Linear Time Algorithms for Some Separable Quadratic Programming Problems

Nimrod Megiddo* and Arie Tamir†

November 1991; revised February 1993

Abstract. A large class of separable quadratic programming problems is presented. The problems in the class can be solved in linear time. The class includes the separable convex quadratic transportation problem with a fixed number of sources and separable convex quadratic programming with nonnegativity constraints and a fixed number of linear equality constraints.

1. Introduction

There is a general interest in finding a strongly polynomial algorithm for linear programming. If a general convex quadratic function can be minimized subject to nonnegativity constraints in strongly polynomial time, then obviously the linear programming problem can be solved in strongly polynomial time. Thus, a natural interest arises in quadratic programs with some separable structure.

The (separable) *quadratic transportation problem* is an optimization problem defined as follows. Given $\boldsymbol{a} \in R^m$, $\boldsymbol{b} \in R^n$, $\boldsymbol{C} = (c_{ij}) \in R^{m\times n}$ $(c_{ij} \geq 0)$, and $\boldsymbol{D} = (d_{ij}) \in R^{m\times n}$, find $\boldsymbol{X} = (x_{ij}) \in R^{m\times n}$ so as to

$$
\text{(QTP)} \qquad \begin{aligned}
\text{Minimize} \quad & \tfrac{1}{2}\sum_{i,j} c_{ij}x_{ij}^2 + \sum_{i,j} d_{ij}x_{ij} \\
\text{subject to} \quad & \sum_{j=1}^{n} x_{ij} = a_i \quad (i = 1,\ldots,m) \\
& \sum_{i=1}^{m} x_{ij} = b_j \quad (j = 1,\ldots,n) \\
& x_{ij} \geq 0 \quad (i = 1,\ldots,m\ ,\ j = 1,\ldots,n)\ .
\end{aligned}
$$

*IBM Almaden Research Center, 650 Harry Road, San Jose, California 95120-6099, and School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel. Research supported in part by ONR under contract N00014-91-C-0026.

†School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel.

Cosares and Hochbaum [4] showed that for any fixed value of m, this problem can be solved in strongly polynomial time. Their algorithm runs in $O(n^{m+1})$ arithmetic operations.

Matsui [9] gave a linear time algorithm for the linear transportation problem (*i.e.*, with $\boldsymbol{C} = \boldsymbol{O}$) for any fixed m. But this is really a special case of the d-dimensional linear multiple choice knapsack problem for which linear time algorithms based on the basic multidimensional search of [10] were given by Dyer [6] and Zemel [12]. Tokuyama and Nakano [11] proved that the linear transportation problem can be solved in $O(m^2 n \log^2 n)$ time if $n > m \log m$.

A somewhat simpler problem is that of a separable convex quadratic programming with a fixed number of linear constraints. Best and Tan [1] gave an $O(n^2 \log n)$ algorithm for the following problem:

$$\begin{aligned} \text{Minimize} \quad & \tfrac{1}{2}\sum_{j=1}^{n} c_j x_j^2 + \sum_{j=1}^{n} d_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} a_j x_j = \alpha \\ & \sum_{j=1}^{n} b_j x_j = \beta \\ & \ell_j \le x_j \le h_j \quad (j = 1, \ldots, n) \ . \end{aligned}$$

We demonstrate in this note how the technique of Lagrangian relaxation provides linear time algorithms for such problems based on the multidimensional search procedure of Megiddo [10], and the improvements by Clarkson [3] and Dyer [5]. We do not describe the mulitdimensional search procedure in detail. The interested reader should consult the references. In Sections 2 and 3 we give the idea of the algorithm for the two special cases. A more general treatment is given in Section 4.

2. Separable quadratic programming

For any vector $\boldsymbol{x} \in R^n$, denote

$$\boldsymbol{x}^2 = (x_1^2, \ldots, x_n^2) \ .$$

Consider an optimization problem as follows. Given $\boldsymbol{c} \ge \boldsymbol{0}$, $\boldsymbol{d}$, $\boldsymbol{e}$ and $\boldsymbol{f}$ in R^n, ($e_j \ge -\infty$, $f_j \le \infty$), $\boldsymbol{A} \in R^{m \times n}$, and $\boldsymbol{b} \in R^m$, find $\boldsymbol{x} \in R^n$ so as to

$$\text{(QP)} \qquad \begin{aligned} \text{Minimize} \quad & \tfrac{1}{2}\boldsymbol{c} \cdot \boldsymbol{x}^2 + \boldsymbol{d} \cdot \boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{e} \le \boldsymbol{x} \le \boldsymbol{f} \ , \end{aligned}$$

and think of m as fixed. For any $\boldsymbol{\lambda} \in R^m$, define $\phi(\boldsymbol{\lambda})$ to be the optimal value of the following optimization problem:

$$(P(\boldsymbol{\lambda})) \qquad \begin{array}{rl} \text{Minimize} & \frac{1}{2}\boldsymbol{c}\cdot\boldsymbol{x}^2 + \boldsymbol{d}\cdot\boldsymbol{x} - \boldsymbol{\lambda}\cdot(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}) \\ \text{subject to} & \boldsymbol{e} \le \boldsymbol{x} \le \boldsymbol{f} \ . \end{array}$$

It is well known that $\phi(\boldsymbol{\lambda})$ is concave and, furthermore, maximizing ϕ is equivalent to solving (QP). Note that the evaluation of $\phi(\boldsymbol{\lambda})$ at any given $\boldsymbol{\lambda}$ is quite easy due to the separability:

$$\phi(\boldsymbol{\lambda}) = \phi_1(\boldsymbol{\lambda}) + \cdots + \phi_n(\boldsymbol{\lambda}) + \boldsymbol{\lambda}\cdot\boldsymbol{b} \ ,$$

where

$$\phi_j(\boldsymbol{\lambda}) = \min\left\{\tfrac{1}{2}c_j x_j^2 + \left(d_j - \sum_{i=1}^m \lambda_i a_{ij}\right) x_j \ : \ e_j \le x_j \le f_j\right\} \ .$$

Denote by $x_j^*(\boldsymbol{\lambda})$ a minimizer that yields $\phi_j(\boldsymbol{\lambda})$ if the latter is finite. If $c_j = 0$, we may choose

$$x_j^*(\boldsymbol{\lambda}) = \begin{cases} e_j & \text{if} \ \ \sum_{i=1}^m \lambda_i a_{ij} \le d_j \\ f_j & \text{if} \ \ \sum_{i=1}^m \lambda_i a_{ij} > d_j \end{cases} \ .$$

If $c_j \ne 0$, denote

$$\delta_j(\boldsymbol{\lambda}) = \frac{\sum_{i=1}^m \lambda_i a_{ij} - d_j}{c_j} \quad (j = 1, \ldots, n) \ .$$

Obviously, in this case

$$x_j^*(\boldsymbol{\lambda}) = \begin{cases} e_j & \text{if} \qquad\qquad \delta_j(\boldsymbol{\lambda}) \le e_j \\ \delta_j(\boldsymbol{\lambda}) & \text{if} \quad e_j \le \delta_j(\boldsymbol{\lambda}) \le f_j \\ f_j & \text{if} \quad f_j \le \delta_j(\boldsymbol{\lambda}) \end{cases} \ .$$

It follows that the function $\phi(\boldsymbol{\lambda})$ is piecewise quadratic and concave and its domains of quadraticity are bounded by hyperplanes represented by equations of the form:

$$\sum_{i=1}^m a_{ij}\lambda_i = d_j + c_j e_j \quad \text{and} \quad \sum_{i=1}^m a_{ij}\lambda_i = d_j + c_j f_j \ .$$

Note that once we know the position of a maximizer $\boldsymbol{\lambda}^*$ relative to the two hyperplanes represented by these equations for some value of j, we can replace the function ϕ_j by the resulting quadratic (or linear) function of $\boldsymbol{\lambda}$,

$$\tfrac{1}{2}c_j(x_j^*(\boldsymbol{\lambda}))^2 + \left(d_j - \sum_{i=1}^m \lambda_i a_{ij}\right) x_j^*(\boldsymbol{\lambda}) \ ,$$

where $x_j^*(\boldsymbol{\lambda})$ is the corresponding linear function. Our algorithm runs in phases so that in each phase a fixed proportion of the remaining functions ϕ_j is converted into a quadratic function over a reduced domain in the $\boldsymbol{\lambda}$-space. In order to identify such proportions, we determine the position of $\boldsymbol{\lambda}^*$ relative to a fixed proportion of hyperplanes using the technique of [10]. More details will be given in Section 4.

3. The quadratic transportation problem

To apply the Lagrangian relaxation approach to (QTP), define $\phi(\boldsymbol{\lambda})$ for any $\boldsymbol{\lambda} \in R^m$ to be the optimal value of the following optimization problem:

$$\begin{array}{rl} \text{Minimize} & \frac{1}{2}\sum_{i,j} c_{ij}x_{ij}^2 + \sum_{i,j}(d_{ij}-\lambda_i)x_{ij} + \boldsymbol{\lambda}\cdot\boldsymbol{a} \\ \text{subject to} & \sum_{i=1}^{m} x_{ij} = b_j \quad (j=1,\ldots,n) \\ & x_{ij} \geq 0 \quad (i=1,\ldots,m\ ,\ j=1,\ldots,n)\ . \end{array}$$

We now have a separable problem:

$$\phi(\boldsymbol{\lambda}) = \phi_1(\boldsymbol{\lambda}) + \cdots + \phi_n(\boldsymbol{\lambda}) + \boldsymbol{\lambda}\cdot\boldsymbol{a}\ ,$$

where $\phi_j(\boldsymbol{\lambda})$ is the optimal value of the problem:

$$\begin{array}{rl} \text{Minimize} & \frac{1}{2}\sum_{i=1}^{m} c_{ij}x_{ij}^2 + \sum_{i=1}^{m}(d_{ij}-\lambda_i)x_{ij} \\ \text{subject to} & \sum_{i=1}^{m} x_{ij} = b_j \\ & x_{ij} \geq 0 \quad (i=1,\ldots,m)\ , \end{array}$$

$(j=1,\ldots,n)$. For any fixed $\boldsymbol{\lambda}$, the latter problem is a quadratic knapsack problem (see Brucker [2]) or a resource allocation problem (see Ibaraki and Katoh [8]):

$$\text{(QRS)} \qquad \begin{array}{rl} \text{Minimize} & \frac{1}{2}\sum_{i=1}^{m} c_i y_i^2 + \sum_{i=1}^{m} d_i y_i \\ \text{subject to} & \sum_{i=1}^{m} y_i = b \\ & y_i \geq 0 \quad (i=1,\ldots,m) \end{array}$$

(where we assume, for simplicity of presentation, that $c_i > 0, i = 1,\ldots,m$). This problem is also used by Cosares and Hochbaum [4]. It can be solved as follows. A vector $\boldsymbol{y} \in R^m$, such that $\sum_i y_i = b$, is an optimal solution of (QRS) if and only if there exists a scalar μ such that

$$y_i = \begin{cases} \frac{\mu - d_i}{c_i} & \text{if} \quad d_i < \mu \\ 0 & \text{if} \quad d_i \geq \mu \end{cases} \qquad (i=1,\ldots,m)\ .$$

The value of μ can be found in $O(m)$ time by searching the set of d_i's (using a linear time median-finding algorithm repeatedly) until the set I of indices i such that $d_i < \mu$ is determined. The value of μ is then calculated from the equation (recall that $c_i > 0$)

$$\sum_{i\in I} \frac{\mu - d_i}{c_i} = b\ .$$

Thus,

$$\mu = \frac{b + \sum_{i\in I} d_i/c_i}{\sum_{i\in I} 1/c_i} \ .$$

In the case of $\phi_j(\boldsymbol{\lambda})$,

$$d_i = d_i(\boldsymbol{\lambda}) = d_i(\boldsymbol{\lambda}; j) = d_{ij} - \lambda_i \quad (i = 1, \ldots, m) \ ,$$

$$b = b_j, \qquad c_i = c_{ij}$$

and

$$\mu = \frac{b_j + \sum_{i\in I}(d_{ij} - \lambda_i)/c_i}{\sum_{i\in I} 1/c_i} \ .$$

Consider the cell partition induced on R^m by the hyperplane equations:

$$d_{ij} - \lambda_i = d_{kj} - \lambda_k \quad (1 \le i < k \le m) \ .$$

The order on the d_i's is fixed within every cell. Assume, for the moment, that we have already determined the cell in which the optimal $\boldsymbol{\lambda}^*$ lies. Without loss of generality, we assume that the indices are such that over this cell $d_1(\boldsymbol{\lambda}) \le \cdots \le d_m(\boldsymbol{\lambda})$. Denote

$$S_k = S_k(\boldsymbol{\lambda}) = \sum_{i=1}^{k} \frac{d_k - d_i}{c_i} \quad (k = 1, \ldots, m) \ ,$$

and $S_{m+1} = \infty$. Obviously, $0 = S_1 \le \cdots \le S_m < S_{m+1}$. Consider a finer cell partition obtained by adding also the hyperplane equations: $S_k(\boldsymbol{\lambda}) = b$ $(k = 1, \ldots, m)$. Thus, over a cell in the new partition, the order on $\{S_1, \ldots, S_m, b\}$ is fixed. Suppose we have determined the cell that contains $\boldsymbol{\lambda}^*$. Let ℓ $(1 \le \ell \le m)$ be the index such that

$$S_\ell < b \le S_{\ell+1} \ .$$

Then, there exists a μ, $d_\ell < \mu \le d_{\ell+1}$, such that

$$\sum_{i=1}^{\ell} \frac{\mu - d_i}{c_i} = b \ ,$$

which, in fact, can be calculated directly:

$$\mu = \frac{b + \sum_{i=1}^{\ell} d_i/c_i}{\sum_{i=1}^{\ell} 1/c_i} \ .$$

The essential point to note is that once the order of the $d_i(\boldsymbol{\lambda})$ is known, and also the index ℓ such that $S_\ell < b \le S_{\ell+1}$ is known, then the value of $\phi_j(\boldsymbol{\lambda})$ can be represented as a quadratic function over a certain polyhedron which is known to contain $\boldsymbol{\lambda}^*$. Again, the multidimensional search of [10] can be employed to convert the functions ϕ_j into quadratic functions by identifying the position of $\boldsymbol{\lambda}^*$ relative to the critical hyperplanes. This is done in $O(\log n)$ phases, where in each phase a fixed proportion of the remaining functions is converted. Examine for example the first phase. We start this phase by considering the cell partition induced by the collection of the $O(m^2n)$ hyperplane equations:

$$d_{ij} - \lambda_i = d_{kj} - \lambda_k \quad (1 \le j \le n, 1 \le i < k \le m) \ .$$

Using the multidimensional search in [10], (see Section 4), we find a cell of this partition containing $\boldsymbol{\lambda}^*$ such that the order of the $d_i(\boldsymbol{\lambda}; j)$ $(i = 1, \ldots, m)$ is fixed over this cell for at least half of the indices j. Let $J \subseteq \{1, ..., n\}$ denote the subset of indices for which the order is fixed. Next, for all indices $j \in J$ we define the hyperplane equations: $S_k(\boldsymbol{\lambda}) = b$ $(k = 1, \ldots, m)$. Consider the cell partition induced by this collection of equations. Again, apply [10] to determine for at least one half of the indices in J an index ℓ $(1 \le \ell \le m)$ such that

$$S_\ell < b \le S_{\ell+1} \ .$$

Thus, at least one quarter of the n original functions ϕ_j can be converted into quadratic functions during the first phase. More details are given in Section 4.

4. The general model

We now present a more general class of separable quadratic programming problems which can be solved in linear time. For $j = 1, \ldots, n$, let $\boldsymbol{x}^j$ denote a vector in R^{k_j}. Consider the following quadratic program:

Problem 4.1. Given $\boldsymbol{a} \in R^m$, and for every j $(j = 1, \ldots, n)$ $\boldsymbol{A}^j \in R^{m \times k_j}$, $\boldsymbol{B}^j \in R^{\ell_j \times k_j}$, $\boldsymbol{b}^j \in R^{\ell_j}$, $\boldsymbol{d}^j \in R^{k_j}$, and a symmetric positive semi-definite $\boldsymbol{D}^j \in R^{k_j \times k_j}$, find non-negative vectors $\boldsymbol{x}^j \in R^{k_j}$ $(j = 1, \ldots, n)$, so as to

$$\text{(GQP)} \quad \begin{array}{lccccc} \text{Minimize} & \frac{1}{2}\boldsymbol{x}^1\boldsymbol{D}^1\boldsymbol{x}^1 + \boldsymbol{d}^1 \cdot \boldsymbol{x}^1 & + \cdots + & \frac{1}{2}\boldsymbol{x}^n\boldsymbol{D}^n\boldsymbol{x}^n + \boldsymbol{d}^n \cdot \boldsymbol{x}^n & \\ \text{subject to} & \boldsymbol{A}^1\boldsymbol{x}^1 & + \cdots + & \boldsymbol{A}^n\boldsymbol{x}^n & = \boldsymbol{a} \\ & \boldsymbol{B}^1\boldsymbol{x}^1 & & & = \boldsymbol{b}^1 \\ & & \ddots & & \vdots \\ & & & \boldsymbol{B}^n\boldsymbol{x}^n & = \boldsymbol{b}^n \end{array}$$

Think of the k_j's, the ℓ_j's, and m as fixed. For any $\boldsymbol{\lambda} \in R^m$, define $\phi(\boldsymbol{\lambda})$ to be the optimal value of the following optimization problem:

$$\begin{array}{lccccc} \text{Minimize} & \frac{1}{2}\boldsymbol{x}^1\boldsymbol{D}^1\boldsymbol{x}^1 + (\boldsymbol{d}^1 - \boldsymbol{\lambda}\boldsymbol{A}^1)\boldsymbol{x}^1 & + \cdots + & \frac{1}{2}\boldsymbol{x}^n\boldsymbol{D}^n\boldsymbol{x}^n + (\boldsymbol{d}^n - \boldsymbol{\lambda}\boldsymbol{A}^n)\boldsymbol{x}^n & + \boldsymbol{\lambda} \cdot \boldsymbol{a} \\ \text{subject to} & \boldsymbol{B}^1\boldsymbol{x}^1 & & & = \boldsymbol{b}^1 \\ & & \ddots & & \vdots \\ & & & \boldsymbol{B}^n\boldsymbol{x}^n & = \boldsymbol{b}^n \\ & \boldsymbol{x}^1 & \cdots & \boldsymbol{x}^n & \ge \boldsymbol{0} \end{array}$$

Separability

Due to the separability, $\phi(\boldsymbol{\lambda})$ can be written as

$$\phi(\boldsymbol{\lambda}) = \sum_{j=1}^{n} \phi_j(\boldsymbol{\lambda}) + \boldsymbol{\lambda} \cdot \boldsymbol{a} \ ,$$

where $\phi_j(\boldsymbol{\lambda})$ is the optimal value of the problem

$$(P(\boldsymbol{\lambda})) \qquad \begin{array}{rl} \text{Minimize} & \frac{1}{2}\boldsymbol{x}^j \boldsymbol{D}^j \boldsymbol{x}^j + (\boldsymbol{d}^j - \boldsymbol{\lambda}\boldsymbol{A}^j)\boldsymbol{x}^j \\ \text{subject to} & \boldsymbol{B}^j \boldsymbol{x}^j = \boldsymbol{b}^j \\ & \boldsymbol{x}^j \geq \boldsymbol{0} \ . \end{array}$$

Fix the value of j for a moment. The function $\phi_j(\boldsymbol{\lambda})$ is concave and piecewise quadratic. Its domains of quadraticity are determined by the linear complementarity problem (LCP), associated with the optimization problem defining $\phi_j(\boldsymbol{\lambda})$, which is formulated as follows. A vector $\boldsymbol{x}^j$ is an optimizer of the problem defining $\phi_j(\boldsymbol{\lambda})$ if and only if there exist $\boldsymbol{u} \in R^{\ell_j}$ and $\boldsymbol{v} \in R^{k_j}$ such that

$$(\text{LCP}) \qquad \begin{aligned} (\boldsymbol{B}^j)^T \boldsymbol{u} + \boldsymbol{v} - \boldsymbol{D}^j \boldsymbol{x}^j &= \boldsymbol{d}^j - (\boldsymbol{A}^j)^T \boldsymbol{\lambda} \\ \boldsymbol{B}^j \boldsymbol{x}^j &= \boldsymbol{b}^j \\ \boldsymbol{x}^j, \boldsymbol{v} &\geq \boldsymbol{0} \\ \boldsymbol{v} \cdot \boldsymbol{x}^j &= 0 \ . \end{aligned}$$

We now analyze basic solutions of (LCP). A basic (and complementary) solution $\boldsymbol{z}^{S,U} = (\boldsymbol{x}^j, \boldsymbol{u}, \boldsymbol{v})$ is characterized by two sets: $S \subseteq K = \{1, \ldots, k_j\}$ and $U \subseteq L = \{1, \ldots, l_j\}$, such that (i) for every $i \in S$, $v_i = 0$, (ii) for every $i \notin S$, $x_i^j = 0$, and (iii) for every $i \notin U$, $u_i = 0$. Each coordinate of $\boldsymbol{z}^{S,U}$ is, in fact, a linear function of $\boldsymbol{\lambda}$, so there exist *linear* functions $\xi_{S,U,i}(\boldsymbol{\lambda})$ ($S \subseteq K$, $U \subseteq L$, $i \in S$), and $\eta_{S,U,i}(\boldsymbol{\lambda})$ ($S \subseteq K$, $U \subseteq L$, $i \notin S$), such that in the basic solution $\boldsymbol{z}^{S,U}$,

$$x_i^j(\boldsymbol{\lambda}) = \xi_{S,U,i}(\boldsymbol{\lambda}) \quad \text{and} \quad v_i(\boldsymbol{\lambda}) = \eta_{S,U,i}(\boldsymbol{\lambda}) \ .$$

Hence, the corresponding value of the objective function $\frac{1}{2}\boldsymbol{x}^j \boldsymbol{D}^j \boldsymbol{x}^j + (\boldsymbol{d}^j - \boldsymbol{\lambda}\boldsymbol{A}^j)\boldsymbol{x}^j$ is a quadratic function of $\boldsymbol{\lambda}$ whenever S and U are fixed.

A typical domain of quadraticity of ϕ_j can be described as follows. Fix S and U and consider the linear equations of $\boldsymbol{\lambda}$ corresponding to x_i^j for $i \in S$ and to v_i for $i \notin S$. The cell $C(S,U)$ corresponding to S and U is defined by

$$C(S,U) = \{\boldsymbol{\lambda} \mid \xi_{S,U,i}(\boldsymbol{\lambda}) \geq 0 \ (i \in S), \ \eta_{S,U,i}(\boldsymbol{\lambda}) \geq 0 \ (i \notin S)\} \ .$$

The hyperplanes that induce the partition of the $\boldsymbol{\lambda}$-space into domains of quadraticity of ϕ_j can be characterized as $\{\boldsymbol{\lambda} \mid \xi_{S,U,i}(\boldsymbol{\lambda}) = 0\}$ for $i \in S$ and $\{\boldsymbol{\lambda} \mid \eta_{S,U,i}(\boldsymbol{\lambda}) = 0\}$ for $i \notin S$.

Note, however, that for any pair (S,i) such that $i \in S$, the equations $\xi_{S,U,i}(\boldsymbol{\lambda}) = 0$ and $\eta_{S\setminus\{i\},U,i}(\boldsymbol{\lambda}) = 0$ are identical and both are induced by $v_i = x_i^j = 0$.

It is important to note that for each j, the total number of hyperplanes is fixed. Moreover, the number of cells is bounded by a fixed constant, since both k_j and ℓ_j are fixed. Thus, for each variable it takes only constant time to construct all the cells (*i.e.*, domains of quadraticity) of $\phi_j(\boldsymbol{\lambda})$, and compute for each cell a solution vector $\boldsymbol{x}^j(\boldsymbol{\lambda})$ whose corresponding objective function value is $\phi_j(\boldsymbol{\lambda})$, where $\boldsymbol{x}^j(\boldsymbol{\lambda})$ is linear over this cell. Note that it may be impossible to choose $\boldsymbol{x}^j(\boldsymbol{\lambda})$ as a continuous function over the whole $\boldsymbol{\lambda}$-space, but this is not really necessary. Let p_j denote the number of hyperplane equations that determine the cells of the finer partition corresponding to $\phi_j(\boldsymbol{\lambda})$. In the model of Section 2, $k_j = 1$ ($\boldsymbol{x}^j \in R$), so $p_j = 2$. In the quadratic transportation model of Section 3, $k_j = m$ ($\boldsymbol{x}^j \in R^m$), and $p_j = O(m^2)$.

Maximizing $\phi(\boldsymbol{\lambda})$

Each function ϕ_j ($j = 1, \ldots, n$) has p_j hyperplane equations. For each j, we compute these p_j functions, all the respective cells, and the linear representation of a solution $\boldsymbol{x}^j(\boldsymbol{\lambda})$ for each cell. Altogether, we have at most $\sum_{j=1}^n p_j = O(n)$ equations and $\sum_{j=1}^n O(p_j^m) = O(n)$ cells of quadraticity of the functions ϕ_j. We note in passing that the cells of the function ϕ can be found by computing intersections of the components ϕ_j; the number of such intersections is $O(n^m)$ and they can all be computed in $O(n^m)$ time (since the total number of hyperplanes is $O(n)$; see chapter 7 in [7]), so ϕ can be maximized in strongly polynomial time whenever m is fixed. Let $\boldsymbol{\lambda}^*$ be a maximizer of $\phi(\boldsymbol{\lambda})$. If a cell of ϕ_j containing $\boldsymbol{\lambda}^*$ is known, then ϕ_j can be replaced by its respective quadratic expression. Furthermore, if such a cell is determined for r values of j, then we can replace r functions ϕ_j by a *single* quadratic function of $\boldsymbol{\lambda}$.

The algorithm works in phases as follows. At the start of Phase s ($s = 1, 2, \ldots$), the function $\phi(\boldsymbol{\lambda})$ is represented as the sum of r_s functions ϕ_j and a single concave quadratic $q(\boldsymbol{\lambda}) = \boldsymbol{\lambda} \cdot \boldsymbol{Q}\boldsymbol{\lambda} + \boldsymbol{a} \cdot \boldsymbol{\lambda}$. During a phase we identify, for each of at least $r_s/2$ functions ϕ_j, a cell of ϕ_j that contains $\boldsymbol{\lambda}^*$. In this way we "discard" $r_s/2$ functions in the sense that we start the next phase with $r_{s+1} \leq r_s/2$ functions and the discarded functions are simply replaced by quadratic functions which are accumulated into the quadratic term $q(\boldsymbol{\lambda})$.

To identify a cell of ϕ_j containing $\boldsymbol{\lambda}^*$, we determine the position of $\boldsymbol{\lambda}^*$ with respect to all of its p_j hyperplane equations. We apply the multidimensional search of [10] (or the improvements suggested in [3] and [5]) for identifying the cells.

Suppose there exists an oracle which accepts as input any hyperplane equation in R^m and outputs the position of $\boldsymbol{\lambda}^*$ with respect to this equation. Consider any set of k hyperplane equations in R^m. ¿From the multidimensional search it follows that there are constants α, $0 < \alpha < 1$, and β, which depend on m but not on k, such that by calling upon the oracle β times, we can identify the position of $\boldsymbol{\lambda}^*$ with respect to at least αk of the given hyperplane equations. In addition to the time spent by the oracle, the multidimensional search takes $O(k)$ effort. Using the multidimensional search repeatedly on the remaining equations, we conclude that for any constant γ, independent of k $(0 < \gamma < 1)$, $\beta \log(1-\gamma)/\log(1-\alpha)$ calls to the oracle plus $O(k)$ additional time, suffice to identify the position of $\boldsymbol{\lambda}^*$ with respect to at least some γk of the given set of k hyperplanes.

The reader is referred to [10; 3; 5] for a detailed description of the multidimensional search. We note in passing that the approach is based on reducing the m-dimensional problem into a number of $(m-1)$-dimensional problems. This number depends on m but not on k. Ultimately, a one-dimensional case is solved with the linear-time median-finding algorithm as the main tool. In this case each of the k hyperplane equations defines a point on the real line. Let $\boldsymbol{\lambda}_0$ be the median of these k points. We determine the position of $\boldsymbol{\lambda}^*$ with respect to $\boldsymbol{\lambda}_0$ by computing the one-sided derivatives of $\phi(\boldsymbol{\lambda})$ at $\boldsymbol{\lambda}_0$. This can clearly be done in linear time. Since $\boldsymbol{\lambda}_0$ is the median, we now know the position of $\boldsymbol{\lambda}^*$ with respect to at least half of the given k points.

We now show how to use the above to identify the cells containing $\boldsymbol{\lambda}^*$ with respect to at least a half of the functions ϕ_j. Let $\{\phi_j\}$, $j = 1, \ldots, r_s$, be the set of piecewise quadratic functions given at the beginning of Phase s. Assume, without loss of generality, that $p_1 \geq p_2 \geq \cdots \geq p_{r_s} \geq 1$. Set $k = \sum_j p_j$ and $\gamma = 1 - 1/(2p_1)$. Also, define $\bar{p}$ to be the mean of the p_j's. Using the above approach, we identify the position of $\boldsymbol{\lambda}^*$ with respect to γk of the k hyperplanes. (Recall that $k = O(r_s)$.) We claim that, having done that, for each of at least $r_s/2$ out of the r_s ϕ_j's, the position of $\boldsymbol{\lambda}^*$ with respect to all the corresponding p_j hyperplanes has already been computed. In other words, the cell of ϕ_j that contains $\boldsymbol{\lambda}^*$ can now be identified. For, if this was not true, then the maximum number of hyperplanes with respect to which the position of $\boldsymbol{\lambda}^*$ has been identified, would be less than

$$r_s/2 + \sum_{j=1}^{r_s}(p_j - 1) = \sum_{j=1}^{r_s} p_j - r_s/2 .$$

However, $\gamma k \geq (1 - 1/(2\bar{p})) \sum_j p_j = \sum_j p_j - r_s/2$.

To summarize, it takes a constant number of calls to the oracle plus $O(r_s)$ time to discard at least half of the r_s functions $\phi_j(\boldsymbol{\lambda})$, which are given at the beginning of Phase s. Since the total number of functions at Phase 1 is $O(n)$, the total number of phases is $O(\log n)$. We will show, however, that the total effort of maximizing the objective function $\phi(\boldsymbol{\lambda})$ is only $O(n)$. The proof goes by induction on the dimension m. (Note that we view the objective function ϕ as a sum of $O(n)$ concave piecewise quadratics defined over R^m, where the number of hyperplanes associated with each term j is some constant p_j.) For the case $m = 1$, we use the usual median-finding scheme as in Zemel [12] to maximize ϕ in $O(n)$ time.

Turning to a general m, it follows from our solution approach, that the $O(n)$ bound is implied if the oracle can find the position of $\boldsymbol{\lambda}^*$ with respect to a single hyperplane during Phase s in $O(r_s)$ time. (Such an oracle ensures that the total effort spent during Phase s is $O(r_s)$, and since $r_{s+1} \leq r_s/2$, the overall bound is $O(n)$.) Consider a hyperplane H in R^m presented to the oracle during Phase s, *i.e.*, we need to find the position of $\boldsymbol{\lambda}^*$ with respect to H. We argue that this task can be accomplished by solving three maximization problems of our generic type over R^{m-1}. Assume that the LCP and the heperplane H are defined by rational data, and let I denote their input length. Then $\boldsymbol{\lambda}^*$ is the minimum of a quadratic function with rational coefficients. The input length of these coefficients can easily be bounded above by a quadratic function of I. If $\boldsymbol{\lambda}^*$ is not on H, then a rational lower bound, say ϵ, on the distance between $\boldsymbol{\lambda}^*$ and H can be predetrmined in terms of the input length I.[1] Let H_- and H_+ be two hyperplanes parallel to H, lying on different sides of H at a distance of ϵ. Due to concavity, by maximizing the objective ϕ over H, H_-, and H_+, (i.e., solving 3 $(m-1)$-dimensional maximization problems), we can clearly conclude the position of $\boldsymbol{\lambda}^*$ with respect to H. By the induction hypothesis, we conclude that the effort involved is $O(r_s)$.

References

[1] M. W. Best and R. Y. Tan, "An $O(n^2 \log n)$ strongly polynomial algorithm for quadratic program with two equations and lower & upper bounds," Research Report 90-04, Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada, 1990.

[2] P. Brucker, "An $O(n)$ algorithm for quadratic knapsack problems," *Operations Research Letters* **3** (1984) 163–166.

[3] K. L. Clarkson, "Linear programming in $O(n3^{d^2})$ time," *Information Processing Letters* **22** (1986) 21–24.

[4] S. Cosares and D. S. Hochbaum, "Strongly polynomial algorithms for the quadratic transportation problem with a fixed number of sources," manuscript, University of California, Berkeley, October 1990.

[5] M. E. Dyer, "On a multidimensional search technique and its application to the Euclidean one-center problem," *SIAM J. Comput.* **15** (1986) 725–738.

[6] M. E. Dyer, "An $O(n)$ algorithm for the multiple-choice knapsack linear program," *Math. Programming* **29** (1984) 57–63.

[7] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer-Verlag, Berlin, Heidelberg, New York, 1987.

[1] We note that instead of parallel hyperplanes we can also develop a method based on subgradients which works over the real numbers as well.

[8] T. Ibaraki and N. Katoh, *Resource allocation problems: Algorithmic approaches*, The M. I. T. Press, Cambridge, Mass. 1988.

[9] T. Matsui, "Linear time algorithm for the Hitchcock transportation with fixed number of supply points," unpublished.

[10] N. Megiddo, "Linear programming in linear time when the dimension is fixed," *J. ACM* **31** (1984) 114–127.

[11] T. Tokuyama and J. Nakano, "Efficient algorithms for the Hitchcock transportation problem," IBM Research Report RT 0060, Tokyo Research Laboratory, Tokyo, Japan, July 1991.

[12] E. Zemel, "An $O(n)$ algorithm for the linear multiple choice knapsack problem and related problems ," *Information Processing Letters* **18** (1984) 123–128.