditions given in Equations (5)-(7) without $\lambda_M\left(\sum_{i=1}^{N}c_{i,M}x_i-C_M\right)=0$. Since $\boldsymbol{x}(\lambda_M)$ is the optimal solution of the KKT conditions in Equations (5)-(7) without $\lambda_M\left(\sum_{i=1}^{N}c_{i,M}x_i-C_M\right)=0$, it must be the optimal solution to problem $P\left(\hat{\boldsymbol{f}}(\lambda_M),M-1\right)$.

A.4 Proof of Proposition 5

Let $\bar{\lambda}_M=\max\left(0,\max_{i=1,\cdots,N}\left\{-g_i(l_i)/c_{i,M}\right\}\right)$. If $\lambda_M^*>\bar{\lambda}_M$, then we have $\lambda_M^*c_{i,M}>-g_i(l_i), i=1,\cdots,N$. From Equation (5), we have

$$w_i^*=g_i(x_i)+\sum_{j=1}^{M}\lambda_j c_{i,j}+v_i>g_i(x_i)+\lambda_M^*c_{i,M}+v_i>0,\ i=1,\cdots,N. \tag{A5}$$

Since $w_i>0$, from Equation (6), we know $x_i=l_i$ and $v_i=0$. Thus, we have

$$\lambda_M^*\left(\sum_{i=1}^{N}c_{i,M}x_i^*-C_M\right)=\lambda_M^*\left(\sum_{i=1}^{N}c_{i,M}l_i-C_M\right)\neq 0. \tag{A6}$$

Equation (A6) violates the slackness condition $\lambda_M\left(\sum_{i=1}^{N}c_{i,M}x_i-C_M\right)=0$ in Equation (7). Therefore, there must be $\lambda_M^*<\bar{\lambda}_M$.