Solving the Convex Ordered Set Problem with Applications to Isotone Regression

by

Ravindra K. Ahuja
James B. Orlin

SWP# 3988 February 1998

Solving the Convex Ordered Set Problem with Applications to Isotone Regression

Ravindra K. Ahuja*
Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
email: ahuja@mit.edu

James B. Orlin
Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
email: jorlin@mit.edu

(Revised January 7, 1998)

* On leave from Indian Institute of Technology, Kanpur 208 016, INDIA.

Solving the Convex Ordered Set Problem with Applications to Isotone Regression

Ravindra K. Ahuja[1] and James B. Orlin[2]

ABSTRACT

The *convex ordered set problem* is to minimize $\sum_{j=1}^{n} C_j(x_j)$ subject to $l \leq x_1 \leq x_2 \leq x_3 \leq \ldots \leq x_n \leq u$, where $C_j(x_j)$ is a strictly convex function of x_j for each j = 1, 2, ..., n, and l and u are specified lower and upper bounds on the x_j's. The convex ordered set problem is a generalization of the isotonic regression problems with complete order, an important class of problems in regression analysis. We describe additional applications of the convex ordered set problem to different scenarios including inverse optimization and an important subproblem in Dial-A-Ride Transit. In this paper, we first derive optimality conditions for the convex ordered set problem and then use these conditions to develop a generic algorithm that solves the convex ordered set problem as a sequence of at most 2n-1 single variable convex minimization problems. This algorithm determines an integer solution of the convex ordered set problem in $O(n^2 \log U)$ time, where $U = \max\{|l|, |u|\}$. We next use a scaling technique in our generic algorithm and improve its running time to $O(n \log U)$. When our algorithms are applied to isotonic regression problems with different L_p norms, we get: (i) an O(n) algorithm for the quadratic cost case (L_2 norm); (ii) an O(n log n) algorithm for the rectilinear cost case (L_1 norm); and (iii) an O(n) algorithm for the unit weight minimax cost case (L_∞ norm), and an $O(\min\{n^2, n \log U\})$ algorithm for the weighted minimax cost case. These time bounds either match the best available time bounds to solve these problems or improve them.

[1] Sloan School of Management, MIT, Cambridge, MA 02139, USA; On leave from Indian Institute of Technology, Kanpur 208 016, INDIA.

[2] Sloan School of Management, MIT, Cambridge, MA 02139, USA.

1. INTRODUCTION

In this section, we study the following problem, which we call the *convex ordered set problem:*

Minimize $\sum_{j=1}^{n} C_j(x_j)$ (1a)

subject to

$l \le x_1 \le x_2 \le x_3 \le \ldots \le x_n \le u,$ (1b)

where $C_j(x_j)$ is a strictly convex function of x_j for each j = 1, 2, …, n, and l and u are specified lower and upper bounds on the x_j's. Let $U = \max\{|l|, |u|\}$. The convex ordered set problem finds applications in different scenarios including isotonic regression, inverse optimization and an important subproblem in Dial-A-Ride Transit.

The *isotonic regression problem* is defined as follows. Given the vector $a = \{a_1, a_2, \ldots, a_n\} \in R^n$ and an integer number p, find $x = \{x_1, x_2, \ldots, x_n\}$, so as to minimize

$$||x - a||_p = \sqrt[p]{\sum_{j=1}^{n}(x_j - a_j)^p} \quad (2)$$

and subject to the *isotonicity* (or *monotonicity*) constraints $x_1 \le x_2 \le \ldots. \le x_n$. Since minimizing $\sqrt[p]{\sum_{j=1}^{n}(x_j - a_j)^p}$ is equivalent to minimizing, we shall henceforth assume that $||x - a||_p = \sum_{j=1}^{n}(x_j - a_j)^p$. The isotonic regression problem is an important problem in regression analysis due to its applications. The isotonic regression problem arises in statistics, production planning, and inventory control and has been studied extensively in the literature. The books by Barlow et al. [1972] and Robertson et al. [1988] describe several applications of the isotone regression problem. These books and the references given below describe several algorithms for solving the isotone regression problem. The convex ordered set problem considered by us includes the isotonic regression problem as a special case since $(x_j - a_j)^p$ is a strictly convex function for every positive integer p.

In this paper, we prove several results concerning optimal solutions of the convex ordered set problem and use these results to develop efficient. We first present a generic algorithm that solves the convex ordered set problem as a sequence of at most 2n-1 single variable convex minimization problems. This algorithm has similarities to the well known PAV (Pool Adjacent Violater) algorithms for the isotonic regression problem (see Best and Chakravarti [1990] and Stromberg [1991]) and it determines an optimal integer solution of the convex ordered set problem in $O(n^2 \log U)$ time. Our major contribution in this paper is a scaling version of the generic algorithm which improves its running time to $O(n \log U)$.

Since the isotonic regression problem is a special case of the convex ordered set problem, our algorithms apply to the isotonic regression problem. In the table shown in Figure 1, we compare the running times of our algorithms versus the running times of existing algorithms for the isotonic regression problem for different cost structures due to other researchers. In the table, for the sake of brevity we give only the selected references for algorithms on isotonic regression problems. Some algorithms in the table obtain integer-valued optimal solutions while others obtain real-valued optimal solutions. We show in Section 4 that given a real-valued optimal solution of the convex ordered set problem, we can convert it to an integer optimal solution in O(n) time by using a simple rounding-off scheme. For all cost structures, our time bounds either match the best available time bounds or improve them.

Objective Function (Minimization Version)	**Solutions (Real/Int)**	**Studied by**	**Best available time bound**	**Our time bound**
$\sum_{j=1}^{n} w_j \lvert x_j - a_j \rvert$	Real	Robertson and Wright [1980], Menendez and Salvador [1987], Chakravarti [1989]	$O(n^2)$	$O(n \log n)$
$\sum_{j=1}^{n} w_j (x_j - a_j)^2$	Real	Best and Chakravarti [1990]	$O(n)$	$O(n)$
$\max\{\lvert x_j - a_j \rvert : 1 \le j \le n\}$	Real	Ubhaya [1974a, 1974b], Liu and Ubhaya [1997]	$O(n)$	$O(n)$
$\max\{w_j \lvert x_j - a_j \rvert : 1 \le j \le n\}$	Real	Ubhaya [1974a, 1974b], Liu and Ubhaya [1997]	$O(n^2)$	$O(n^2)$
$\max\{w_j \lvert x_j - a_j \rvert : 1 \le j \le n\}$	Integer	Liu and Ubhaya [1997]	$O(n^2)$	$O(\min\{n^2, n \log U\})$
$\sum_{j=1}^{n} C_j (x_j - a_j)$, where each C_j is a convex function	Integer	Stromberg [1991]	$O(n^2 \log U)$	$O(n \log U)$

Figure 1. Comparison of the running times of our algorithms with the best available time bounds for solving the isotonic regression problem.

2. APPLICATIONS

Perhaps the most well known application of the convex ordered set problem is in isotonic regression and for these applications we refer the reader to the books of Barlow et al. [1972] and Robertson et al. [1988]. We describe two additional applications of the convex ordered set problem.

2.1 DIAL-A-RIDE TRANSIT

Dial-A-Ride Transit (DART) is a shared taxicab system that typically serves areas of low travel demand and/or a population with special needs. Customers call a dial-a-ride agency sufficiently in advance (say, one day before) requesting to be carried from specific origins to specific destinations during specified times. The agency dispatches a vehicle to meet several such demands and customers are pooled to reduce the operational costs. A vehicle schedule typically consists of picking up and dropping off of some customers in a specific sequence, and at any point of time several customers can be on-board the vehicle. Due to the customer pooling, the transit times for customers are larger than the direct transit times, and the customers' pickup and delivery times often may not be met. Consequently, customers specify a time window for the pickup time and a time window for the delivery time, and a customer is picked up and delivered in its specified time window. Thus, dial-a-ride transit problems are a subclass of vehicle routing problems with time windows.

Dial-a-ride transit problems are extensively studied in the literature. We refer the reader to the papers by Kontoravdis and Bard [1994], Desrosiers et al. [1995], and Ahuja and Orlin [1996]. Researchers have developed exact as well as heuristic algorithms for dial-a-ride transit problems. Since exact algorithms can solve only small sized problems, heuristic algorithms have been more extensively studied. A heuristic algorithm typically performs two functions: routing and scheduling. The routing part determines the route of each vehicle - the order in which specific customers assigned to a vehicle will be picked up and delivered. The scheduling part assigns a time schedule to the route - the times at which the customers will be picked up and delivered. We will show that determining the optimal schedule for a given route can be formulated as the convex ordered set problem.

Consider the following scheduling problem: We are given a sequence of stops (on an increasing time scale) 1-2-3- .... n, where each stop denotes a pickup or a delivery point. We assume that the vehicle takes t_j time to go from stop j to stop j+1. Each stop j has a desired pickup or delivery time a_j and deviation from the desired time is penalized. If the vehicle visits stop j at time x_j, then the penalty is given by $C_j(x_j - a_j)$, where $C_j(x_j - a_j)$ is a convex function of x_j. Observe that the time window constraints can be incorporated in the definition of the convex function by making the penalty too high for the pickup/delivery outside the window limits. We assume that the vehicle is allowed to wait with customers on board the vehicle. Allowing the vehicle to wait idly between the stops permits greater flexibility and generally yields a lower cost solution. This vehicle scheduling problem can be formulated as follows:

$$\text{Minimize } \sum_{j=1}^{n} C_j(x_j - a_j) \tag{3a}$$

subject to

$$x_j + t_j \leq x_{j+1} \quad \text{for all } j = 1, 2, \ldots, (n-1). \tag{3b}$$

This formulation is somewhat different than the formulation of the convex ordered set problem given in (1), but a transformation of variables will make them equivalent. Let $y_1 = x_1$, and $y_j = x_j - \sum_{k=1}^{j-1} t_k$. Substituting y_j's in (3) gives us the following formulation:

$$\text{Minimize } \sum_{j=1}^{n} C_j(y_j + \sum_{k=1}^{j-1} t_k) \tag{4a}$$

subject to

$$y_j \leq y_{j+1} \quad \text{for all } j = 1, 2, \ldots, (n-1), \tag{4b}$$

which is a special case of the convex ordered set problem (after suitable upper and lower bounds have been imposed on y_j's).

2.2 Inverse Optimization

Inverse problems have been studied originally by researchers working with geophysical data. In his book, Tarantola [1986] defines inverse problems in the following manner: "*Given a certain amount of (a priori) information on some model parameters, and given an uncertain physical law relating some observable parameters to the model parameters, in which sense should we modify the a priori information, given the uncertain results of some experiments*". Recently, there has been a flurry of activities on inverse problems corresponding to well known optimization problems (see, for example, Ahuja and Orlin [1997a, 1997b, 1997c] for some recent results and additional references). We can define an inverse optimization problem as follows. Let $\mathcal{X}$ denote the set of feasible solutions of an optimization problem. Given a solution $x^* \in \mathcal{X}$ and an a priori estimated cost vector c, the inverse optimization problem is to identify another cost vector d so that $dx^* \leq dx$ for all $x \in \mathcal{X}$ and such that the deviation of d from c is minimum. Roughly speaking, the inverse optimization problem is to identify a cost vector d which is nearest to a specified cost vector c and with respect to which the given solution x^* is an optimal solution of the optimization problem. In this sense of inverse optimization, the inverse sorting problem can be stated as: Given n numbers $a_1, a_2, \ldots, a_n$, perturb these numbers to $x_1, x_2, \ldots, x_n$, respectively, so that $x_1 \leq x_2 \leq x_3 \leq \ldots \leq x_n$ and the cost of perturbation given by $\sum_{j=1}^{n} C_j(x_j - a_j)$ is minimum, where each $C_j(x_j - a_j)$ is a convex function of x_j. Clearly, the inverse sorting problem is a special case of the convex ordered set problem. We give below two applications of the inverse sorting problem.

Job Shop Scheduling

Consider a job shop scheduling problem in which one wants to minimize the total flow time (or, the weighted flow time). This problem can be solved by processing the jobs in the ascending order of the shortest processing time (or, the weighted shortest processing times). Suppose that the processing times are not known precisely and S denotes the optimal schedule with respect to the estimated processing times. Suppose that the processing times of jobs in this schedule are revealed to be $a_1, a_2, \ldots, a_n$. The solution of the convex ordered set problem with $\sum_{j=1}^{n} C_j(x_j - a_j)$ as the objective function is a measure of the inaccuracy of the schedule S. This gives the cost of perturbing the a_j's so that S is an optimal schedule with respect to the perturbed processing times. As such, it is one metric for the deviation from optimality.

Chess Ranking

Players in chess (and other sports activities) are ranked through points. However, the point rankings may not reflect the true playing ability. So one can propose an alternate ranking R. The inverse optimization problem on R would be: What is the minimum amount that one may perturb the point rankings so that the points are consistent with R? This may be of interest if one proposes alternative rankings.

3. PRELIMINARIES

In this section, we describe the notation, state the assumptions, as well as develop some results used later in this paper.

Assumptions

We consider the convex ordered set problem subject to the following two assumptions:

Assumption 1. Each function $C_j(x_j)$ is a strictly convex function of x_j.

Assumption 2. Each function $C_j(x_j)$ can be evaluated in O(1) time for a given value of x_j.

Assumption 1 implies that the function $C_j(x_j)$ has a unique optimal solution value. This result simplifies our subsequent presentation. This assumption is made without any loss of generality since we may add $\varepsilon(x_j - L)^2$ to $C_j(x_j)$ for sufficiently small ε. (This can be implemented using lexicography.) With this perturbation, there is also a unique minimum integral solution. Assumption 2 allows us to analyze the worst-case complexity of the algorithms developed in this paper since they all involve evaluating the cost functions.

Minimizing Single-Variable Convex Functions

The first generic algorithm proposed in this paper proceeds by finding the minima of a single-variable convex function $F(\theta)$ that varies in the range $[l, u]$. There are several well known search methods, including binary search and Fibonacci search, that maintain a search interval containing the optimal solution and perform one or two function evaluations to reduce the search interval by a constant factor (see, for example, Bazaraa, Sherali and Shetty [1992]). These search methods terminate when the length of the search interval decreases below some acceptable limit ε. The number of iterations performed by these search methods is $O(\log(U/\varepsilon))$. Each iteration of these search methods performs O(1) function evaluations; hence, the running time of these search methods is $O(\log(U\text{-}L)/\varepsilon)$ evaluations of the function $F(\theta)$. In case we want to find an integer optimal solution of the function $F(\theta)$, then we can terminate the search method whenever $\varepsilon < 1$. In this case the running time of the method will be the time taken by O(log U) function evaluations.

Subproblems

The algorithms described in this paper solve the convex ordered set problem, with variables $x_1 \le x_2 \le \ldots \le x_n$, by repeatedly solving it on a subset of given variables, which we refer to as a *subproblem*. We define the subproblem P[p, q] as a convex ordered set problem on the variables $x_p, x_{p+1}, \ldots, x_q$ only. The subproblem P[p, q] can be stated as:

$$\text{Minimize } \sum_{j=p}^{q} C_j(x_j) \tag{5a}$$

subject to

$$l \le x_p \le x_{p+1} \le \ldots \le x_q \le u. \tag{5b}$$

We refer to the constraints (5b) as the *feasibility constraints*. We denote a solution, not necessarily feasible, of the subproblem P[p, q] as x[p, q]. Thus, x[p, q] = $\{x_p, x_{p+1}, \ldots, x_q\}$. We denote the optimal objective function value of P[p, q] as z*[p, q]. We say that a feasible solution x[p, q] is a *single-valued solution* of P[p, q] if $x_p = x_{p+1} = \ldots = x_q$. We also denote the solution x for (5) when restricted to the subproblem P[p, q] by x[p, q]. In other words, if $x = \{x_1, x_2, \ldots, x_n\}$, then $x[p, q] = \{x_p, x_{p+1}, \ldots, x_q\}$. In view of our notation, P[1, n] refers to the original convex ordered set problem stated in (1). We shall denote P[1, n] as $\mathcal{P}$ and denote its optimal solution z*[1, n] as z*.

An ordered subset of integers $i_1, i_2, \ldots, i_k$ is said to be *consecutive* if $i_j = i_{j-1} + 1$ for each j = 2, 3, ..., k. We abbreviate the consecutive subset {i, i+1, ... , j} by [i, j]. Our algorithms described in this paper maintain consecutive subsets of {1, 2, 3, ... , n}. We shall usually refer to them more briefly as subsets. We call two (consecutive) subsets [i, j] and [k, l] *adjacent* if k = j + 1. If two subsets [i, j] and [k, l] are adjacent, then the two subproblems P[i, j] and P[k, l] are also said to be adjacent. We represent a *family* ***F*** *of subsets* (or, simply a family) by a partition of {1, 2, ... , n} into adjacent subsets. Each family **F** defines a collection of subproblems. For example, if **F** = [[1, 4], [5, 5], [6, 8]], then the associated collection of subproblems is P[1, 4], P[5, 5], and P[6, 8]. Our algorithms described in this paper proceed by maintaining a family of subsets in every iteration. We shall use the following result later in the paper.

Lemma 1. *If **F** is a family, then* $\sum_{(p,q)\in \mathbf{F}} z^*[p, q] \le z^*$.

Proof. Let x* be an optimal solution of the problem $\mathcal{P}$. Then, for each subset [p, q] ∈ **F**, x*[p, q] is a feasible solution of the subproblem P[p, q]. Since the optimal solution of the subproblem P[p, q] must be at least as good as x*[p, q], we get $z^*[p, q] \le \sum_{j=p}^{q} C_j(x_j^*)$. Summing these inequalities for all subproblems in **F**, we get $\sum_{(p,q)\in \mathbf{F}} z^*[p, q] \le \sum_{(p,q)\in \mathbf{F}} \sum_{j=p}^{q} C_j(x_j^*) = \sum_{j=1}^{n} C_j(x_j^*) = z^*$. ■

Single-Valued Solutions

Consider the subproblem P[p, q]. Suppose that we restrict attention to those solutions of P[p, q] where each variable in the subproblem has the same value, that is, $x_p = x_{p+1} = \ldots = x_q = \theta$ for some θ. This gives us the following objective function for the subproblem:

$$F(p, q, \theta) = \sum_{j=p}^{q} C_j(\theta). \tag{6}$$

Since each function $C_j(\theta)$ is a strictly convex function of θ, it follows that for fixed values of p and q, F(p, q, θ) is also a strictly convex function and therefore possesses a unique optimal solution. With our perturbation it also has a unique integer solution. We denote by θ_{pq} the value of $\theta \in [l, u]$ for which F(p, q, θ) attains its minimum. If we require all variables for the subproblem P[p, q] to have the same value, then $x_p = x_{p+1} = \ldots = x_q = \theta_{pq}$ defines its unique optimal single-valued solution. If this solution also happens to be an optimal solution for the subproblem (4), we say that the subproblem P[p, q] has a *single-valued optimal solution.* A subproblem P[p, q] may or may not have a single-valued optimal solution. An important issue for our algorithmic approach is to characterize subproblems with single-valued optimal solutions. We give in Theorem 1 the necessary and sufficient conditions for subproblems with single-valued optimal solutions; the proof of the theorem is given in Appendix.

Theorem 1. *A subproblem P[p, q] has a single-valued optimal solution if and only if the following sets of conditions are satisfied:*

(i) $\theta_{pj} \geq \theta_{pq}$ *for all* $j = p, p+1, \ldots, q$; *and* *(7a)*

(ii) $\theta_{jq} \leq \theta_{pq}$ *for all* $j = p, p+1, \ldots, q$. *(7b)*

4. PROPERTIES OF OPTIMAL SOLUTIONS

In this section, we characterize the optimal solutions of the convex ordered set problem. We will use this characterization to prove the correctness of the convex ordered set algorithm described in the next section. We also prove some additional results, which allow us to develop highly efficient algorithms for solving the convex ordered set problem. Special cases of some results in this section appear in the literature devoted to the isotonic regression problem and use duality theory of linear or nonlinear programming. Our proofs are stand-alone proofs and do not require duality theory.

Optimality Conditions

Let the solution x be an ordered sequence of n numbers that is not necessarily feasible for (1). We define the *family* ***F****(x)*, associated with the solution x, as the collection of (consecutive) subsets satisfying the following property: two consecutive indices (that is, j and j+1) are in the same subset if and only if the two variables corresponding to these indices have the same value in x (that is, $x_j = x_{j+1}$). For example, if x = {2, 3, 3, 1, 1, 2, 7}, then **F**(x) = {[1,1], [2, 3], [4, 5], [6, 6], [7, 7]}. We call a (possibly infeasible) solution x a *good solution* if and only if x[p, q] is an optimal solution of the subproblem P[p, q] for every [p, q] $\in$ **F**(x). Observe that by definition each value in the solution x[p, q] is the same. Since x[p, q] is an optimal solution of P[p, q], each value in it must equal θ_{pq}.

The following theorem gives a characterization of the optimal solutions of the convex ordered set problem, which we refer to as the *optimality conditions*.

Theorem 2. *A solution x* is an optimal solution of the convex ordered set problem if and only if x* is both feasible and good.*

Proof. We will first prove the necessity of the optimality conditions. Let x* be an optimal solution of the convex ordered set problem. Clearly, x* must be feasible. We next show that x* must be good. Define the family **F**(x*) of subsets. Suppose that for some subset [p, q] $\in$ **F**(x*), x*[p, q] is not an optimal solution of P[p, q]; instead x′[p, q] is an optimal solution of P[p, q] with lower objective function value. Now consider the solution y defined as follows: $y_j = x_j^*$ for $j < p$ or $j > q$ and $y_j = (1-\varepsilon)\ x_j^* + \varepsilon\ x'_j$ for $p \le j \le q$, where ε is chosen to be sufficiently small so that $y_{p-1} \le y_p$ and $y_q \le y_{q+1}$. Then y is a feasible solution of the convex ordered set problem with cost lower than that of x*, which contradicts the optimality of the solution x*. Therefore, x*[p, q] is an optimal solution of every subproblem P[p, q] with [p, q] $\in$ **F**(x*).

We now show the sufficiency of the optimality conditions by proving that if the solution x* is feasible and x*[p, q] is an optimal solution of P[p, q] for every [p, q] $\in$ **F**(x*), then x* must be an optimal solution of the convex ordered set problem. Lemma 1 implies that $\sum_{[p,q]\in \mathbf{F}(x^*)} z^*[p,\ q] \le z^*$. The fact that x* is a feasible solution of (1) implies that $\sum_{[p,q]\in \mathbf{F}(x^*)} z^*[p,\ q] \ge z^*$. Combining the preceding two inequalities, we get $\sum_{[p,q]\in \mathbf{F}(x^*)} z^*[p,\ q] = z^*$, establishing that x* is an optimal solution of the convex ordered set problem. ■

Just as we defined good solutions, we can define good families. We call a family **F** *good* if for each subset [p, q] $\in$ **F** the subproblem P[p, q] has a single-valued optimal solution. Notice that for each good solution x, we can associate a good family **F**(x) in the manner defined earlier. Conversely, for a good family **F**, we can associate a good solution x in the following manner: for every [p, q] $\in$ **F** we set $x_j = \theta_{pq}$ for all $j \in [p, q]$. Hence there is a one-to-one correspondence between good solutions and good families. The algorithms described in this paper maintain good families. The algorithms explicitly maintain only the family **F** but not the corresponding solution x, which can be uniquely determined from **F**.

Consider a good family **F**. We call a pair of adjacent subsets [p, q] and [q+1, r] *out-of-order* if $\theta_{pq} > \theta_{q+1,r}$, and *in-order* otherwise. For example, if x = {2, 3, 3, 1, 1, 2, 7}, then the adjacent pair of subsets [2, 3] and [4, 5] are out-of-order, whereas all other adjacent pairs of subsets are in-order. Observe that a good family **F** is feasible if and only if the family **F** has no out-of-order pairs of subsets. This result in view of Theorem 1 implies the following property.

Property 1. *The solution corresponding to a good family **F** is an optimal solution of the convex ordered set problem if and only if **F** has no out-of-order pair of subsets.*

Satisfying Optimality Conditions

Theorem 2 tells us that in our search for the optimal solution, we can limit our search to solutions corresponding to good families. It is easy to obtain good but infeasible families. Theorem 3 (to be proved next) tells us how to gradually convert a good but infeasible family into a good and feasible family. Before we can prove this theorem, we need to prove two lemmas.

Lemma 2. *Suppose that the subproblem P[p, q] has a single-valued optimal solution with each value equal to θ_{pq}. Let the subproblem P'[p, q] be the subproblem P[p, q] with the additional constraint that $x_q \leq \alpha$, where $\alpha \leq \theta_{pq}$. Then the subproblem P'[p, q] also has a single-valued optimal solution with each value equal to α.*

Proof. Suppose that the solution $y[p, q] = \{y_p, y_{p+1}, \ldots, y_q\}$ is an optimal solution of the subproblem P'[p, q]. This solution satisfies exactly one of the following two conditions: (i) $y_p < \alpha$ and (ii) $y_p = \alpha$. If the solution satisfies the second condition, then $y_p = y_{p+1} = \ldots = y_q = \alpha$, and the lemma is true. We will now show that the first condition leads to a contradiction.

Suppose that $y_p < \alpha$. Further, let h denote the largest index satisfying $y_p = y_{p+1} = \ldots = y_h$. Consider the function $F(p, h, \theta)$. Notice that $\theta_{pq} \leq \theta_{ph}$; for otherwise the subsequence $\{x_p, x_{p+1}, \ldots, x_q\}$ defined as $x_p = x_{p+1} = \ldots = x_h = \theta_{ph}$, and $x_{h+1} = x_{h+2} = \ldots = x_q$ is a feasible solution of P[p, q] with a lower cost and thereby contradicting that the subproblem P[p, q] has a single-valued optimal solution. Further, we have assumed that $y_p < \alpha$. Therefore, $y_p < \alpha \leq \theta_{pq} \leq \theta_{ph}$. These inequalities together with the convexity of the function $F(p, h, \theta)$ and the fact that $F(p, h, \theta)$ attains its minimum at θ_{ph} imply that the subsequence $y[p, q] = \{y_p, y_{p+1}, \ldots, y_q\}$ can be improved. To see this, let $\varepsilon = y_{h+1} - y_h$, and let $w[p, q] = \{w_p, w_{p+1}, \ldots, w_q\}$ with $w_k = y_k + \varepsilon$, $1 \leq k \leq h$, and w_k

$= y_k$, $h+1 \leq k \leq q$. Then w[p, q] is a feasible solution of the subproblem P′[p, q] with lower cost, contradicting that $\{y_p, y_{p+1}, \ldots, y_q\}$ is an optimal solution of the subproblem P′[p, q]. ■

The following result is complementary to Lemma 2 with an analogous proof.

Lemma 3. *Suppose that the subproblem P[p, q] has a single-valued optimal solution with each value equal to* θ_{pq}. *Let the subproblem P′[p, q] be the subproblem P[p, q] with the additional constraint that* $x_p \geq \alpha$ *where* $\alpha \geq \theta_{pq}$. *Then the subproblem P′[p, q] also has a single-valued optimal solution with each value equal to* α.

We are now ready to prove Theorem 3, which is the foundation stone of our algorithms for the convex ordered set problem described in Sections 5 and 6.

Theorem 3. *Suppose that each of the two adjacent subproblems P[p, q] and P[q+1, r] has a single-valued optimal solution. If* $\theta_{pq} \geq \theta_{q+1,r}$ *then the subproblem P[p, r] also has a single-valued optimal solution with* θ_{pr} *satisfying* $\theta_{q+1,r} \leq \theta_{pr} \leq \theta_{pq}$.

Proof. Clearly, if $\theta_{q+1,r} = \theta_{pq}$ then the subproblem P[p, r] has a single-valued optimal solution. We will henceforth consider the case when $\theta_{pq} > \theta_{q+1,r}$. Let $x[p, r] = \{x_p, x_{p+1}, \ldots, x_q, \ldots, x_r\}$ be an optimal (not necessarily single-valued) solution of the subproblem P[p, r]. Let $\alpha = x_q$. We claim that α satisfies $\theta_{q+1,r} \leq \alpha \leq \theta_{pq}$; for otherwise (i) if $\alpha > \theta_{pq}$ then $y[p, r] = \{y_p, y_{p+1}, \ldots, y_q, \ldots, y_r\}$ defined as $y_p = y_{p+1} = \ldots = y_q = \theta_{pq}$ and $y_k = x_k$ for $k = (q+1), \ldots, r$, is a lower cost solution of the subproblem P[p, r] compared to the solution x[p, r]; and (ii) if $\alpha < \theta_{q+1,r}$ then the solution $y[p, r] = \{y_p, y_{p+1}, \ldots, y_q, \ldots, y_r\}$ defined as $y_k = x_k$ for $k = p, \ldots, q$, and $\ldots = y_q = \theta_{q+1,r}$ and $y_{q+1} = y_{q+2} = \ldots = y_r = \theta_{q+1,r}$ is a lower cost solution of P[p, r] compared to the solution x[p, r]. Thus α satisfies $\theta_{q+1,r} \leq \alpha \leq \theta_{pq}$. Now observe that the solution $\{x_p, x_{p+1}, \ldots, x_q\}$ must be an optimal solution of the subproblem P[p, q] with the additional constraint that $x_q \leq \alpha$, and the solution $\{x_{q+1}, \ldots, x_r\}$ must be an optimal solution of the subproblem P[q+1, r] with the additional constraint that $x_{q+1} \geq \alpha$. Lemma 2 implies that $x_k = \alpha$, $p \leq k \leq q$, and Lemma 3 implies that $x_k = \alpha$, $(q+1) \leq k \leq r$. Combining the two implications yields that the subproblem P[p, r] has a single-valued optimal solution.

Converting Real Optimal Solutions to Integer Optimal Solutions

The optimal solution of a convex ordered set problem in general may or may not be integral. In some situations, we may want only an integer solution of the convex ordered set problem. We will describe a simple scheme which converts an optimal solution x of the convex ordered set problem into an optimal integer solution y in O(n) time. This result was proved for the quadratic cost case by Goldstein and Kruskal [1976]. Let $x = \{x_1, x_2, \dots, x_n\}$ be an optimal solution of the convex ordered set problem. The algorithm proceeds by considering a subset of non-integer numbers in the solution x with the same integer value and rounding them up or down to the nearest integer. Consider a subsequence $\{x_p, x_{p+1}, \dots, x_q\}$ of the solution x containing all non-integer numbers with the same value of the integer part. It is easy to show using the convexity of functions $C_j(x_j)$'s that the numbers $x_p, x_{p+1}, \dots, x_q$ will either be rounded down to $\lfloor x_p \rfloor$ or rounded up to $\lceil x_p \rceil$ consistent with the constraints (1b); for otherwise the solution can be improved. Now consider the convex ordered set problem with the additional restriction that the variables $x_p, x_{p+1}, \dots, x_q$ must be integer-valued. The facts that (i) the variables $x_p, x_{p+1}, \dots, x_q$ will either be rounded off to $\lfloor x_p \rfloor$ or to $\lceil x_p \rceil$, and that (ii) they must satisfy (1b), imply that the optimal solution of our restricted convex ordered set problem will be one of the following solutions, S[k] for all k = p-1, ..., q, defined in the following manner:

$$S[k] = \begin{cases} \{x_1, x_2, \dots, x_{p-1}, \lceil x_p \rceil, \lceil x_{p+1} \rceil, \dots, \lceil x_q \rceil, x_{q+1}, \dots, x_n & \text{for } k = p-1, \\ \{x_1, x_2, \dots, x_{p-1}, \lfloor x_p \rfloor, \lfloor x_{p+1} \rfloor, \dots, \lfloor x_k \rfloor, \lceil x_{k+1} \rceil, \dots, \lceil x_q \rceil, x_{q+1}, \dots, x_n & \text{for } p \le k < q, \\ \{x_1, x_2, \dots, x_{p-1}, \lfloor x_p \rfloor, \lfloor x_{p+1} \rfloor, \dots, \lfloor x_q \rfloor, x_{q+1}, \dots, x_n & \text{for } k = q. \end{cases}$$

We evaluate the costs of these solutions and the least cost solution among these solutions is the desired solution. Given the solution x, we can determine the cost of the solution S[p-1] in O(q-p) time. Given the cost of the solutions S[k], we can determine the cost of the solution S[k+1] in O(1) time for every k = p, ... , q, because the two solutions differ only in two terms and each term can be evaluated in O(1) time. Consequently, we can find the least cost solution among the solutions S[p-1], S[p], ... , S[q] in O(q-p) time. We can repeat this process for the modified sequence by selecting another subsequence of all non-integer numbers with the same integer part. Eventually, the entire solution of the convex ordered set problem becomes integer. The total time taken by this method is O(n).

The preceding discussion implies that obtaining an integer optimal solution of the convex ordered set problem is no more difficult than obtaining a real optimal solution. The converse result is not true and obtaining a real optimal solution is in general harder than obtaining an integer optimal solution.

5. THE GENERIC ALGORITHM

Using Theorems 2 and 3, we can obtain a straightforward algorithm for solving the convex ordered set problem. This algorithm always maintains a good family **F**. The family **F** may contain some pairs of subsets which are out-of-order. In every iteration, the algorithm selects an out-of-order pair of subsets [p, q] and [q+1, r] in **F**. Theorem 3 implies that the subproblem P[p, r] also has a single-valued optimal solution. The algorithm thus replaces the subsets [p, q] and [q+1, r] by the subset [p, r]; we refer to this process as *merging*. The algorithm repeatedly merges out-of-order pairs of subsets until there are no out-of-order pairs. It follows by Property 1 that the good family finally obtained by the algorithm is feasible, and its corresponding solution is an optimal solution of the convex ordered set problem. In the literature devoted to isotonic regression problems, this algorithm is known as the PAV (Pool Adjacent Violaters) algorithm. The PAV algorithm has been studied by many researchers for different special cases of the convex ordered set problem. Stromberg [1991] considers PAV algorithm for the general convex cost case and obtains results comparable to those obtained by us in this section. We give in Figure 2 an algorithmic description of our generic convex ordered set algorithm.

algorithm *convex ordered set*;
begin
    F = [[1, 1], [2, 2], ... , [n, n]];
    while there exists an out-of-order pair of adjacent subsets in **F do**
    begin
        select a pair of out-of-order subsets [p, q] and [q+1, r] in **F**;
        replace the two subsets [p, q] and [q+1, r]
            by the subset [p, r] and update **F**;
        compute θ_{pr};
    end;
    for each subset [p, q] ∈ **F do** $x_j^* = \theta_{pq}$ for all j ∈ [p, q];
    x* is an optimal solution of the convex ordered set problem;

end;

Figure 2. The generic convex ordered set algorithm.

We now illustrate the convex ordered set algorithm using a numerical example. Suppose that the cost function for the convex ordered set problem is $\sum_{j=1}^{n} (x_j - a_j)^2$ with n = 9, $a_1 = 7$, $a_2 = 8$, $a_3 = 1$, $a_4 = 2$, $a_5 = 5$, $a_6 = 6$, $a_7 = 6$, $a_8 = 9$, and $a_9 = 4$. For the quadratic cost function F(p, q, θ), θ_{pq} is given by $\sum_{j=p}^{q} a_j/(q-p+1)$. The table shown in Figure 3 gives the details of the solutions obtained during different iterations of the algorithm. In this illustration, we selected the leftmost out-of-order pair of subsets violating the optimality condition for merging. The pair of subsets selected during an iteration is shown in bold type. The optimal solution found by the algorithm is S = {4.5, 4.5, 4.5, 4.5, 5, 6, 6, 6.5, 6.5}.

Iteration 1	**[p, q]**	[1,1]	[2,2]	[3,3]	[4,4]	[5,5]	[6,6]	[7,7]	[8,8]	[9,9]
	θ_{pq}	7	**8**	**1**	2	5	6	6	9	4
	x[p, q]	{7}	{8}	{1}	{2}	{5}	{6}	{6}	{9}	{4}
Iteration 2	**[p, q]**	[1,1]	[2,3]		[4,4]	[5,5]	[6,6]	[7,7]	[8,8]	[9,9]
	θ_{pq}	7	**4.5**		**2**	5	6	6	9	4
	x[p, q]	{7}	{4.5, 4.5}		{2}	{5}	{6}	{6}	{9}	{4}
Iteration 3	**[p, q]**	[1,3]			[4,4]	[5,5]	[6,6]	[7,7]	[8,8]	[9,9]
	θ_{pq}	**5.3**			**2**	5	6	6	9	4
	x[p, q]	{5.3, 5.3, 5.3}			{2}	{5}	{6}	{6}	{9}	{4}
Iteration 4	**[p, q]**	[1,4]				[5,5]	[6,6]	[7,7]	[8,8]	[9,9]
	θ_{pq}	4.5				5	6	6	**9**	**4**
	x[p, q]	{4.5, 4.5, 4.5. 4.5}				{5}	{6}	{6}	{9}	{4}
Iteration 5	**[p, q]**	[1,4]				[5,5]	[6,6]	[7,7]	[8,9]	
	θ_{pq}	4.5				5	6	6	6.5	
	x[p, q]	{4.5, 4.5, 4.5. 4.5}				{5}	{6}	{6}	{6.5, 6.5}	

Figure 3. Illustrating the generic convex ordered set problem.

We now analyze the worst-case complexity of the generic convex ordered set algorithm. First we consider the time needed to identify the out-of-order pairs of subsets. At the beginning of the algorithm, there are at most n out-of-order pairs of subsets.

Subsequently, whenever a merge operation is performed, a new out-of-order pair may be created involving the newly created subsets. Using simple data structures, we can easily keep track of the pairs of out-of-order subsets and select them in O(1) time per pair and in O(n) total time. Consequently, identifying the out-of-order pairs of subsets is not a bottleneck operation in the algorithm.

We next consider the merge operation. Each merge operation decreases the number of subsets by one; hence, there will be at most n-1 merge operations. The bottleneck operation in a merge operation is the computation of θ_{pr} for the subset [p, r] and this involves determining the minimum of the convex function F(p, q, θ) = $\sum_{j=p}^{q} C_j(a_j - \theta)$. We have seen in Section 3 that finding an integer optimal solution of a convex function requires O(log U) function evaluations. Each evaluation of the function F(p, q, θ) takes O(n) time since it may involve as many as n function evaluations, each of which can be performed in O(1) time (from Assumption 2). Hence the following theorem.

Theorem 4. *The generic convex ordered set algorithm obtains an optimal integer solution of the convex ordered set problem in $O(n^2 \log U)$ time.*

It is easy to see that if we want to determine an optimal fractional solution of the convex ordered set problem where the fraction has a denominator of K, then the convex ordered set algorithm would take $O(n^2 \log(UK))$ time.

6. AN IMPROVED CONVEX ORDERED SET PROBLEM

In this section, we will describe an improved convex ordered set algorithm that determines an optimal integer solution of the convex ordered set problem in O(n log U) time. The improved algorithm uses a scaling technique in the generic algorithm described in Section 5 to obtain a speedup by a factor of O(n). Scaling techniques are widely used in the literature to improve the running times of combinatorial algorithms. We refer the reader to the book of Ahuja, Magnanti and Orlin [1993] for a discussion of scaling algorithms as well as its many applications to network optimization problems.

A scaling algorithm typically decomposes an optimization problem into a series of approximate problems and gradually refines the approximation. In the convex ordered set algorithm described in Section 5, the computation of θ_{pq} was a bottleneck operation.

We needed θ_{pq} to identify out-of-order pairs of subsets. (In this section, θ_{pq} denotes the optimal integer solution of the function of $F(p, q, \theta)$ since we are interested in the optimal integer solution.) The scaling algorithm computes θ_{pq} approximately as $\theta^{\Delta}_{pq} = \Delta \lfloor \theta_{pq}/\Delta \rfloor$, which is the largest integral multiple of Δ less than or equal to θ_{pq}. The algorithm performs a number of scaling phases: we call a scaling phase with a specific value of Δ as the *Δ-scaling phase*. The algorithm starts with $\Delta = 2^{\lfloor \log(U+1) \rfloor}$ and in each subsequent scaling phase decreases Δ by a factor of 2. Eventually, Δ becomes 1 and the algorithm terminates with an optimal integral solution of the convex ordered set problem.

In the Δ-scaling phase, the algorithm maintains θ^{Δ}_{pq} for each subset $[p, q] \in \mathbf{F}$, where $\theta^{\Delta}_{pq} = \Delta \lfloor \theta_{pq}/\Delta \rfloor$. Since we are interested in the optimal integer solution of the convex ordered set problem, $\theta^{1}_{pq} = \theta_{pq}$. Therefore, if $\Delta = 1$ then $\theta^{\Delta}_{pq} = \theta_{pq}$. The definition of θ^{Δ}_{pq} implies the following property:

Property 2. $\theta^{\Delta}_{pq} \leq \theta_{pq} < \theta^{\Delta}_{pq} + \Delta$.

Our scaling algorithm also uses the following lemma.

Lemma 4. *For a pair of adjacent subsets [p, q] and [q+1, r],*

(a) if $\theta^{\Delta}_{pq} > \theta^{\Delta}_{q+1,r}$, then $\theta_{pq} > \theta_{q+1,r}$; and

(b) if $\theta^{\Delta}_{pq} < \theta^{\Delta}_{q+1,r}$, then $\theta_{pq} < \theta_{q+1,r}$.

Proof. Observe that if $\theta_{pq} \leq \theta_{q+1,r}$ then $\theta^{\Delta}_{pq} \leq \theta^{\Delta}_{q+1,r}$. The contrapositive of this result is result in part (a). The proof of part (b) is similar. ■

Similar to the generic algorithm described in Section 5, our scaling algorithm maintains a good family $\mathbf{F}$ of subsets; that is, for each subset $[p, q] \in \mathbf{F}$ the subproblem $P[p, q]$ has a single-valued optimal solution. For a good family $\mathbf{F}$, we can associate a Δ-good solution in the following manner: for every subset $[p, q] \in \mathbf{F}$, we set $x_j = \theta^{\Delta}_{pq}$ for all

$j \in [p, q]$. Our scaling algorithm always maintains a good family and θ^{Δ}_{pq} for every subset $[p, q] \in$ **F**, so that the corresponding good solution can be easily obtained. In the Δ-scaling phase, we define a pair of adjacent subsets [p, q] and [q+1, r] to be *Δ-out-of-order* if $\theta^{\Delta}_{pq} > \theta^{\Delta}_{q+1,r}$, and *Δ-in-order* otherwise. We call a family **F** to *be Δ-optimal* if it contains no Δ-out-of-order pair of adjacent subsets.

We are now in a position to describe the scaling algorithm for the convex ordered set problem. We give an algorithmic description of the algorithm in Figure 4. The algorithm starts with a sufficiently large value of Δ and a family **F** which is Δ-optimal. It then repeatedly calls a procedure improve-approximation(**F**, Δ) which takes a 2Δ-optimal family **F** and converts it into a Δ-optimal family **F**. The procedure first computes θ^{Δ}_{pq} for each subset $[p, q] \in$ **F**. We will show later in Lemma 5 how it computes θ^{Δ}_{pq} using $\theta^{2\Delta}_{pq}$. It then identifies Δ-out-of-order pairs of subsets (say, [p, q] and [q+1, r]) and replaces them by the merged subset [p, r]. It then computes θ^{Δ}_{pr}. When there are no Δ-out-of-order pairs of subsets in the family **F**, the procedure terminates. The algorithm repeats this process until Δ =1, at which point the solution associated with the family **F** satisfies the optimality conditions and the algorithm terminates with an optimal solution of the convex ordered set problem.

algorithm *improved convex ordered set*;
begin
  $\Delta := 2^{\lfloor \log(U+1) \rfloor}$;
  F := [[1, 1], [2, 2], … , [n, n]];
  if $l < 0$ then temp := -Δ **else** temp := 0;
  for each i := 1 to n **do** θ^{Δ}_{pq} := temp;
  while Δ > 1 **do** *improve-approximation(**F**, Δ)*;
  for each subset $[p, q] \in$ **F** **do** $x^*_j = \theta^1_{pq}$ for all $j \in [p, q]$;
  x* is an optimal solution of the convex ordered set problem;
end;

```
procedure improve- approximation(F, Δ);
begin
    Δ := Δ/2;
    for each subset [p, q] ∈ F do compute θ^Δ_pq ;
    while the family F is not Δ-optimal do
    begin
        select a Δ-out-of-order pair of subsets [p, q] and [q+1, r];
        replace them by the subset P[p, r] and update F;
        compute θ^Δ_pr ;
    end;
end;
```

Figure 4. The improved convex ordered set algorithm.

We will now discuss the worst-case complexity of the algorithm. The algorithm executes the procedure improve-approximation O(log U) times. We will show that the procedure can be implemented in O(n) time, thus giving a time bound of O(n log U) for the algorithm. The potential bottleneck step in the algorithm is the computation of θ^{Δ}_{pq}. The procedure uses $\theta^{2\Delta}_{pq}$ to compute θ^{Δ}_{pq}. The following lemma establishes a relationship between $\theta^{2\Delta}_{pq}$ and θ^{Δ}_{pq}.

Lemma 5. $\theta^{2\Delta}_{pq} \leq \theta^{\Delta}_{pq} \leq \theta^{2\Delta}_{pq} + \Delta$.

Proof. Property 2 implies that $\theta^{\Delta}_{pq} \leq \theta_{pq}$ (Result 1) and $\theta_{pq} < \theta^{\Delta}_{pq} + \Delta$ (Result 2). Property 2 also implies that $\theta^{2\Delta}_{pq} \leq \theta_{pq}$ (Result 3) and $\theta_{pq} < \theta^{2\Delta}_{pq} + 2\Delta$ (Result 4). Combining Result 2 and Result 3 yields $\theta^{2\Delta}_{pq} < \theta^{\Delta}_{pq} + \Delta$, and therefore $\theta^{2\Delta}_{pq} \leq \theta^{\Delta}_{pq}$ (because both sides are integral multiples of Δ). This establishes the first inequality in the statement of the lemma. Combining Result 1 and Result 4 yields $\theta^{\Delta}_{pq} < \theta^{2\Delta}_{pq} + 2\Delta$, and therefore $\theta^{\Delta}_{pq} \leq \theta^{2\Delta}_{pq} + \Delta$, establishing the second inequality in the lemma and completing the proof of the lemma. ■

At the beginning of the procedure improve-approximation in the Δ–scaling phase, we compute θ^{Δ}_{pq} for every subset [p, q] ∈ **F**. From the previous scaling phase, we know the value of $\theta^{2\Delta}_{pq}$. It follows from Lemma 5 that $\theta^{\Delta}_{pq} = \theta^{2\Delta}_{pq}$ or $\theta^{\Delta}_{pq} = \theta^{2\Delta}_{pq} + \Delta$. If $\theta^{2\Delta}_{pq} + \Delta >$

u, then clearly $\theta^{\Delta}_{pq} = \theta^{2\Delta}_{pq}$; otherwise we proceed further. It follows from the convexity of the function F(p, q, θ) and the fact that F(p, q, θ) attains its minimum at θ_{pq}, that if $\theta^{2\Delta}_{pq} + \Delta \le \theta_{pq}$ then $\theta^{\Delta}_{pq} = \theta^{2\Delta}_{pq} + \Delta$; otherwise $\theta^{\Delta}_{pq} = \theta^{2\Delta}_{pq}$. We check whether $\theta^{2\Delta}_{pq} + \Delta \le \theta_{pq}$ in the following manner. Let $\beta = \theta^{2\Delta}_{pq} + \Delta$. We compute F(p, q, β-1) and F(p, q, β). If F(p, q, β) ≤ F(p, q, β-1), then $\theta^{\Delta}_{pq} = \theta^{2\Delta}_{pq} + \Delta$; otherwise $\theta^{\Delta}_{pq} = \theta^{2\Delta}_{pq}$. This computation takes O(p-q+1) time for the subset [p, q] and O(n) time for all the subsets in the family **F**.

The algorithm also determines the value of θ^{Δ}_{pr} for the subset [p, r] obtained by merging the subsets [p, q] and [q+1, r]. Notice that we merge the subsets [p, q] and [q+1, r] in the Δ-scaling phase only if $\theta^{\Delta}_{pq} > \theta^{\Delta}_{q+1,r}$. If this merging occurs then $\theta^{2\Delta}_{pq} = \theta^{2\Delta}_{q+1,r}$; for if $\theta^{2\Delta}_{pq} > \theta^{2\Delta}_{q+1,r}$, we would have merged the subsets in the 2Δ-scaling phase, and if $\theta^{2\Delta}_{pq} < \theta^{2\Delta}_{q+1,r}$ then by Lemma 4(b) $\theta^{\Delta}_{pq} < \theta^{\Delta}_{q+1,r}$, giving a contradiction in both the cases. Since $\theta^{2\Delta}_{pq} = \theta^{2\Delta}_{q+1,r}$, it follows that $\theta^{2\Delta}_{pr} = \theta^{2\Delta}_{pq} = \theta^{2\Delta}_{q+1,r}$. Lemma 5 implies that $\theta^{\Delta}_{pr} = \theta^{2\Delta}_{pr}$ or $\theta^{\Delta}_{pr} = \theta^{2\Delta}_{pr} + \Delta$ whichever happens to give a lower value of the function F(p, q, θ). If $\theta^{2\Delta}_{pr} + \Delta > u$, then clearly $\theta^{\Delta}_{pr} = \theta^{2\Delta}_{pr}$; otherwise we proceed further. If $\theta^{2\Delta}_{pr} + \Delta \le \theta_{pr}$ then $\theta^{\Delta}_{pr} = \theta^{2\Delta}_{pr} + \Delta$; otherwise $\theta^{\Delta}_{pr} = \theta^{2\Delta}_{pr}$. We check whether $\theta^{2\Delta}_{pr} + \Delta \le \theta_{pr}$ in the following manner. Let $\beta = \theta^{2\Delta}_{pr} + \Delta$. We next compute F(p, r, β-1) and F(p, r, β). Now notice that F(p, r, θ) = F(p, q, θ) + F(q+1, r, θ). Since both F(p, q, θ) and F(q+1, r, θ) have been determined earlier in the algorithm for both θ = β–1 and θ = β, we can compute both F(p, r, β-1) and F(p, r, β) in O(1) time. If F(p, r, β) ≤ F(p, r, β-1), then $\theta^{\Delta}_{pr} = \theta^{2\Delta}_{pr} + \Delta$; otherwise $\theta^{\Delta}_{pr} = \theta^{2\Delta}_{pr}$. We have thus shown that an execution of the procedure improve-approximation takes O(n) time, giving us the following theorem.

Theorem 5. *The improved convex ordered set obtains an integer optimal solution of the convex ordered set problem in O(n log U) time.*

7. SPECIAL CASES OF THE CONVEX ORDERED SET PROBLEM

In this section, we will study three special cases of the convex ordered set problem and develop faster algorithms. We consider the quadratic cost case (L_2 norm), the minimax cost case (L_∞ norm), and the rectilinear cost case (L_1 norm). We consider both the weighted and unweighted (that is, $c_j = 1$ for all j) cases. In most cases, we obtain optimal real-valued solutions. Using the method described in Section 4, an optimal real-valued solution can be converted to an optimal integer-valued solution in O(n) time, if needed.

Quadratic Cost Ordered Set Problem

We will adapt the generic convex ordered set algorithm for this case. For the quadratic cost ordered set problem, we assume that the cost function is given by $\sum_{j=1}^{n} c_j(x_j - a_j)^2$, where the c_j's and a_j's are specified constants. For this cost function, $F(p, q, \theta) = \sum_{j=p}^{q} c_j(\theta - a_j)^2$. It is easy to verify that $F(p, q, \theta)$ will achieve its minimum value at $\theta_{pq} = (\sum_{j=p}^{q} c_j a_j)/(\sum_{j=p}^{q} c_j)$. In this case, a straightforward computation of θ_{pq} takes O(q-p) = O(n) time and in view of our previous discussion we can solve the convex ordered set problem in $O(n^2)$ time. However, we can do even better by slightly modifying the algorithm. For each subset [p, q] in the family **F**, we maintain two additional values α_{pq} and β_{pq} defined as follows: $\alpha_{pq} = \sum_{j=p}^{q} c_j a_j$ and $\beta_{pq} = \sum_{j=p}^{q} c_j$. Notice that $\theta_{pq} = \alpha_{pq}/\beta_{pq}$. When we start the algorithm, each subset consists of a singleton element and the computation of α, β, and θ takes a total of O(n) time. Now consider the merging of two subsets [p, q] and [q+1, r]. Notice that $\alpha_{pr} = \alpha_{pq} + \alpha_{q+1,r}$, $\beta_{pr} = \beta_{pq} + \beta_{q+1,r}$, and $\theta_{pr} = \alpha_{pr}/\beta_{pr}$. Then, we can determine θ_{pr} for the subset generated by a merge operation in O(1) time. The total time for the merge operations over the entire algorithm is O(n). Consequently, the modified convex ordered set algorithm can solve the quadratic cost problem in O(n) time. This time bound matches the best available time for the quadratic cost isotonic regression problem due to several researchers (see Best and Chakravarti [1990]).

Theorem 6. *The unweighted and weighted versions of the quadratic cost ordered set problem can be solved in O(n) time.*

Minimax Cost Ordered Set Problem

We will next adapt the generic convex ordered set for the minimax cost ordered set problem for the unit weight case. In this case, we assume that the cost function is given by $\max\{|x_j - a_j|: 1 \leq j \leq n\}$. For this cost function, $F(p, q, \theta) = \max\{|\theta - a_j|: p \leq j$

$\leq q\}$. It is easy to verify that F(p, q, θ) achieves its lowest value at θ_{pq} = ½[max{a_j: $p \leq j \leq q$}+ min{a_j: $p \leq j \leq q$}]. We can use a technique similar to that for the quadratic cost case to compute the θ value for a merged problem in O(1) time. For each subset [p, q], we maintain α_{pq} = max{a_j: $p \leq j \leq q$}and β_{pq} = min{a_j: $p \leq j \leq q$}, and whenever two subsets [p, q] and [q+1, r] are merged to form the subset [p, r], we update α_{pr} = max{α_{pq}, $\alpha_{q+1,r}$}, β_{pr} = min{β_{pq}, $\beta_{q+1,r}$}, and recompute θ_{pr} = ½{$\alpha_{pr} + \beta_{pr}$}. All of these operations can be performed in O(1) time per merge and in O(n) time over the entire algorithm. This time bound matches the O(n) time bound obtained by Liu and Ubhaya [1997] for the unweighted case.

We next consider the weighted version of the minimax ordered set problem, that is, where the objective function is to minimize max{$c_j|x_j - a_j|$: $1 \leq j \leq n$}. Our approach for the unweighted case does not apply to the weighted version because there is no closed form formula for determining the value of θ for which F(p, q, θ) attains the lowest value. In this case, = max{$c_j|x_j - a_j|$: $p \leq j \leq q$} = max{max{$c_j(x_j - a_j)$, $c_j(a_j - x_j)$}: $p \leq j \leq q$}, which is the upper envelope of 2(q-p+1) linear functions. If the slopes of all linear functions are already arranged in non-decreasing order, then their upper envelope as well as the lowest point on the envelope can be determined in O(q-p+1) = O(n) time. Thus when we apply the generic convex ordered set algorithm to this case, we maintain for each subproblem P[p, q] the slopes of the linear functions in F(p, q, θ) in non-decreasing order. When we merge two subproblems P[p, q] and P[q+1, r], then the sets maintaining the slopes are merged too, and the merging of these two ordered sets can be done in O(r-p) = O(n) time. Consequently, each iteration of the generic convex ordered set algorithm can be implemented in O(n) time, and the total time taken by the algorithm is $O(n^2)$. This time bound matches the $O(n^2)$ time bound obtained by Liu and Ubhaya [1997] for the weighted case.

We next consider the adaptation of the improved convex ordered set algorithm for the weighted minimax case. Let a_{min} = min{a_j : $1 \leq j \leq n$} and a_{max} = max{a_j : $1 \leq j \leq n$}. It is easy to see that in the optimal solution, each x_j will satisfy $a_{min} \leq x_j \leq a_{max}$. Consequently, for this case U = max{|*l*|, |u|} = max{|a_j| : $1 \leq j \leq n$}. When the improved convex ordered set algorithm is applied to this problem, it obtains an integer optimal solution of the problem in O(n log U) time. Under the similarity assumption, that is, U = $O(n^k)$ for some integer k, the preceding time bound becomes O(n log n). Under this assumption or whenever $\log U = o(n)$, it improves the time bound of $O(n^2)$ due to Liu and Ubhaya [1997]. We have shown the following result.

Theorem 7. *The unweighted minimax cost ordered set problem can be solved in O(n) time and its weighted version in $O(n^2)$ time. An integer optimal solution of the weighted minimax cost ordered set problem can be obtained in O(n log U) time.*

Rectilinear Cost Ordered Set Problem

We will now consider adaptation of the generic convex ordered set algorithm for the rectilinear costs, that is, where the objective function is $\sum_{j=1}^{n} c_j|x_j - a_j|$, with c_j's and a_j's being specified constants. For this problem, $F(p, q, \theta) = \sum_{j=p}^{q} c_j|\theta - a_j|$, and it is well known (see, for example, Francis and White [1976]) that a "median solution" is its optimal solution. A solution θ equal to some a_j with no more that half of the sum of c_j's on either side is said to be a *median solution*, that is, $\theta = a_k$ for some k satisfying $\sum_{j=1}^{k-1} c_j \le \frac{1}{2} \sum_{j=1}^{n} c_j$ as well as $\sum_{j=k+1}^{n} c_j \le \frac{1}{2}\sum_{j=1}^{n} c_j$. We can determine the exact value of θ_{pq} for a subset P[p, q] in O(q-p) = O(n) time by applying a median finding algorithm. Clearly, in this case the exact computation of θ_{pq} takes O(q-p) = O(n) time and, consequently, we can solve the convex ordered set problem in $O(n^2)$ time.

We next consider the adaptation of the improved convex ordered set algorithm for the rectilinear cost problem. For this case, $U = \max\{|a_j| : 1 \le j \le n\}$. When the improved convex ordered set algorithm is applied to this problem, it runs in O(n log U) time. We will show that a simple transformation can be used to modify the problem so that all data is integer and U = n and, consequently, the improved convex ordered set algorithm will solve this problem in O(n log n) time.

The improved convex ordered set algorithm proceeds by determining θ_{pq} values for the subproblems P[p, q] obtained during its execution. The θ_{pq} value is the minimum value of the function $F(p, q, \theta) = \sum_{j=p}^{q} c_j|a_j - \theta|$. It is well known (see, for example, Francis and White [1976]) that a "median solution" is the optimal solution of the function $F(p, q, \theta)$. A solution θ equal to some a_j with no more that half of the sum of c_j's on either side is said to be a *median solution*, that is, $\theta = a_k$ for some k satisfying

$\sum_{a_j<a_k} c_j \le \frac{1}{2} \sum_{j=p}^{q} c_j$ and $\sum_{a_j>a_k} c_j \le \frac{1}{2} \sum_{j=p}^{q} c_j$. Now observe from this formula that while determining the median solution, the magnitude of the a_j's is unimportant; it is the relative ordering of the a_j's with respect to one-another that is important. This observation allows us to use the following method to determine the median solution for any subproblem. We sort a_j's in the non-decreasing order. Let $\sigma(j)$ denote the position of a_j in this order. For example, if $n = 5$, $a_1 = 50$, $a_2 = 10$, $a_3 = 70$, $a_4 = 20$, and $a_5 = 40$, then $\sigma(1) = 4$, $\sigma(2) = 1$, $\sigma(3) = 5$, $\sigma(4) = 2$, and $\sigma(5) = 3$. Observe that the median solution for subproblem P[p, q] is a_k for some k satisfying $\sum_{\sigma(j)<\sigma(k)} c_j \le \frac{1}{2} \sum_{j=p}^{q} c_j$ and $\sum_{\sigma(j)>\sigma(k)} c_j \le \frac{1}{2} \sum_{j=p}^{q} c_j$. We next replace each a_j by $\sigma(j)$ and apply the improved convex ordered set algorithm. For the modified problem, all data is integer and U = O(n), hence the improved convex ordered set algorithm would determine an optimal solution of this problem in O(n log n) time. In the optimal solution y*, each number varies between 1 to n. We can convert the optimal solution y* of the modified problem into an optimal solution x* of the original problem in the following manner: $x_j^* = a_j$ if and only if $y_j^* = \sigma(j)$. Notice that the optimal solution x* of the original problem may or may not be integer.

We have thus shown that the rectilinear cost ordered set problem can be solved in O(n log n) time. This improves the best available running time of $O(n^2)$ for the same problem due to Chakravarti [1989]. Chakravarti [1989] claims that his algorithm can be implemented in O(n log n) time, but implementation details are not provided to obtain this time bound. Hence we could not verify the claimed O(n log n) running time. We summarize the preceding discussion in the form of the following theorem:

Theorem 8. *The unweighted and weighted versions of the rectilinear cost ordered set problem can be solved in O(n log n) time.*

ACKNOWLEDGMENTS

We gratefully acknowledge support from the Office of Naval Research under contract ONR N00014-96-1-0051 as well as a grant from the UPS Foundation.

REFERENCES

Ahuja, R. K., T. L. Magnanti, and J. B. Orlin. 1993. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, NJ.

Ahuja, R. K., and J. B. Orlin. 1996. Routing and Scheduling Algorithms for ADART. Working Paper, Sloan School of Management, MIT, Cambridge, MA.

Ahuja, R. K., and J. B. Orlin. 1997a. Inverse optimization, Part I: General problem and linear programming. Working Paper, Sloan School of Management, MIT, Cambridge, MA.

Ahuja, R. K., and J. B. Orlin. 1997b. Inverse optimization, Part II: Network flow problems. Working Paper, Sloan School of Management, MIT, Cambridge, MA.

Ahuja, R. K., and J. B. Orlin. 1997c. Combinatorial algorithms for inverse network flow problems. Working Paper, Sloan School of Management, MIT, Cambridge, MA.

Barlow, R. E., D. J. Bartholomew, D. J. Bremner, and H. D. Brunk. 1972. *Statistical Inference under order restrictions.* Wiley, New York.

Bazaraa, M., H. Sherali, and C. M. Shetty. 1992. *Nonlinear Programming: Theory and Algorithms*. John Wiley and Sons.

Best, M. J., and N. Chakravarti. 1990. Active set algorithms for isotonic regression; a unifying framework. *Mathematical Programming* **47**, 425-439.

Chakravarti, N. 1989. Isotonic median regression; a linear programming approach. *Mathematics of Operations Research* **14**, 303-308.

Desrosiers, J., Y. Dumas, M. M. Solomon, and F. Soumis. 1995. Time constrained routing and scheduling. *Handbooks in Operations Research and Management Science, Volume 8: Network Routing*, North Holland, Amsterdam.

Francis, R. L., and J. A. White. 1976. *Facility Location and Layout.* Addison-Wesley, Reading, MA.

Goldstein, A. J., and J. B. Kruskal. 1976. Least-square fitting by monotonic functions having integer values. *Journal of American Statistical Association* **71**, 370-373.

Kontoravdis, K, and J. F. Bard. 1994. A GRASP for the vehicle routing problem with time windows. *ORSA Journal on Computing* **7**, 10-23.

Liu, M. H., and V. A. Ubhaya. 1997. Integer isotone optimization. *SIAM Journal on Optimization* **7**, 1152-1159.

Menendez, J. A., and B. Salvador. 1987. An algorithm for isotonic median regression. *Computational Statistics and Data Analysis* **5**, 399-406.

Robertson, T., and F. T. Wright. 1980. Algorithms in order restricted statistical inference and the Cauchy mean property value. *Annals of Statistics* **8**, 645-651.

Robertson, T., F. T. Wright, and R. L. Dykstra. 1988. *Order Restricted Statistical Inference*. John Wiley & Sons, New York.

Stromberg, U. 1991. An algorithm for isotonic regression with arbitrary convex distance function. *Computational Statistics and Data Analysis* **11**, 205-219.

Tarantola, A. 1987. *Inverse Problem Theory: Methods for Data Fitting and model Parameter Estimation*. Elsevier, Amsterdam.

Ubhaya, V. A. 1974a. Isotone optimization, I. *Journal of Approximation Theory* **12**, 146-159.

Ubhaya, V. A. 1974b. Isotone optimization, II. *Journal of Approximation Theory* **12**, 315-342.

APPENDIX

Theorem 1. *If a subproblem P[p, q] has a single-valued optimal solution, then the following sets of conditions are satisfied:*

(a) $\theta_{pj} \geq \theta_{pq}$ *for all j = p, p+1, ..., q; and* *(7a)*

(b) $\theta_{jq} \leq \theta_{pq}$ *for all j = p, p+1, ..., q.* *(7b)*

Proof. We first prove the necessity of these conditions. Suppose that the subproblem P[p, q] has a single-valued optimal solution x[p, q] = $\{x_p, x_{p+1}, \ldots, x_q\}$ with $x_p = x_{p+1} = \ldots = x_q = \theta_{pq}$. We will show that this solution will satisfy the conditions stated in (7). We will prove this result by contradiction. Suppose that the subproblem P[p, q] violates (7). Then there are two possibilities to consider.

Case 1a. *The subproblem P[p, q] violates (7a).* Suppose that for some index h, $1 \leq h < q$, $\theta_{ph} < \theta_{pq}$. Define the solution y[p, q] = $\{y_p, y_{p+1}, \ldots, y_q\}$ as follows: $y_p = y_{p+1} = \ldots. = y_h = \theta_{ph}$, and $y_{h+1} = y_{h+2} = \ldots = y_q = \theta_{pq}$. Since $\theta_{ph} < \theta_{pq}$, the solution y[p, q] satisfies the feasibility constraints (4b) and hence is a feasible solution of the subproblem P[p, q]. Let z[p, q] and z'[p, q] respectively denote the costs of the solutions x[p, q] and y[p, q]. It is easy to see that z[p, q] - z'[p, q] = F(p, h, θ_{pq}) - F(p, h, θ_{ph}). Since the function F(p, h, θ) is a strictly convex function of θ and achieves its minimum value at $\theta = \theta_{ph}$, it follows that F(p, h, θ_{pq}) > F(p, h, θ_{ph}). This implies that z[p, q] > z'[p, q], contradicting that the solution x[p, q] is an optimal solution of the subproblem P[p, q].

Case 1b. *The subproblem P[p, q] violates (7b).* Suppose that for some index h, $1 < h \leq q$, the subsequence satisfies $\theta_{hq} > \theta_{pq}$. Using the reasoning analogous to that in Case 1, it can be shown that the solution y[p, q] = $\{y_p, y_{p+1}, \ldots, y_q\}$ defined as $y_p = y_{p+1} = \ldots = y_{h-1} = \theta_{pq}$ and $y_h = y_{h+1} = \ldots = y_q = \theta_{hq}$ is a lower cost solution than x[p, q], thereby contradicting that the solution x[p, q] is an optimal solution of the subproblem P[p, q].

We will now prove the sufficiency of the theorem. Suppose that a subproblem P[p, q] satisfies the conditions in (7). We will show that the subproblem P[p, q] has a single-valued optimal solution. Let x[p, q] = $\{x_p, x_{p+1}, \ldots, x_q\}$ be the optimal solution of the subproblem P[p, q]. If $x_p = x_q = \theta_{pq}$, then the solution x[p, q] is a single-valued solution and the theorem is true. If $x_p = x_q \neq \theta_{pq}$, then by the convexity of the function F[p, q, θ], the solution y[p, q] defined as $y_j = \theta_{pq}$ for all j = p, p+1, … , q is a lower cost solution than x[p, q], contradicting the optimality of the solution x[p, q]. We will henceforth consider the case when $x_p < x_q$. There are two cases to consider: $x_p < \theta_{pq}$ and $x_p \geq \theta_{pq}$, which we will consider separately.

Case 2a. $x_p < \theta_{pq}$. Choose the index r so that $x_p = x_{p+1} = \dots = x_r < x_{r+1}$. Let $\alpha = \min\{x_{r+1}, \theta_{pq}\}$. Define the solution y[p, q] as follows: $y_j = \alpha$ for all j = p, p+1, … , r and $y_j = x_j$ for all j = r+1, … , q. It follows from the convexity of the function F(p, r, θ) and the fact that $\theta_{pr} \geq \theta_{pq}$ (from (7a)) that the solution y[p, q] is a lower cost solution than x[p, q], contradicting the optimality of x[p, q].

Case 2b $x_p \geq \theta_{pq}$. Since $x_p < x_q$, it follows that $x_q > \theta_{pq}$. Choose the index r so that $x_{r-1} < x_r = \dots = x_q$. Let $\alpha = \max\{x_{r-1}, \theta_{pq}\}$. Define the solution y[p, q] as follows: $y_j = x_j$ for all j = p, … , r-1 and $y_j = \alpha$ for all j = r, r+1, … , q. It follows from the convexity of the function F(r, q, θ) and the fact that $\theta_{rq} \geq \theta_{pq}$ (from (7b)) that the solution y[p, q] is a lower cost solution than x[p, q], again contradicting the optimality of x[p, q].

We have thus shown that all cases except the one in which $x_p = x_q = \theta_{pq}$ result in contradictions. This completes the proof of the sufficiency and also the proof of the theorem. ■