Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 13, No. 2, pp. 301–313, April 2017

A LINEAR TIME ALGORITHM FOR THE CONTINUOUS QUADRATIC KNAPSACK PROBLEM WITH ℓ_1 REGULARIZATION*

Yong-Jin Liu† and Yanan Wen

Abstract: Recent interests in sparse optimization arising from kernel density learning, sparse portfolio selection, and index tracking motivate us to consider the continuous quadratic knapsack problem with ℓ_1 regularization (CQKPL1) which favors sparsity. In this paper, we aim to develop an efficient algorithm based on median search strategy to solve CQKPL1. A remarkable feature of the algorithm is that the closed-form solution of CQKPL1 would be achieved with order n operations. Numerical results validate the linear time complexity of our algorithm and the sparsity of optimal solution, as well as demonstrate that our algorithm is able to efficiently solve CQKPL1 with the problem sizes up to ten million within about two minutes.

Key words: *continuous quadratic knapsack problem, ℓ_1 regularization, linear time complexity, median search method*

Mathematics Subject Classification: *90C20, 90C25*

1 Introduction

This paper considers a class of convex nonsmooth optimization problems, called the continuous quadratic knapsack problem with ℓ_1 regularization (abbreviated as CQKPL1):

$$\begin{array}{ll} \min & \frac{1}{2}\|x-c\|^2 + \|Wx\|_1 \\ \text{s.t.} & a^T x = b, \\ & l \le x \le u, \end{array} \tag{P}$$

where $x \in \Re^n$ is the vector of variables, the data $c, a, l, u \in \Re^n, w \in \Re^n_{++}$ and $b \in \Re$ are given, and W is the diagonal matrix with the entries $w_i > 0, i = 1, 2, \ldots, n$ on the diagonal, here $\|\cdot\|$ and $\|\cdot\|_1$ respectively stand for the Euclidean and ℓ_1 norm in $\Re^n$. We first assume that $l < u$, and possibly $l_i = -\infty$ or $u_i = +\infty$. Without loss of generality, we further assume that $a > 0$, because if $a_k < 0$, then we carry out the transformation $(x_k, c_k, a_k, l_k, u_k) \leftarrow (-x_k, -c_k, -a_k, -u_k, -l_k)$ and essentially solve the same form as problem (P), whereas if

*The first author is supported by the National Natural Science Foundation of China under Grants 11001180 and 11371255, the Program for Liaoning Excellent Talents in University under Grant LR2015047, and the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry.

†Corresponding author

$a_k = 0$, then the optimal value of x_k is the minimizer of $\min\{\frac{1}{2}(x_k - c_k)^2 + w_k|x_k| : l_k \le x_k \le u_k\}$ which has an analytic expression (see Section 2) and hence x_k may be removed from problem (P).

In the absence of ℓ_1 regularization term, problem (P) is referred to as the continuous quadratic knapsack problem (abbreviated as CQKP) which is well known to appear in a wide variety of applications such as resource allocation [21], support vector machine [8], portfolio selection [1], and so on. Much research and a large number of algorithms have been developed to study CQKP. The existing algorithms mainly fall into three broad categories: (i) *bisection or median search methods*, see, e.g., Helgason et al. [11], Brucker [4], Calamai and Moré [5], Cosares and Hochbaum [7], Hochbaum and Hong [12], Kiwiel [14,15], Maculan and de Paula [17], Maculan et al. [18], Pardalos and Kovoor [20]; (ii) *pegging/variable fixing methods*, see, e.g., Bitran and Hax [2], Bretthauer and Shetty [3], Kiwiel [16], Maculan et al. [18], Robinson et al. [22]; and (iii) *Newton-type methods*, see, e.g., Dai and Fletcher [8], Cominetti et al. [6], Davis et al. [9]. We should remark that the literature on developments and algorithms for solving CQKP is vast and here we only mention some of them. For more applications, algorithms and developments of CQKP, we refer the reader to an excellent survey paper of Patriksson [21] and the second chapter of Jeong's PhD dissertation [13], and the references therein.

Due to the demand of many practical applications such as kernel density learning [19], sparse portfolio selection [1], and index tracking [26,27], the corresponding sparse optimization has received increasing attention, see, e.g., Xu et al. [25,26] for the cardinality or ℓ_0 constrained optimization problem. As broad applications of the ℓ_1 regularization in the fields of machine learning, compressive sensing, signal and image processing convince us that the ℓ_1 regularization induces the sparsity of optimization problems, we are motivated to consider CQKP with ℓ_1 regularization (i.e., CQKPL1) intending to find a sparse solution.

It should be noted that CQKPL1 can equivalently reformulated as the convex quadratic programming (QP) of the following form:

$$\begin{array}{ll} \min & \dfrac{1}{2}\|x-c\|^2 + \displaystyle\sum_{i=1}^{n}(w_i x_i + 2y_i) \\ \text{s.t.} & a^T x = b, \\ & Wx + y \ge 0, \\ & l \le x \le u,\ y \ge 0. \end{array} \tag{1.1}$$

Thus one may apply efficient algorithms in the literature, which have been proposed for general convex QP problems, to solve (1.1). However, as far as we can see, the QP reformation (1.1) is computationally expensive because it has more $2n$ linear inequalities and one more n-dimensional variable than the original problem (P). The computational cost and memory requirement of the QP reformation (1.1) are especially unnecessarily high when n is large. Unlike the QP reformation (1.1), in this paper we propose an algorithm which does not destroy the specific structure of CQKPL1. In particular, we first derive the simple computation of $\min\{\frac{1}{2}\|x - x_0\|^2 + \|Wx\|_1 : l \le x \le u\}$ and then propose an efficient algorithm in which we only need to deal with an univariate equation.

With nonsmooth ℓ_1 term, CQKPL1 becomes more complicated than CQKP. One may doubt that the algorithms developed for solving CQKP are not applicable to CQKPL1. Our study reveals that the algorithms for CQKP can be also extended to solve CQKPL1 with some modifications. The purpose of this paper is to consider the median search based linear time algorithm proposed by Brucker [4] and Calamai et al. [5] among others to find the closed-form of optimal solution to CQKPL1. The work of this paper is built upon a key result that problem (P) without linear equality constraint admits an analytic expression of the optimal solution within $O(n)$ operations (see Proposition 2.1). After using dual reformulation, we may apply median search algorithm to find a zero point of a one-dimensional monotone piecewise linear function. Our analysis shows that the median search algorithm for solving CQKPL1 has the linear time complexity of $O(n)$ which is the same as for CQKP. We efficiently implement our algorithm in MATLAB Language and perform numerical experiments for randomly generated CQKPL1 in order to test the complexity and performance of our algorithm. Numerical results not only validate the theoretical results on linear time complexity of our algorithm and the sparsity of the optimal solution to CQKPL1, but also demonstrate that our algorithm is able to efficiently solve CQKPL1 with the problem sizes up to ten million within about two minutes that outperforms favourably in terms of the CPU time the state-of-the-art standard solver Gurobi [10] via the QP reformulation (1.1).

The remainder of this paper are organized as follows. In Section 2, we shall develop some preliminary results which play a crucial role in designing an efficient algorithm. Section 3 is devoted to presenting a linear time complexity of algorithm for solving CQKPL1. Numerical tests are performed in Section 4 to validate the feature of our algorithm as well as to evaluate the performance of our algorithm. We make final conclusions in Section 5.

Notation. The following notations will be used throughout this paper. For an index set $\mathcal{I}$, we use $|\mathcal{I}|$ to denote the cardinality of $\mathcal{I}$ and $x_{\mathcal{I}} \in \Re^{|\mathcal{I}|}$ to denote the sub-vector of x obtained by removing all the components of x not in $\mathcal{I}$. We denote $\mathrm{mid}(l, x, u)$ as the component-wise median of three vectors. Similarly, for given $w > 0$, we let $\mathrm{mid}(l, x + w, 0, x - w, u)$ stand for the vector whose i-th component is defined by

$$\mathrm{mid}(l_i, x_i + w_i, 0, x_i - w_i, u_i) = \begin{cases} l_i, & x_i + w_i \le l_i, \\ x_i + w_i, & l_i < x_i + w_i < 0, \\ 0, & -w_i \le x_i \le w_i, \\ x_i - w_i, & 0 < x_i - w_i < u_i, \\ u_i, & x_i - w_i \ge u_i. \end{cases}$$

For a vector or set x, we use the conventional notation $\mathrm{median}(x)$ to denote the median value of the elements in x.

2 Preliminaries

In this section, we shall develop some preliminary results which are crucial to design an efficient algorithm. We begin with the study on analytic expression of the optimal solution to problem (P) without linear equality constraint. We then consider Lagrangian dual reformulation of CQKPL1.

2.1 Problem without linear constraint

In this subsection, we aim to derive the analytic expression of the optimal solution to problem (P) without linear equality constraint.

For any $y \in \Re^n$, we consider the optimal solution to problem (P) without linear equality constraint, i.e.,

$$\begin{aligned} \min \quad & \frac{1}{2}\|x - y\|^2 + \|Wx\|_1 \\ \text{s.t.} \quad & l \le x \le u. \end{aligned} \tag{2.1}$$

For a special case of problem (2.1) without ℓ_1 regularization, it is well known that the unique optimal solution is the metric projection operator onto the box sets $\{x \in \Re^n : l \leq x \leq u\}$, which is given by $\mathrm{mid}(l, x, u)$. For another special case of problem (2.1) without lower and upper bound constraints, it is also known that the unique optimal solution is referred to as the soft thresholding operator computed by $\mathrm{sign}(y)\max\{|y| - w, 0\}$, which is commonly used in developing efficient algorithms in the fields of compressive sensing, signal processing, and statistical learning.

Next, we would like to show that the optimal solution to general case of problem (2.1) surprisingly admits an analytic expression. Due to the separability of problem (2.1), one only needs to consider the following univariate problem:

$$\begin{array}{ll} \min & \frac{1}{2}(s-t)^2 + \tau|s| \\ \text{s.t.} & \alpha \leq s \leq \beta, \end{array} \tag{2.2}$$

where the real numbers $t, \alpha, \beta, \tau > 0$ are given. Let s^* be the optimal solution to problem (2.2). Then, by considering the following three cases: (i) $\alpha \geq 0$, (ii) $\beta \leq 0$, and (iii) $\alpha < 0$ and $\beta > 0$, after simple manipulations we can readily derive the following analytic expression of the unique optimal solution s^* to problem (2.2):

$$s^* = \begin{cases} \mathrm{mid}\{\alpha, t-\tau, \beta\}, & \alpha \geq 0, \\ \mathrm{mid}\{\alpha, t+\tau, \beta\}, & \beta \leq 0, \\ \mathrm{mid}\{\alpha, t+\tau, 0, t-\tau, \beta\}, & \text{otherwise.} \end{cases} \tag{2.3}$$

Since problem (2.1) is equivalent to finding the solutions to each of univariate problems (2.2) separately, i.e.,

$$\begin{array}{ll} \min & \frac{1}{2}(x_i - y_i)^2 + w_i|x_i| \\ \text{s.t.} & l_i \leq x_i \leq u_i \end{array}, \quad i = 1, 2, \ldots, n,$$

from (2.3) we easily obtain the results on analytic expression of the unique optimal solution to problem (2.1) in the following proposition.

Proposition 2.1. *Assume that $y, l, u \in \Re^n$ and $w \in \Re^n_{++}$ are given. Then, the unique optimal solution to problem (2.1) can be explicitly computed by $x^* = (x_1^*, x_2^*, \ldots, x_n^*)^T$, where for any $i = 1, 2, \ldots, n$, x_i^* is defined as*

$$x_i^* = \begin{cases} \mathrm{mid}\{l_i, y_i - w_i, u_i\}, & l_i \geq 0, \\ \mathrm{mid}\{l_i, y_i + w_i, u_i\}, & u_i \leq 0, \\ \mathrm{mid}\{l_i, y_i + w_i, 0, y_i - w_i, u_i\}, & \textit{otherwise.} \end{cases} \tag{2.4}$$

2.2 Lagrangian dual reformulation

In this subsection, we shall consider Lagrangian dual reformulation of CQKPL1 and present some useful properties associated with the optimal solution of CQKPL1 and its dual.

The Lagrangian function of CQKPL1 is defined as

$$L(x; \lambda) := \frac{1}{2}\|x - c\|^2 + \|Wx\|_1 + \lambda(a^T x - b).$$

Thus, it follows from the dual theory (see, e.g., [23, 24]) that the Lagrangian dual problem associated with problem (P) is as follows:

$$\max_{\lambda \in \Re} \; d(\lambda), \tag{D}$$

where the objective $d(\cdot)$ is defined as the optimal value function of the following strongly convex optimization problem:

$$\inf_{l \le x \le u} \left\{ L(x;\lambda) = \frac{1}{2}\|x-c\|^2 + \|Wx\|_1 + \lambda(a^T x - b) \right\},$$

or equivalently,

$$\begin{array}{ll} \min & \frac{1}{2}\|x-(c-\lambda a)\|^2 + \|Wx\|_1 + \lambda a^T c - \frac{1}{2}\lambda^2\|a\|^2 - \lambda b \\ \text{s.t.} & l \le x \le u. \end{array} \tag{2.5}$$

Let us denote $x^*(\lambda)$ as the unique optimal solution to problem (2.5). Then, from Proposition 2.1, we obviously obtain the following results on the characterization of $x^*(\lambda)$.

Proposition 2.2. *Let $c, l, u \in \Re^n$ and $a, w \in \Re^n_{++}$ be given. Then, for any given $\lambda \in \Re$, $x_i^*(\lambda), i = 1, 2, \ldots, n$ are given by*

$$x_i^*(\lambda) = \begin{cases} \operatorname{mid}\{l_i, c_i - w_i - \lambda a_i, u_i\}, & l_i \ge 0, \\ \operatorname{mid}\{l_i, c_i + w_i - \lambda a_i, u_i\}, & u_i \le 0, \\ \operatorname{mid}\{l_i, c_i + w_i - \lambda a_i, 0, c_i - w_i - \lambda a_i, u_i\}, & \textit{otherwise}. \end{cases} \tag{2.6}$$

Remark 2.3. From Proposition 2.2, we can easily see that for any $i = 1, 2, \ldots, n$, the function $x_i^*(\cdot)$ is piecewise linear. Moreover, it is also monotonically nonincreasing on $\Re$ due to $a > 0$.

Let us define three index subsets of $\{1, 2, \ldots, n\}$ as

$$\mathcal{I}_1 := \{i : l_i \ge 0\},\; \mathcal{I}_2 := \{i : u_i \le 0\},\; \mathcal{I}_3 := \{i : l_i < 0 \text{ and } u_i > 0\}. \tag{2.7}$$

It follows from Proposition 2.2 that the function ψ has at most $2|\mathcal{I}_1| + 2|\mathcal{I}_2| + 4|\mathcal{I}_3|$ breakpoints, which are indicated as follows:

$$\begin{aligned} t_i^U &:= \frac{c_i - w_i - u_i}{a_i},\; t_i^L := \frac{c_i - w_i - l_i}{a_i},\; i \in \mathcal{I}_1, \\ t_i^U &:= \frac{c_i + w_i - u_i}{a_i},\; t_i^L := \frac{c_i + w_i - l_i}{a_i},\; i \in \mathcal{I}_2, \\ t_i^U &:= \frac{c_i - w_i - u_i}{a_i},\; s_i^U := \frac{c_i - w_i}{a_i},\; s_i^L := \frac{c_i + w_i}{a_i},\; t_i^L := \frac{c_i + w_i - l_i}{a_i},\; i \in \mathcal{I}_3. \end{aligned}$$

We remark that CQKPL1 has more $2|\mathcal{I}_3|$ breakpoints compared to CQKP. From (2.6), we note that for any λ within each interval containing no breakpoints, the structure of $x^*(\lambda)$ is unchanged.

We now turn to characterize the condition about $\lambda \in \Re$ under which the optimal solution $x^*(\lambda)$ of problem (2.5) with λ is also optimal for CQKPL1. To achieve this aim, we define the function $\psi : \Re \to \Re$ by

$$\psi(\lambda) := a^T x^*(\lambda) - b. \tag{2.8}$$

By the Saddle Point Theorem (see, e.g., [23, Theorem 28.3]), one easily obtain the following proposition.

Proposition 2.4. *Let $c, l, u \in \Re^n$ and $a, w \in \Re^n_{++}$ be given. Let the function ψ be defined by (2.8). Then, $x^*(\lambda^*)$ is the unique optimal solution to CQKPL1 for any $\lambda^* \in \Re$ such that $\psi(\lambda^*) = 0$.*

Proposition 2.4 tells us that the optimal solution to CQKPL1 can be computed by solving a one-dimensional equation $\psi(\lambda) = 0$. Thus, to solve CQKPL1, our main task is to develop efficient algorithm to solve $\psi(\lambda) = 0$. Before continuing, we present some related properties of the function $\psi(\cdot)$.

Proposition 2.5. *The function ψ defined as in (2.8) is piecewise linear and monotonically nonincreasing.*

Proof. We note that the function ψ can be rewritten as $\psi(\lambda) = \sum_{i=1}^n a_i x_i^*(\lambda) - b$. Since $x_i^*(\cdot), i = 1, 2, \ldots, n$ are piecewise linear functions, it follows from the definition of ψ that ψ is a piecewise linear function. Moreover, due to the monotonically nonincreasing properties of $x_i^*(\cdot), i = 1, 2, \ldots, n$, combining with $a > 0$, we know that ψ is a monotonically nonincreasing function. This completes the proof. □

Let us define:

$$\lambda_{\min} := \min\{t_i^U : i = 1, 2 \ldots, n\}, \ \lambda_{\max} := \max\{t_i^L : i = 1, 2, \ldots, n\}. \tag{2.9}$$

Then, it must hold that $\psi(\lambda_{\min}) \geq 0$ and $\psi(\lambda_{\max}) \leq 0$ because otherwise, by Proposition 2.5 we would know that problem (P) is infeasible. If $\psi(\lambda_{\min}) = 0$ or $\psi(\lambda_{\max}) = 0$, we fortunately get the desired value λ^*. Otherwise, the continuity of the function ψ implies that the bracket $(\lambda_{\min}, \lambda_{\max})$ contains a solution λ^* of the equation $\psi(\lambda) = 0$. In the next section we will propose a strategy for updating $\lambda_{\min}$ or $\lambda_{\max}$ so that the bracket $(\lambda_{\min}, \lambda_{\max})$ is uniformly reduced, which leads to linear time algorithm for solving CQKPL1.

3 Linear time algorithm

This section is devoted to giving the detailed description of a linear time algorithm for solving CQKPL1, which is based on median search method. It should be emphasized here that the algorithm described below is essentially modifications of the one proposed by Brucker [4] for CQKP.

Details of the algorithm can be roughly depicted as follows. Let $\lambda_{\min}$ and $\lambda_{\max}$ be defined as in (2.9). We assume that $\psi(\lambda_{\min}) > 0$ and $\psi(\lambda_{\max}) < 0$ (otherwise, $\lambda^* = \lambda_{\min}$ or $\lambda^* = \lambda_{\max}$). Then, it holds that

$$\lambda^* \in [\lambda_{\min}, \lambda_{\max}]. \tag{3.1}$$

In the algorithm, the bracket $[\lambda_{\min}, \lambda_{\max}]$ is uniformly reduced by updating $\lambda_{\min}$ or $\lambda_{\max}$ so that the property (3.1) is always satisfied until it contains no breakpoints. It then follows that ψ is linear on the final bracket $[\lambda_{\min}, \lambda_{\max}]$ and hence the solution λ^* of the equation $\psi(\lambda) = 0$ is easily computed by

$$\lambda^* = \lambda_{\min} - (\psi(\lambda_{\min}) - b)\frac{\lambda_{\max} - \lambda_{\min}}{\psi(\lambda_{\max}) - \psi(\lambda_{\min})}.$$

It still remains vague how the algorithm updates $\lambda_{\min}$ and $\lambda_{\max}$. We incorporate the median search strategy into the algorithm. In particular, at each stage of the algorithm, we choose λ as the median value of the breakpoints in $[\lambda_{\min}, \lambda_{\max}]$ and calculate $\psi(\lambda)$. We then update $\lambda_{\min}$ and $\lambda_{\max}$ as follows: If $\psi(\lambda) > 0$, then set $\lambda_{\min} := \lambda$; If $\psi(\lambda) < 0$, then set $\lambda_{\max} := \lambda$; Otherwise, $\lambda^* = \lambda$.

We are in the position to formally state a linear time algorithm for solving CQKPL1, which has the following template.

Algorithm MSLTA: A median search based linear time algorithm for solving CQKPL1.

0. If $a^T l < b$ or $a^T u > b$, then stop and CQKPL1 is infeasible. Otherwise, set $\mathcal{I} := \{1, 2, \ldots, n\}$ and $\mathcal{IC} := \emptyset$. Let $\lambda_{\min} := \min(t^U_{\mathcal{I}})$ and $\lambda_{\max} := \max(t^L_{\mathcal{I}})$.

1. If $\mathcal{I} \neq \emptyset$, compute $t^L := \text{median}(\{t^L_{\mathcal{I}}\}),\ t^U := \text{median}(\{t^U_i : t^L_i \geq t^L, i \in \mathcal{I}\})$. Otherwise, compute $t^L := \text{median}(\{s^L_{\mathcal{IC}}\}),\ t^U := \text{median}(\{s^U_i : s^L_i \geq t^L, i \in \mathcal{IC}\})$.

2. For $\lambda = t^L, t^U$. If $\lambda_{\min} < \lambda < \lambda_{\max}$, then calculate $\psi(\lambda)$. If $\psi(\lambda) = 0$, then stop; else, if $\psi(\lambda) > 0$, set $\lambda_{\min} := \max(\lambda_{\min}, \lambda)$; if $\psi(\lambda) < 0$, set $\lambda_{\max} := \min(\lambda_{\max}, \lambda)$.

3. Eliminate the index i from $\mathcal{I} \cup \mathcal{IC}$ if the structure of $x^*_i(\lambda)$ is unchanged for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$. Add the index i in $\mathcal{I} \cap \mathcal{I}_3$ into $\mathcal{IC}$ if $[\lambda_{\min}, \lambda_{\max}] \subset [t^U_i, t^L_i]$ but the structure of $x^*_i(\lambda)$ is not determined for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$.

4. If $\mathcal{I} = \emptyset$ and $\mathcal{IC} = \emptyset$, then stop. Otherwise, go to step 1.

To understand the procedure of Algorithm MSLTA, we below give some more detailed descriptions on step 3.

For given $\mathcal{I} \neq \emptyset$, we define some index subsets of $\mathcal{I}$ by

$$\begin{aligned} \mathcal{IL} &:= \{i \in \mathcal{I} : t^L_i \leq \lambda_{\min}\}, \\ \mathcal{IU} &:= \{i \in \mathcal{I} : \lambda_{\max} \leq t^U_i\}, \\ \mathcal{IM} &:= \{i \in \mathcal{I} : t^U_i < \lambda_{\min},\ \lambda_{\max} < t^L_i\} \end{aligned}$$

and three index subsets of $\mathcal{IM}$ by

$$\mathcal{IM}1 := \mathcal{IM} \cap \mathcal{I}_1,\ \mathcal{IM}2 := \mathcal{IM} \cap \mathcal{I}_2,\ \mathcal{IM}3 := \mathcal{IM} \cap \mathcal{I}_3,$$

where $\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3$ are defined as in (2.7). Then, by virtue of Proposition 2.2, we can eliminate elements from the index set $\mathcal{I}$ by:

(1) $\mathcal{I} := \mathcal{I} \setminus \mathcal{IL}$, because $x^*_i(\lambda) = l_i$, $i \in \mathcal{IL}$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$;

(2) $\mathcal{I} := \mathcal{I} \setminus \mathcal{IU}$, because $x^*_i(\lambda) = u_i$, $i \in \mathcal{IU}$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$;

(3) $\mathcal{I} := \mathcal{I} \setminus \mathcal{IM}$, because

- (i) $x^*_i(\lambda) = c_i - w_i - \lambda a_i$, $i \in \mathcal{IM}1$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$;
- (ii) $x^*_i(\lambda) = c_i + w_i - \lambda a_i$, $i \in \mathcal{IM}2$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$;
- (iii) $\mathcal{IC} := \mathcal{IC} \cup \mathcal{IM}3$.

For each i in the index set $\mathcal{IC}$, we have to make additional effort to identify the structure of $x_i^*(\lambda)$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$. To see this, we define the following index subsets of $\mathcal{IC}$:

$$\begin{aligned}\mathcal{ICL} &:= \{i \in \mathcal{IC} : s_i^L \le \lambda_{\min}\},\\ \mathcal{ICU} &:= \{i \in \mathcal{IC} : \lambda_{\max} \le s_i^U\},\\ \mathcal{ICM} &:= \{i \in \mathcal{IC} : s_i^U < \lambda_{\min},\ \lambda_{\max} < s_i^L\}.\end{aligned}$$

Similarly, again by virtue of Proposition 2.2, we can eliminate elements from the index set $\mathcal{IC}$ by considering the following three cases:

(1) $\mathcal{IC} := \mathcal{IC} \setminus \mathcal{ICL}$, because $x_i^*(\lambda) = c_i + w_i - \lambda a_i$, $i \in \mathcal{ICL}$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$;

(2) $\mathcal{IC} := \mathcal{IC} \setminus \mathcal{ICU}$, because $x_i^*(\lambda) = c_i - w_i - \lambda a_i$, $i \in \mathcal{ICU}$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$;

(3) $\mathcal{IC} := \mathcal{IC} \setminus \mathcal{ICM}$, because $x_i^*(\lambda) = 0$, $i \in \mathcal{ICM}$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$.

Concerning step 2, we wish to show that one can evaluate $\psi(\lambda)$ efficiently with order $|\mathcal{I}| + |\mathcal{IC}|$ operations. Indeed, by setting the initial values $sLU = 0, sIm = 0, sImt = 0$, we update them by

$$\begin{aligned}sLU &= sLU + \sum_{i\in\mathcal{IL}} a_i l_i + \sum_{i\in\mathcal{IU}} a_i u_i,\\ sIm &= sIm + \sum_{i\in\mathcal{IM}1\cup\mathcal{ICL}} a_i(c_i + w_i) + \sum_{i\in\mathcal{IM}2\cup\mathcal{ICU}} a_i(c_i - w_i),\\ sImt &= sImt + \sum_{i\in\mathcal{IM}1\cup\mathcal{IM}2\cup\mathcal{ICL}\cup\mathcal{ICU}} a_i^2.\end{aligned}$$

Then, we can obtain that for any $\lambda \in [\lambda_{\min}, \lambda_{\max}]$,

$$a^T x^*(\lambda) = sLU + sIm - \lambda \cdot sImt + \sum_{i\in\mathcal{I}\cup\mathcal{IC}} a_i x_i^*(\lambda). \tag{3.2}$$

This shows that we can evaluate $\psi(\lambda)$ efficiently with order $|\mathcal{I}| + |\mathcal{IC}|$ operations.

Remark 3.1. At each stage of Algorithm MSLTA, the sums $a_i x_i^*(\lambda)$ over $i \in \mathcal{ICM}$ are omitted because $x_i^*(\lambda) = 0$ for all $\lambda \in [\lambda_{\min}, \lambda_{\max}]$. Numerical experiments show that this helps us to save much time especially when the optimal solution to CQKPL1 is very sparse.

We finally remark that Algorithm MSLTA terminates when $\psi(\lambda) = 0$ for some λ at step 2, or when $\mathcal{I} = \emptyset, \mathcal{IC} = \emptyset$ at step 4. In the latter case, λ^* can be computed by

$$\lambda^* = (sLu + sIm - b)/sImt.$$

Before closing this section, we shall analyze the complexity of Algorithm MSLTA which has linear time complexity. We only carry out simple analysis here since the details can be found in several papers (see, e.g., [4, 5]).

Theorem 3.2. *Algorithm MSLTA has a linear time complexity of $O(n)$.*

Proof. At step 1, the median element of $\mathcal{I}$ (or $\mathcal{IC}$) can be achieved in at most $O(|\mathcal{I}|)$ (or $O(|\mathcal{IC}|)$) operations since median(S) can be calculated in at most $O(|S|)$ steps. At step 2, it follows from (3.2) that the evaluation of $\psi(\lambda)$ requires order $|\mathcal{I}| + |\mathcal{IC}|$ operations. At step 3, each iteration reduces $|\mathcal{I}|$ (if $\mathcal{I} \ne \emptyset$) or $|\mathcal{IC}|$ (if $\mathcal{I} = \emptyset$) at least by half. Thus, Algorithm MSLTA admits linear time complexity. This completes the proof. □

4 Numerical experiments

In this section, we perform our numerical experiments to verify the linear time complexity of Algorithm MSLTA and explore the sparsity of the optimal solution to CQKPL1.

Algorithm MSLTA for solving CQKPL1 is implemented in `MATLAB` language. All numerical experiments are performed on a Laptop of Intel Core i7-3520M CPU 2.9GHz with 8GB RAM memory, running Windows 10 and `MATLAB R2013b`.

The first test we shall perform is to confirm the linear time complexity of Algorithm MSLTA and compare it with the state-of-the-art standard solver, called Gurobi [10]. To perform this test, we randomly generate all the parameters of CQKPL1 at uniform distribution in the following way.

Example 4.1. The parameters $c, w, a \in \Re^n$ are randomly generated at uniform distribution with all entries in the interval $[-2, 2]$, $[0.5, 1.5]$, $[-1, 1]$, respectively. The parameters $l, u \in \Re^n$ are also randomly generated with all entries uniformly distributed between $[-0.3, 0.7]$, $[1, 2]$ respectively such that $l < u$. The scalar b is generated by $(a^T l + a^T u)/2$ to ensure feasibility.

In this test, we compare Algorithm MSLTA with the standard solver Gurobi [10] via the QP reformulation (1.1). The parameters for Gurobi are set to be default values. We perform numerical experiments with problem sizes varying from one thousand up to one million. For each problem size, we test 10 instances and record the average running times. It should be noted that we transform these problems into the form of CQKPL1 with $a > 0$ before running our algorithm. The running times are inclusive of times spent in problem transformation.

Figure 1: Numerical results for Example 4.1.

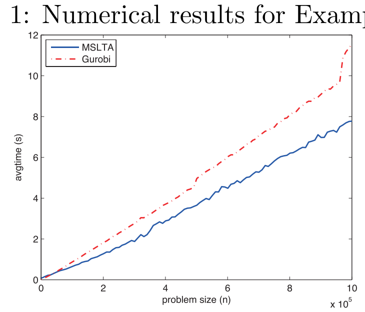

Figure 1 reports the average running times (avgtime) of ten runs in second format. As shown in Figure 1, Algorithm MSLTA is able to find the analytic solution to each instance with the problem sizes up to one million less than 8 seconds in terms of the CPU time. An analysis of the results highlights that the average running times grow linearly with the problem sizes in accordance with the results of Theorem 3.2 on the linear time complexity of Algorithm MSLTA. From Figure 1, we can also see that Algorithm MSLTA outperforms favourably Gurobi in terms of the CPU time taken to solve the instances with problem size greater than 5000, while Algorithm MSLTA finds the exact solutions and Gurobi achieves the approximate solutions.

From this experiment, we further observe that Algorithm MSLTA solves the instance with $n = 5 \times 10^6$ in only about 45 seconds, while Gurobi takes about 72 seconds to solve it. When n is set to be 10^7, our algorithm is able to solve it in about 100 seconds, but Gurobi fails to solve it due to excessive computer memory required.

Table 1: Numerical results for Example 4.2.

Problem			**Results**					
n	wc	nr	Number of zeros in *optx*			Time (s)		
			min	max	avg	min	max	avg
1000	0.1	456	118	153	137	0.0502	0.0937	0.0712
	1.0	455	323	367	341	0.0521	0.0993	0.0732
	5.0	458	423	481	447	0.0694	0.1038	0.0883
10000	0.1	4543	1280	1409	1356	0.0954	0.1641	0.1309
	1.0	4538	3341	3496	3396	0.1065	0.1669	0.1403
	5.0	4531	4366	4600	4487	0.1257	0.1869	0.1573
100000	0.1	45484	13462	13729	13590	0.5120	0.7171	0.6257
	1.0	45495	33905	34396	34101	0.6289	0.8306	0.7474
	5.0	45501	45060	45777	45316	0.6605	0.7966	0.7366
1000000	0.1	454995	135821	136877	136463	7.3727	8.6366	7.9873
	1.0	455119	340323	342287	341344	8.9849	15.0477	10.6590
	5.0	455014	453225	455534	454280	8.2922	9.9414	9.2788
10000000	0.1	4550032	1363248	1367949	1365070	87.0112	106.7948	96.0116
	1.0	4550441	3411409	3414545	3413073	108.1301	134.7835	122.9104
	5.0	4550232	4544023	4551070	4547371	108.4530	130.1875	115.0954

The second test we consider in the numerical experiments is to explore the sparsity of the optimal solution to CQKPL1 with respect to w. The parameters in CQKPL1 are described in the following that are almost the same as in Example 4.1 only after some small modifications for our purpose of numerical tests.

Example 4.2. All the parameters are generated as in Example 4.1 except that the parameters l and w_0 in $\Re^n$ are randomly generated at a uniform distribution with each entry in the interval $[-0.7, 0.3]$ and $[0, 1]$ respectively, and the parameter $w \in \Re^n$ is set to be $wc + w_0$ for given scalar $wc > 0$. We consider $wc = 0.1, 1, 5$ and test three cases of each instance with problem sizes $n = 1000, 10^4, 10^5, 10^6, 10^7$.

In Table 1, we report the performance of Algorithm MSLTA for solving randomly generated CQKPL1 ranging in size from $n = 1000$ to $n = 10^7$, in which the minimum, maximum and average of the number of zeros in the optimal solution *optx* and the running times (in second format) over 20 randomly generated tests for each problem size are listed. To better explore the sparsity of the optimal solution, we also report the average of cardinality of the index set $\{i : l_i \leq 0, u_i \geq 0\}$ over 20 randomly generated tests by nr. The running times of Table 1 are also inclusive of times spent in problem transformation.

The results, depicted in Table 1, show that Algorithm MSLTA in all the instances is able to efficiently find the optimal solution to CQKPL1 with n from one thousand to ten million which at the same time favors sparsity. As can be observed from Table 1, for $n = 10000000$, Algorithm MSLTA averagely takes about 115 seconds to solve problem with $wc = 5$, while it averagely takes about 96 seconds to solve problem with $wc = 0.1$. We also see that the sparsity of the optimal solution to CQKPL1 for each dimension grows as the parameter w increases, especially for each instance with $wc = 5$, each element of optimal solutions whose index in $\{i : l_i \leq 0, u_i \geq 0\}$ is almost forced to be zero. Moreover, for problem size $n \geq 100000$, the average running times for problems with $wc = 5$ are faster than the average running times for problems with $wc = 1$, but slower than the average running times for problems with $wc = 0.1$. This behavior is expected because larger wc would lead to a larger number of iterations for determining the structure of the optimal solution and also lead to cheaper calculations due to more sparsity of the optimal solution (see Remark 3.1).

5 Conclusions

In this paper, we have developed an algorithm to efficiently find the sparse optimal solution of the continuous quadratic knapsack problem with ℓ_1 regularization. Our algorithm that is based on dual reformulation combined with median search strategy has a linear time complexity. Numerical experiments on randomly generated test problems validate the linear time complexity of our algorithm and also demonstrate the effective performance of our algorithm.

Acknowledgements

The first author is deeply indebted to Dr. Zaiwen Wen for his invitation and generous hospitality during his visit to Beijing International Center for Mathematical Research. The authors would like to thank Dr. Yu-Hong Dai at Chinese Academy of Sciences and Dr. Fengmin Xu at Xi'an Jiaotong University for providing us their manuscript [25].

References

[1] D. Bertsimas and R. Shioda, Algorithm for cardinality-constrained quadratic optimization, *Comput. Optim. Appl.* 43 (2009) 1–22.

[2] G. R. Bitran and A. C. Hax, Disaggregation and resource allocation using convex knapsack problems with bounded variables, *Manag. Sci.* 27 (1981) 431–441.

[3] K. M. Bretthauer and B. Shetty, A pegging algorithm for the nonlinear resource allocation problem, *Comput. Oper. Res.* 29 (2001) 505–527.

[4] P. Brucker, An $O(n)$ algorithm for quadratic knapsack problems, *Oper. Res. Lett.* 3 (1984) 163–166.

[5] P. H. Calamai and J. J. Moré, Quasi–Newton updates with bounds, *SIAM J. Numer. Anal.* 24 (1987) 1434–1441.

[6] R. Cominetti, W. F. Mascarenhas and P. J. S. Silva, A Newton ' s method for the continuous quadratic knapsack problem, *Math. Program. Comput.* 6 (2014) 151–169.

[7] S. Cosares and D. S. Hochbaum, Strongly polynomial algorithms for the quadratic transportation problem with a fixed number of sources, *Math. Oper. Res.* 19 (1994) 94–111.

[8] Y.–H. Dai and R. Fletcher, New algorithms for singly linearly constrained quadratic programs subject to lower and upper bounds, *Math. Program.* 106 (2006) 403–421.

[9] T. A. Davis, W. W. Hager and J. T. Hungerford, An efficient hybrid algorithm for the separable convex quadratic knapsack problem, *ACM Trans. Math. Software* 42 (2016), Article 22.

[10] Gurobi Optimization, Inc., *Gurobi Optimizer Reference Manual*, 2016, http://www.gurobi.com.

[11] R. Helgason, J. Kennington and H. Lall, A polynomially bounded algorithm for a singly constrained quadratic program, *Math. Program.* 18 (1980) 338–343.

[12] D. S. Hochbaum and S.-P. Hong, About strongly polynomial time algorithms for quadratic optimization over submodular constraints, *Math. Program.* 69 (1995) 269–309.

[13] J. Jeong, *Indefinite Knapsack Separable Quadratic Programming: Methods and Applications*, PhD dissertation, University of Tennessee, 2014.

[14] K. C. Kiwiel, On linear-time algorithms for the continuous quadratic knapsack problem, *J. Optim. Theory Appl.* 134 (2007) 549–554.

[15] K. C. Kiwiel, Breakpoint searching algorithms for the continuous quadratic knapsack problem, *Math. Program.* 112 (2008) 473–491.

[16] K. C . Kiwiel, Variable fixing algorithms for the continuous quadratic knapsack problem, *J. Optim. Theory Appl.* 136 (2008) 445–458.

[17] N. Maculan and G. G. de Paula, A linear-time median-finding algorithm for projecting a vector on the simplex of $\Re^n$, *Oper. Res. Lett.* 8 (1989) 219–222.

[18] N. Maculan, C. P. Santiago, E. M. Macambira and M. H. C. Jardim, An $O(n)$ algorithm for projecting a vector on the intersection of a hyperplane and a box in $\mathbb{R}^n$, *J. Optim. Theory Appl.* 117 (2003) 553–574.

[19] K. Muller, S. Mika, G. Ratsch, K. Tsuda and B. Scholkopf, An introduction to kernel-based learning algorithms, *IEEE Trans. Neural Netw.* 12 (2001) 181–201.

[20] P.M. Pardalos and N. Kovoor, An algorithm for a singly constrained class of quadratic programs subject to upper and lower bounds, *Math. Program.* 46 (1990) 321–328.

[21] M. Patriksson, A survey on the continuous nonlinear resource allocation problem, *European J. Oper. Res.* 185 (2008) 1–46.

[22] A. G. Robinson, N. Jiang and C.S. Lerme, On the continuous quadratic knapsack problem, *Math. Program.* 55 (1992) 99–108.

[23] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, 1970.

[24] R. T. Rockafellar, *Conjugate Duality and Optimization*, SIAM Publications, Philadelphia, 1974.

[25] F. M. Xu, Y.-H. Dai, Z. H. Zhao and Z.-B. Xu, Efficient projected gradient methods for a class of l_0 constrained optimization problems, preprint.

[26] F. M. Xu, Z. S. Lu and Z.-B. Xu, An efficient optimization approach for a cardinality-constrained index tracking problem, *Optim. Methods Softw.* 31 (2016) 258–271.

[27] Z.-B. Xu, X. Y. Chang, F. M. Xu and H. Zhang, $L_{1/2}$ regularization: a thresholding representation theory and a fast solver, *IEEE Trans. Numer. Netw. Learn. Syst.* 23 (2012) 1013–1027.

Manuscript received 5 November 2016
revised 5 November 2016
accepted for publication 21 January 2017

Yong-Jin Liu
School of Science, Shenyang Aerospace University
Shenyang 110136, People's Republic of China
E–mail address: `yjliu@sau.edu.cn`

Yanan Wen
School of Science, Shenyang Aerospace University
Shenyang 110136, People's Republic of China
E–mail address: `wenyanan4869@163.com`