Capturing Incomplete Information in Resource Allocation Problems through Numerical Patterns

Arun Marar
Warren B. Powell

Department of Operations Research and Financial Engineering,
Princeton University, Princeton, NJ 08544

October 17, 2006

Abstract

We look at the problem of optimizing complex operations with incomplete information where the missing information is revealed indirectly and imperfectly through historical decisions. Incomplete information is characterized by missing data elements governing operational behavior and unknown cost parameters. We assume some of this information may be indirectly captured in historical databases through flows characterizing resource movements. We can use these flows or other quantities derived from these flows as "numerical patterns" in our optimization model to reflect some of the incomplete information. Since regression trees represent probabilistic distributions of a numerical value specific to a vector of data elements known as a "state", they drive our methodology for representing information in resource allocation models. We use a popular goodness-of-fit measure known as the Cramer-Von Mises metric based on the *empirical distribution function* (EDF) as the foundation of our approach. We then use a hybrid approach of solving a cost model with a term known as the "pattern metric" that minimizes the deviations of model decisions from observed quantities in a historical database. We see that when using this methodology, our problem of representing information pertaining to numerical patterns is combinatorial in the order statistics of the decision variables in the optimization model. We present a novel iterative method to solve this problem to get a stationary solution. Results with real-world data from a large freight railroad are presented.

Introduction

A challenge in modeling a real-world system such as a complex freight logistics operation is the problem of optimizing the system under incomplete information. Normally we solve a cost function to reflect actual behavior. For complex problems such a methodology is inadequate because of elements of data not observable to the modeler. In some cases, however, this incomplete information is revealed, although imperfectly, through past decisions made by humans since typically in actual operations humans make decisions with more information than is available to the modeler. As most freight logistics operations are modeled as resource allocation problems, information regarding actual decisions are represented in a historical database as flows characterizing resource movements. Based on a desired observation statistic, these flows may be aggregated or used to derive certain quantities that capture some of the missing information. Such quantities are known as “numerical patterns” implying there is a natural ordering among them.

The following examples illustrate some real-world situations where the modeler does not have complete information:

- In a trucking operation the average length of haul for sleepers is high, but that does not mean sleepers are not assigned to short loads. We can observe from history the actual distribution of the length of the load assigned to a particular sleeper. In this case, an observation statistic is a particular sleeper and the numerical pattern is the length of its load.
- Consider flowing shipments through a less-than-truckload (LTL) network. The capacity of a truck is a function of the weights of the individual shipments and the densities of these shipments. Very often the density of a shipment is not available to the modeler. The modeler can observe from a historical database the distribution of the weights of all the trucks traversing the link represented by a particular origin-destination pair. In this case the observation statistic is a truck and the numerical pattern is the weight of the truck.
- The capacity of a transfer terminal is a function of operating conditions such as the

manpower available and weather and the modeler does not have this information in his database. However, distributions of terminal capacities at these transfer nodes on a specific day of the week for a particular month can be observed in a historical database and these distributions capture some of the missing information underlying the operating conditions. In this case the observation statistic is a specific day of the week for a particular month and the numerical pattern is the capacity of the transfer terminal.

- In a locomotive scheduling model every train requires a certain amount of locomotive power based on the tonnage of the train and the terrain over which the track is laid. There may be a steep grade on the train route that warrants the use of more locomotive power than that calculated based on tonnage considerations. The modeler does not have this information in his database, but can observe in a historical database a pattern of over-powered trains (labeled so relative to the power calculated by the modeler) over certain segments of the train route through a statistic known as the "horsepower per trailing ton" which reflects the ratio of the amount of locomotive power (expressed in horsepower) to the tonnage of the train. Here the observation statistic is a train and the numerical pattern is the "horsepower per trailing ton" assigned to the train in history.

A method that takes into account missing elements of data is to solve the optimization model under uncertainty using stochastic programming techniques (see Sen & Higle (1999),Birge (1997)). These techniques are useful when we know what the missing elements of data are and can observe them after the fact. For example while solving a real-time model we may have to forecast the tonnage on a train before making a decision. In this case we are able to observe the actual tonnage on the train after the train has departed and thus we are able to build a posterior distribution of the train tonnage. In our problem we are dealing with data elements that are not directly observable, even after the fact.

There has been past research in the realm of inverse optimization that uses observable decisions from a real-world system in an optimization model. Inverse optimization algorithms aim to perturb the cost parameters of a model while minimizing this perturbation, such that the observed decisions are optimal with respect to the perturbed cost parameters. Inverse optimization has been applied to a variety of situations such as shortest path problems (Burton et al. (1997),Burton & Toint (1994)), capacitated facility location problem (Bitran et al. (1981)) and general linear programming and network problems (Zhang & Liu (1999),Sokkalingam et al. (1999)). Inverse optimization techniques cannot be used to solve our problem since these methods assume the observed solutions from history to be optimal whereas in applications such as ours even if they are made with complete information, decisions may be suboptimal, due to the humans' inability to process large amounts of data.

To capture incomplete information in an optimization model we look at pattern recognition systems to represent observed decisions from a database in a model since a pattern is simply a set of data elements and the decision pertaining to this set of data elements. The focus of this paper is to present a modeling framework that combines the use of cost functions (bottom-up representation of operations that can be quantified as costs) with a representation of historical decisions (top-down representation of operations that are captured through patterns). The main contribution of this paper is that we introduce and study for the first time integrating an engineering cost model with historical patterns that represent quantities or flows as a way of capturing incomplete information.

The paper is organized as follows. In section 1 we present the principles of pattern recognition and the basis of our approach. In section 2 we introduce the literature on a type of pattern recognition system known as a regression tree that is particularly suited to our problem. In section 3 we introduce the literature on empirical distribution function (EDF) and its relation to some goodness-of-fit metrics for continuous distributions. We show how we can apply this methodology to our problem of representing information from regression trees. In section 4 we present the model formulation for representing information from regression trees using a function known as the "pattern metric" that penalizes deviations of model decisions from observed quantities in a historical database. The functional form for the pattern metric is derived using the Cramer-Von Mises test statistic. In section 5 we present our hybrid approach of solving a cost function along with a pattern metric. We see that our methodology involves solving a combinatorial problem arising due to the order statistics of the quantities we represent in the pattern metric. We avoid the combinatorial optimization directly by an iterative methodology that does not require the need to specify additional constraints to account for the order statistics. Given an initial starting solution, we prove convergence of this method to a stationary point in a finite number of iterations. In section 6 we present the experimental design and the measures of model performance that can be used to evaluate our methodology. We present results of our research using real-world data from a large freight railroad. In section 7 we present our conclusions.

1 Basic Approach

The historical database for a typical logistics operation consists of a sequence of decisions or actions in time and elements of data governing these decisions. It is natural to look at pattern recognition systems to mimic human behavior since they aim to identify a decision with a particular set of data elements. Pattern recognition is a mature area and extensive treatment of these techniques are found in (Fayyad et al. (1996),Devroye et al. (1996),Nilsson (1990)). In the event where the pattern is a number such as a flow or a quantity such as the horsepower per trailing ton, the term "pattern regression" is used instead of pattern recognition. For example a simple pattern representing the decision to assign locomotive power to a train might include the following data elements as shown:

{Train Tonnage, Train Origin, Train Destination, Train Type}

In complex operations such as the one our problem represents, a historical database does not reflect all the data elements that defines actual decisions. This is due to missing elements of data that are not retained in the database, but govern actual decisions. The traditional method of optimization using cost functions has the ability to handle vectors of large dimensionality since optimization models are useful in capturing the global effects of a decision. However, we are limited in our ability to observe data elements with the same level of detail from a historical database. For this reason we cannot adopt a pure pattern recognition approach to solve our problem, but need to retain the cost function approach and complement it with pattern recognition techniques.

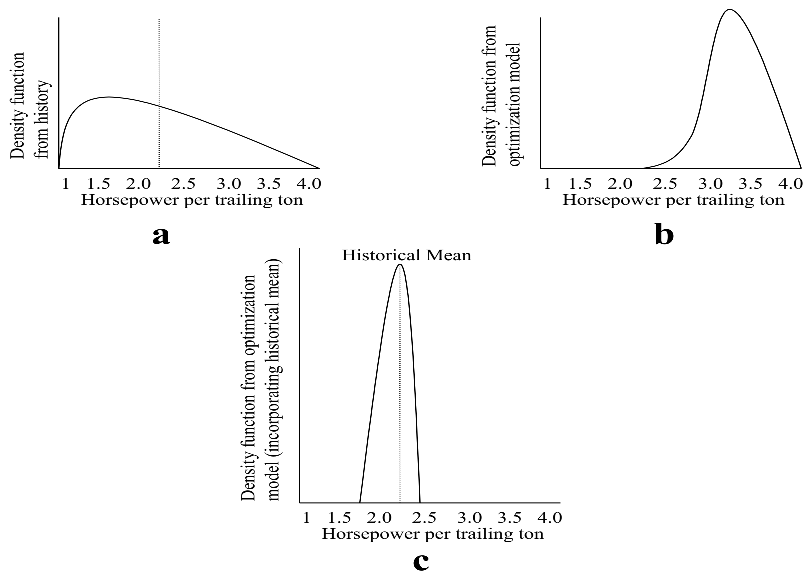

Figure 1: (a) & (b): Observed horsepower in history and model, (c):Using the historical mean as information in the optimization model

Pattern regression is simply a technique to identify a number such as a quantity or flow that pertains to a set of data elements, also known as a state, based on observed patterns from a historical database. A particular class of pattern regression methods known as regression trees (Breiman et al. (1984),Quinlan (1993)) are of interest because of their ability to handle data elements in a state that are *numerical* (these data elements are real numbers) and *categorical* (these data elements take values in a finite set not having any natural ordering). For example the tonnage of a train is a numerical attribute whereas the locomotive type is a categorical attribute taking a value in the finite set consisting of the different locomotive types. Thus the methodology we propose in this paper for incorporating historical decisions in an optimization model is based on a typical regression tree.

Consider the case of observing the locomotive power assigned to a particular train from history as depicted in figure 1(a). Such a distribution is very common in history since we work with simple states that do not convey all the information underlying the decision. For instance in figure 1(a) we may not have all the information regarding why a particular train was assigned a horsepower-per-trailing ton of 4.0 on a certain day instead of the more likely value of 2.0. It is possible that a typical optimization model might produce the distribution shown in figure 1(b). This distribution not only does not have the same mean, it does not match any of the higher moments. We could try to force the optimization model to match the mean, but the result might be the distribution shown in figure 1(c). The problem with just matching the mean is that we are trying to force every observation to match the mean. Since the actual distribution is much wider, clearly it is acceptable to deviate from the mean, but we want the overall average to match the historical average. In the next section we introduce the literature review on regression trees and develop the notation to represent numerical patterns in a resource allocation model.

2 Regression Trees for Resource Allocation Models

Regression trees are rule-based systems that identify a state which generally consists of both categorical and numerical attributes by a real number, which in our problem is a flow or a quantity. The concept of a regression tree has its origin in the Automatic Interaction Detection (AID) program (Morgan & Sonquist (1963)). A popular regression tree system used widely is Classification and Regression Tree (CART) (Breiman et al. (1984)). A regression tree in general has a binary tree structure due to hierarchical bipartitioning of the attribute set into sets with finer detail.

We present the general methodology underlying a regression tree and show how we can extend this to the class of resource allocation problems. We use the superscript 'h' to indicate whenever we are working with a historical database or data pertaining to past events in general. Thus we have:

$a =$ Vector of attributes for a resource.

$\mathcal{A}^h =$ Attribute space of a in history.

$\mathcal{D}_a =$ Set of decisions d that can be applied to a resource with attribute vector a.

$x^h_{ad} =$ Number of times decision d is applied to a resource with attribute vector a where $a \in \mathcal{A}^h$ and $d \in \mathcal{D}_a$.

For instance, in a locomotive scheduling model a locomotive state could be as shown below:

$$a = \{\text{Location, Time, Locomotive ID, Locomotive Type, Horsepower}\}$$

If the decision d is to assign this locomotive to a train then x^h_{ad} takes a value 1, otherwise its value is 0.

As we noted previously, we typically do not observe in history all the information underlying a particular decision and any pattern regression function that predicts a unique quantity ignores these missing elements of data. For these reasons the model has to work with a simple vector of data elements such as a resource attribute vector a using a probability based pattern regression that accounts for missing information. From now on our use of a state will unambiguously refer to a resource attribute vector unless explicitly defined otherwise.

In general not all the components of the resource attribute vector are used for building a regression tree. For instance in a locomotive scheduling model a simple state could be as shown below:

$$a = \{\text{Location, Time, Locomotive ID, Locomotive Type, Delay(in minutes)}\}$$

The possible instances of the attribute "Location" could be the list of nodes in the rail network. An aggregated resource attribute vector $\hat{a}$ would look like:

$$\hat{a} = \{\text{Region, Time, Locomotive ID, Locomotive Type, Delay(in minutes)}\}$$

Although the new state vector $\hat{a}$ preserves the same cardinality as the original state a, the possible space of the attribute vector $\hat{a}$ is reduced due to the aggregation on the component "Location".

Analogous to aggregation on the resource attribute vector we can define aggregation on the decision. For example consider the set of decisions pertaining to a locomotive state a as shown below:

$$\mathcal{D}_a = \{\text{Move to San Francisco, Move to Seattle, Move to Boston, Move to New York}\}$$

We could map San Francisco and Seattle to the "North-West" region and map Boston and New York to the "North-East" region. In this case an aggregated decision $\hat{d}$ is an element of the following set:

$$\hat{\mathcal{D}}_a = \{\text{Move to North-West, Move to North-East}\}$$

We define a family of observation statistics denoted by the set $\mathcal{S}$. An example of an observation statistic is a train departing Los Angeles for Kansas City. Some of the quantities we can observe for each instance of this observation statistic from a historical database are total flow of locomotives, horsepower per trailing ton, total horsepower assigned to the train, horsepower of a specific locomotive type assigned to the train and so forth. For each instance of the observation statistic there are, in general, multiple pairs of (a, d) that are associated with this instance. If we are concerned with the total flow of locomotives any locomotive assigned to the train from Los Angeles to Kansas will pertain to that instance of the observation statistic. We also define:

N_s^h = Number of observations of the observation statistic $s \in \mathcal{S}$ in a historical database.

$\mathcal{L}_{sk}^h$ = Set of all pairs (a, d) in a historical database pertaining to the k-th observation of the observation statistic $s \in \mathcal{S}$, $k \in \{1, 2, \ldots, N_s^h\}$, $d \in \mathcal{D}_a$, $a \in \mathcal{A}^h$.

We assume that for each observation statistic $s \in \mathcal{S}$ and for each observation k of the statistic s we generate a quantity $\hat{q}_{sk}^h$ based on a desired objective. We use the notation "$\hat{q}$" to indicate that sometimes the quantities may be derived from aggregations of state-decision pairs. For example, for a particular train leaving Los Angeles for Kansas a quantity derived could be the total horsepower assigned to the train. In our problem the quantities that we are typically interested in are derived from observed flows of resources derived from a historical database in a manner as shown:

$$\hat{q}_{sk}^h = \sum_{\forall (a,d) \in \mathcal{L}_{sk}^h} q_{sad} x_{ad}^h, \quad \forall k \in \{1, 2, \ldots, N_s^h\}, \quad \forall s \in \mathcal{S} \tag{1}$$

q_{sad} is the contribution of the state-action pair (a, d) in the quantity for the observation statistic $s \in \mathcal{S}$. To illustrate, consider a locomotive state as shown below:

$$a = \{\text{Location, Locomotive ID, Locomotive Type, Horsepower(HP)}\}$$

A typical decision d can be characterized by a train as shown below:

$$d = \{\text{Origin, Destination, Train ID, Train Type, Tonnage}\}$$

The decision variable x^h_{ad} takes a value 1 in equation (1) if we assign this locomotive to the train, otherwise its value is 0.

If we wish to derive the distribution of the horsepower per trailing ton for trains between a specific origin and destination which is the observation statistic, then $q_{sad} = \frac{a_{HP}}{d_{Tonnage}}$ in (1) where a_{HP} denotes the rated horsepower attribute of a locomotive attribute state a and $d_{Tonnage}$ represents the tonnage on the particular train implied by the decision d. If the quantity we are interested in is simply the number of locomotives assigned to the train we set $q_{sad} = 1$ for all state-action pairs (a, d) in the set $\{(a, d) \in \mathcal{L}^h_{sk}\}$.

For each observation statistic $s \in \mathcal{S}$, the cumulative probability function underlying the vector of quantities derived using equation(1) for all $k \in \{1, 2, \ldots, N^h_s\}$ characterizes the *pattern regression* denoted by ψ_s for the observation statistic $s \in \mathcal{S}$.

The goal of this research is to incorporate the pattern regression ψ_s in an optimization framework since this reveals some of the missing information not captured in a cost function. In this section we showed how to derive the attribute-based pattern regression ψ_s for an observation statistic $s \in \mathcal{S}$. In the next section we show how we can define a distance metric for our probability regression ψ_s based on popular goodness-of-fit metrics in statistics.

3 The Empirical Distribution Function

In this section we look at how we can use the attribute-based pattern regression function for a quantity as derived in (1) to develop a framework for representing information pertaining to distribution of a quantity. To develop this methodology we need to introduce the concept of an *empirical distribution function* (EDF) (Read & Cressie (1988)) and see how it relates to the decision variables in the optimization model. We define:

$N_s =$	Number of times the observation statistic $s \in \mathcal{S}$ is observed in the model.
$\hat{q}_{sk}(x) =$	The quantity corresponding to the k-th instance of the observation statistic $s \in \mathcal{S}$, $k \in \{1, 2, \ldots, N_s\}$, in the optimization model as a function of the model decision variable x.

It should be noted that the number of times the observation statistic $s \in \mathcal{S}$ is observed in the model is in general different from the number of times the observation statistic is observed in history. For example, we may observe a train leaving Los Angeles for Kansas 20 times in our historical database, but we may see the same train only five times in the model and even this number could change depending on the time horizon in the model.

We can generate the order statistics of the quantities evaluated over all the observations as shown below:

$$\hat{q}_{s(1)}(x) \leq \hat{q}_{s(2)}(x) \leq \ldots \leq \hat{q}_{s(N)}(x) \quad \text{where } \ N = N_s$$

The notation $\hat{q}_{s(k)}$ which is standard in statistics literature is interpreted as the k^{th} largest quantity observed among all the instances of the observation statistic $s \in \mathcal{S}$ in the optimization model. For example, a particular train may be available for assignment in the model five times. The quantity we may be interested in for this observation statistic could be the amount of horsepower assigned to each instance of this train in the model. In this case the order statistics is applied to the five values of horsepower assigned in the model to all the instances of this train. The *empirical distribution function* (EDF) for the order statistics $\hat{q}_{s(k)}$ in the optimization model is given by:

$$F_s(\hat{q}_{s(k)}(x)) \;\; = \;\; \frac{k}{N_s}, \quad \forall k\{1, 2, \ldots, N_s\}, \quad \forall s \in \mathcal{S} \tag{2}$$

Thus the empirical distribution function F_s is a step function that increases by $\frac{1}{N_s}$ at each numerical value in the vector of ordered statistics.

An important application of the EDF in statistics is in testing the hypothesis that a sample of data is derived from a fully specified continuous distribution (for instance, a normal or an exponential distribution) also known as the null hypothesis. This procedure is commonly known as a goodness-of-fit test for a continuous distribution. Each instance of data is assumed to be an independent and identically distributed random variable. The goodness-of-fit test statistic evaluates the sample of data using a metric and the null hypothesis is rejected if this measure exceeds a certain threshold.

In our case the null hypothesis is the pattern regression ψ_s and the sample of data are the values of horsepower assigned to each instance of a particular train in the model. It should be noted that the attribute-based pattern regression ψ_s is derived from a finite history of data and hence the obtained cumulative probability function is not a continuous function, but a step function just as the EDF in equation (2).

Two popular goodness-of-fit measures based on the EDF are the Kolmogorov-Smirnov (K-S) and the Cramer-Von-Mises (C-V) test statistics (Chakravarthi et al. (1967)). The Kolmogorov-Smirnov test statistic for our problem is as shown below:

$$D_s^{KS}(x) \quad = \quad \max_{1 \le k \le N_s} |\psi_s(\hat{q}_{s(k)}(x)) - F_s(\hat{q}_{s(k)}(x))|, \quad \forall s \in \mathcal{S} \tag{3}$$

Substituting for $F_s(\hat{q}_{s(k)})$ in equation (3) using the expression in (2) we get:

$$D_s^{KS}(x) \quad = \quad \max_{1 \le k \le N_s} \left| \psi_s(\hat{q}_{s(k)}(x)) - \frac{k}{N_s} \right|, \quad \forall s \in \mathcal{S}$$

The Cramer-Von Mises test statistic is given by:

$$D_s^{CV}(x) \quad = \quad \left[\sum_{k=1}^{N_s} \left(\psi_s(\hat{q}_{s(k)}(x)) - \frac{2k-1}{2N_s} \right)^2 \right] + \frac{1}{12N_s}, \quad \forall s \in \mathcal{S} \tag{4}$$

We are interested in a function that represents the pattern regression ψ_s exactly in an optimization framework. Clearly this function is closely related to a goodness-of-fit statistic on the model decision variables that attempts to infer if the sample of data (here the quantities assigned in the model to all instances of the observation statistic $s \in \mathcal{S}$) is derived from a historical distribution (here the pattern regression ψ_s). Alternately we can look at optimizing a function similar to a goodness-of-fit statistic to capture information from a historical distribution. In the next section we propose a modification of the Cramer-Von Mises test statistic as the functional form for representing information characterized by ψ_s in resource allocation problems.

4 The Pattern Metric

A method to induce an optimization model to make decisions that closely mimic a historical distribution of quantities is to include a term in the objective function that penalizes deviations of the corresponding quantities in the model from the observed historical distribution. We consider capturing this behavior by solving a cost function with a function that penalizes deviations of quantities evaluated in the optimization model from observed quantities from a historical database. This penalty function is referred to as the *pattern metric*. We develop the pattern metric using the functional forms of goodness-of-fit metrics introduced in the previous section.

Large-scale resource allocation problems representing operations in freight logistics are often solved as dynamic models where the dynamic problem is represented as an iterative time-staged network where each stage is solved separately (Powell & Carvalho (1997)). Thus it is convenient to use a goodness-of-fit measure that is separable in the data instances pertaining to the observation statistic $s \in \mathcal{S}$ since instances of this observation statistic may be realized at different stages in the dynamic model. Thus we propose developing our functional form of the pattern metric using the C-V test statistic because this statistic is separable in each instance of the observation statistic. The C-V statistic introduced in the previous section (note that the second term in equation (4) is omitted since it is a constant) is given by:

$$D_s^{CV}(x) \quad = \quad \sum_{k=1}^{N_s} \left(\psi_s(\hat{q}_{s(k)}(x)) - \frac{2k-1}{2N_s} \right)^2, \quad \forall s \in \mathcal{S} \tag{5}$$

The C-V goodness-of-fit measure in (5) is a function of the decision variables in the optimization model. The discontinuous nature of the attribute-based pattern regression ψ_s in the decision variables poses problems if the exact form of the test statistic is used in an optimization model. Even if we approximate ψ_s by a continuous distribution and fix the order statistics *a priori* we have to deal with nonconvexity issues since cumulative probability functions are in general nonconvex.

We note that ψ_s is a cumulative probability function and hence is a monotonically increasing function in the order statistics of the decision variables, that is, we have:

$$\psi_s(\hat{q}_{s(1)}(x)) \leq \psi_s(\hat{q}_{s(2)}(x)) \leq \ldots \leq \psi_s(\hat{q}_{s(N)}(x)) \text{ where } N = N_s, \quad \forall s \in \mathcal{S}$$

Instead of using a function that measures the difference between the cumulative distribution function and the empirical distribution function we propose to work with the following function:

$$\tilde{D}_s^{CV}(x) \quad = \quad \sum_{k=1}^{N_s} \left(\hat{q}_{s(k)}(x) - \psi_s^{-1}\left(\frac{2k-1}{2N_s}\right)\right)^2, \quad \forall s \in \mathcal{S} \tag{6}$$

where we use ψ_s^{-1} to denote the inverse cumulative function. We note that since the order statistics themselves have a monotone property we can use a distance metric that measures deviations of the order statistics of the quantities in the model from the corresponding order statistics derived from a historical database given by:

$$\hat{q}_{s(k)}^{h*} \quad = \quad \psi_s^{-1}\left(\frac{2k-1}{2N_s}\right), \quad \forall k \in \{1, 2, \ldots, N_s\}, \quad \forall s \in \mathcal{S} \tag{7}$$

If we have a large sample of historical data it may be possible to approximate ψ_s by a strictly increasing continuous function. In this case we can see that if for a particular pattern $\tilde{D}_s^{CV} = 0$ it implies that the C-V test statistic $D_s^{CV} = 0$. This is because if $\hat{q}_{s(k)} = \psi_s^{-1}(\frac{2k-1}{2N_s})$ for strictly increasing probability functions then we have:

$$\psi_s(\hat{q}_{s(k)}(x)) \quad = \quad \frac{2k-1}{2N_s}$$

Thus the functional form presented in (6) is closely related to the C-V test statistic and also offers us a distribution-free representation of historical patterns. If we fix the order statistics *a priori*, the function indicated in equation (6) is convex. We define the *pattern metric* to be:

$$H(x) \quad = \quad \sum_{\forall s \in \mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s(k)}(x) - \hat{q}^{h*}_{s(k)}\right)^2$$

We assume that the modeler solves a functional approximation of the real-world objective by an engineering cost function $C(x)$ that is convex in x. We present the optimization model in a pattern metric as shown below:

$$x^*(\theta) \quad = \quad \arg\min C(x) + \theta \sum_{s \in \mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s(k)}(x) - \hat{q}^{h*}_{s(k)}\right)^2 \tag{8}$$

subject to:

$$Ax \quad = \quad b$$

$$x \geq 0$$

where θ is a positive scaling factor. We use x to denote the complete vector of decision variables $\{x_{ad}\}_{a \in \mathcal{A}, d \in \mathcal{D}_a}$. Using equation (1) we can derive the quantities $\hat{q}$ in the model as a function of the model decision variables x similar to the quantities derived from history.

The concept behind the optimization in equation (8) is that it combines engineering costs with prior decisions which capture information that is not available to the modeler. In the next section we show how the problem in (8) is a combinatorial problem the order statistics are not known *a priori*.

5 Optimizing the Model with a Pattern Metric

In this section we present our methodology to solve the model with a pattern metric indicated in equation (8). We introduce the notation defining the order statistics of the quantities and then present two algorithms that we can use to solve the optimization model with a pattern metric. We also present some theoretical properties regarding the algorithms.

5.1 The Order Statistics

The order statistics of the quantities cannot be fixed in the model in (8) since we do not know the value of the decision variables *a priori*. This implies that our problem of representing information pertaining to the numerical patterns is a combinatorial problem. For each observation statistic $s \in \mathcal{S}$ we define:

$$\begin{aligned}\Phi_s &= \text{The set of permutations of the index vector } \{1, 2, \ldots, N_s\}.\\ \phi_s &= \text{A generic permutation of the index vector } \{1, 2, \ldots, N_s\},\ \phi_s \in \Phi_s.\\ \phi_{sk} &= \text{The } k\text{-th element of the generic permutation } \phi_s,\ k \in \{1, 2, \ldots, N_s\}.\end{aligned}$$

For example, if we observe a train from Los Angeles to Kansas City five times the cardinality of the index set corresponding to this observation statistic is five and a particular permutation of this index vector could be $\{2, 5, 4, 1, 3\}$. In this case the third element of this permutation is the fourth instance of the train observed in the optimization model. We denote $\phi = \{\phi_s\}_{s \in \mathcal{S}}$ as a generic permutation vector over the set of observation statistics $\mathcal{S}$. We define the set:

$$\Phi = \times_{s \in \mathcal{S}} \Phi_s$$

with element $\phi \in \Phi$. We then define the quantities under the permutation vector ϕ as shown below:

$$\hat{q}_{s\phi(k)}(x) = \text{The quantity corresponding to the } \phi_{sk}\text{-th instance of the observation statistic } s \in \mathcal{S} \text{ in the model}, \quad k \in \{1, 2, \ldots, N_s\},\ \phi \in \Phi.$$

We can represent the optimization model in a pattern framework indicated in (8) as a combinatorial problem as shown below:

$$\phi^*(\theta) = \arg\min_{\phi \in \Phi} C(x) + \theta \sum_{s \in \mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi(k)}(x) - \hat{q}^{h*}_{s(k)}\right)^2 \tag{9}$$

subject to:

$$\begin{aligned} \hat{q}_{s\phi(k)}(x) &\leq \hat{q}_{s\phi(k+1)}(x), \quad \forall k \in \{1,2,\ldots,(N_s-1)\}, \quad \forall s \in \mathcal{S} \qquad (10) \\ Ax &= b \\ x &\geq 0 \end{aligned}$$

The optimal decision variables $x^*(\theta)$ for the optimization model in equation (9) are calculated as shown below:

$$x^*(\theta) \;=\; \arg\min_{\phi=\phi^*} C(x) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi(k)}(x) - \hat{q}^{h*}_{s(k)}\right)^2$$

subject to the same set of constraints as the model in equation (9).

Constraint (10) ensures that for each permutation ϕ the order statistics conditions are satisfied. The model in (9) is an extremely difficult problem to solve for two reasons. The first reason is the size of the set Φ (for an observation statistic $s \in \mathcal{S}$, $|\Phi_s| = (N_s)!$) which can explode for large-scale resource allocation problems. The second reason is that for each index $\phi \in \Phi$ we require solving a constrained optimization problem further complicated by the order statistics constraints in (10).

A recent research interest in combinatorial optimization is the application of heuristic methods such as tabu search (Glover (1990)), simulated annealing (Kirkpatrick et al. (1983)) and genetic algorithms (Holland (1975)) to solve some large-scale combinatorial problems such as the traveling salesman and quadratic assignment problems. The success of these methods in solving combinatorial problems is due to the ease with which a potential solution can be evaluated in these problems. In our problem as indicated by the formulation in (9), evaluating a new point is relatively expensive and hence we are limited in our ability to use these techniques easily.

There is another technique that is suited for highly nonconvex problems introduced in Pardalos (1989). In this method a number of "starting points" are evaluated separately using a modified version of the Frank-Wolfe method (Bazaraa et al. (1993),Frank & Wolfe (1956)) that ensures convergence to a local optimum. By evaluating a number of starting points the algorithm attempts to arrive at a local optimum that is close to the global optimum. In our problem the formulation in (9) the objective function does not provide any guidance on generating a set of starting points other than the most obvious one which is given by the solution to the model: $\min C(x)$ subject to: $Ax = b, x \geq 0$. In the next section we present an iterative method that, given this initial feasible solution, our algorithm converges to a stationary solution in a finite number of iterations.

5.2 Algorithms

We present the method to solve the optimization model in (9). We fix the order statistics *a priori* depending on the solution x^0 obtained from solving the model: $\min C(x)$ subject to: $Ax = b, x \geq 0$. Let $\hat{q}^0 = \{\hat{q}^0_{sk}\}_{s\in\mathcal{S},k\in\{1,2,\ldots,N_s\}}$ be the the initial vector of quantities corresponding to this solution. Let $\phi^0 \in \Phi$ be the permutation that defines the order statistics of the solution vector $\hat{q}^0$, that is:

$$\hat{q}^0_{s\phi(k)} \leq \hat{q}^0_{s\phi(k+1)}, \quad \phi = \phi^0 \in \Phi, \quad \forall k \in \{1,2,\ldots,(N_s-1)\}, \quad \forall s \in \mathcal{S}$$

We solve the following problem:

$$x^*(\theta) \;=\; \arg\min C(x) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi(k)}(x) - \hat{q}^{h*}_{s(k)}\right)^2 \tag{11}$$

subject to:

$$\begin{aligned} \phi &= \phi^0 \in \Phi \\ \hat{q}_{s\phi(k)}(x) &\leq \hat{q}_{s\phi(k+1)}(x), \quad \forall k \in \{1,2,\ldots,(N_s-1)\}, \quad \forall s \in \mathcal{S} \\ Ax &= b \\ x &\geq 0 \end{aligned} \tag{12}$$

This model is a convex programming problem since we know *a priori* the permutation vector ϕ^0 and this model yields a feasible solution to the original problem in (9). Since the order statistics are fixed before solving (11) we refer to this algorithm as the *fixed ordering* (FO) method.

We propose an alternate method that simplifies the model formulation in (11). Instead of solving the model stated in (11) we can replace this model by a sequence of simpler models that do not require the constraints defining the order statistics stated in (12). Thus at an iteration m of this sequence we solve the following model as shown below:

$$x^m(\theta) \quad = \quad \arg\min C(x) + \theta \sum_{s \in \mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi(k)}(x) - \hat{q}^{h*}_{s(k)}\right)^2 \tag{13}$$

subject to:

$$\begin{aligned} \phi &= \phi^{m-1} \in \Phi \\ Ax &= b \\ x &\geq 0 \end{aligned}$$

The permutation ϕ^{m-1} defines the order statistics of the following vector of quantities:

$$\hat{q}^{m-1} \quad = \quad \{\hat{q}^{m-1}_{sk}\}_{s \in \mathcal{S}, k \in \{1,2,\ldots,N_s\}}$$

which is derived from the solution vector x^{m-1}. As our method involves a reordering (that is the permutation is iteration-specific) of the order statistics after every iteration of the sequence we call our methodology the *sequential ordering* (SO) method.

We show in the next section that solving a sequence of the model indicated in (13) converges to a stationary point in a finite number of iterations. We propose this method because it has two major advantages:

- We are guaranteed a solution that is as good as (in many cases better than) the solution obtained by the FO method, that is at every index m in the sequence of models solved in (13) we have:

$$C(x^m) + \theta \sum_{s \in \mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}^m_{s\phi(k)} - \hat{q}^{h*}_{s(k)}\right)^2 \leq C(x^*) + \theta \sum_{s \in \mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}^*_{s\phi^0(k)} - \hat{q}^{h*}_{s(k)}\right)^2$$

where $\phi = \phi^m$, $\phi, \phi^0 \in \Phi$ and

$$\hat{q}^* \quad = \quad \{\hat{q}^*_{sk}\}_{s\in\mathcal{S},k\in\{1,2,\ldots,N_s\}}$$

is the vector of quantities derived from the solution vector x^*, the solution to the model in (11).

- We do not have to deal with constraints specifying the order statistics. Note that the number of additional constraints needed to specify the order statistics in (12) is given by $\sum_{s\in\mathcal{S}}(N_s - 1)$.

5.3 Properties

The superiority of the SO method over the FO method is established in the following theorem which states that even though the order statistics are not specified as constraints, the pattern metric evaluated over the new order statistics (since the ordering may change if the constraints are not specified in the SO methodology) is guaranteed to be less than the pattern metric evaluated for the order statistics of the current solution. We prove the following theorem:

Theorem 5.1 *Given a vector of strictly positive solutions $\{\hat{q}_{sk}\}_{k\in\{1,2,\ldots,N_s\}}$ we have:*

$$\min_{\phi'\in\Phi_s} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi'(k)} - \hat{q}^{h*}_{s(k)}\right)^2 \quad = \quad \sum_{k=1}^{N_s} \left(\hat{q}_{s(k)} - \hat{q}^{h*}_{s(k)}\right)^2$$

Proof. We first note that:

$$\sum_{k=1}^{N_s} \left(\hat{q}_{s\phi'(k)}\right)^2 \quad = \quad \sum_{k=1}^{N_s} \left(\hat{q}_{s(k)}\right)^2, \quad \forall \phi' \in \Phi_s$$

since the vectors $\{\hat{q}_{s(k)}\}_{k\in\{1,2,\ldots,N_s\}}$ and $\{\hat{q}_{s\phi'(k)}\}_{k\in\{1,2,\ldots,N_s\}}$ are permutations of each other for all $\phi' \in \Phi_s$. Thus we have:

$$\begin{aligned}\sum_{k=1}^{N_s}\left(\hat{q}_{s\phi'(k)} - \hat{q}^{h*}_{s(k)}\right)^2 - \sum_{k=1}^{N_s}\left(\hat{q}_{s(k)} - \hat{q}^{h*}_{s(k)}\right)^2 &= \\ -2\sum_{k=1}^{N_s}\hat{q}^{h*}_{s(k)}\left(\hat{q}_{s\phi'(k)} - \hat{q}_{s(k)}\right), \ \ \forall\phi' \in \Phi_s \qquad (14)\end{aligned}$$

Using lemma 7.1 where we set $y = \hat{q}^{h*}_s$, $x = \hat{q}_s$,$n = N_s$ and $\Phi_n = \Phi_s$ in the lemma we see that the right hand side of equation (14) is nonnegative. Thus it follows that:

$$\sum_{k=1}^{N_s}\left(\hat{q}_{s\phi'(k)} - \hat{q}^{h*}_{s(k)}\right)^2 \ \geq \ \sum_{k=1}^{N_s}\left(\hat{q}_{s(k)} - \hat{q}^{h*}_{s(k)}\right)^2) \ \ \forall\phi' \in \Phi_s$$

Since $\{\hat{q}_{s(k)}\}_{k\in\{1,2,\ldots,N_s\}}$ is a valid permutation on the decision variables $\{\hat{q}_{sk}\}_{k\in\{1,2,\ldots,N_s\}}$ we have the following relation:

$$\min_{\phi'\in\Phi_s}\sum_{k=1}^{N_s}\left(\hat{q}_{s\phi'(k)} - \hat{q}^{h*}_{s(k)}\right)^2 \ = \ \sum_{k=1}^{N_s}\left(\hat{q}_{s(k)} - \hat{q}^{h*}_{s(k)}\right)^2$$

which is the statement required to be proved.

Q.E.D.

In the next section we state and prove convergence of the SO method to a stationary point.

Proposition 5.2 *If the feasible region of the model indicated in (13) is compact, the sequence of optimal solutions obtained from solving the sequence of models solved in equation (13) converges to a stationary point in a finite number of iterations.*

Proof. The feasible region of the model indicated in (13) is denoted by $\mathcal{X}$. Since $\mathcal{X}$ is compact there exists a minimum for any convex function minimized over $\mathcal{X}$. At any iteration m of the SO methodology we solve the following problem:

$$x^m \ = \ \arg\min_{x\in\mathcal{X},\phi=\phi^{m-1}} C(x) + \theta\sum_{s\in\mathcal{S}}\sum_{k=1}^{N_s}\left(\hat{q}_{s\phi(k)} - \hat{q}^{h*}_{s(k)}\right)^2 \qquad (15)$$

We have using theorem 5.1 the following inequality:

$$\begin{aligned} C(x^m) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}^m_{s\phi^m(k)} - \hat{q}^{h*}_{s(k)}\right)^2 &\leq \\ C(x^m) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}^m_{s\phi^{m-1}(k)} - \hat{q}^{h*}_{s(k)}\right)^2 & \end{aligned} \tag{16}$$

The left hand side of equation (16) gives the objective for the model whose solution is x^m since the pattern metric reflects the order statistics with respect to the quantities derived from this solution. Now consider the model solved at iteration $m+1$. We have:

$$x^{m+1} = \arg\min C(x) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi(k)} - \hat{q}^{h*}_{s(k)}\right)^2$$

where $\phi = \phi^m$. Since x^m is a feasible solution to the above problem we have:

$$\begin{aligned} C(x^{m+1}) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi^m(k)} - \hat{q}^{h*}_{s(k)}\right)^2 &\leq \\ C(x^m) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi^m(k)} - \hat{q}^{h*}_{s(k)}\right)^2 & \end{aligned} \tag{17}$$

Again using theorem 5.1 for the order statistics pertaining to the quantities derived from solution vector x^{m+1} we have:

$$\begin{aligned} C(x^{m+1}) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi^{m+1}(k)} - \hat{q}^{h*}_{s(k)}\right)^2 &\leq \\ C(x^{m+1}) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi^m(k)} - \hat{q}^{h*}_{s(k)}\right)^2 & \end{aligned}$$

From equation (17) we see that:

$$\begin{aligned} C(x^{m+1}) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi^{m+1}(k)} - \hat{q}^{h*}_{s(k)}\right)^2 &\leq \\ C(x^m) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi^m(k)} - \hat{q}^{h*}_{s(k)}\right)^2 & \end{aligned}$$

From this equation we see that the objective function corresponding to the solution x^{m+1} cannot exceed the objective function corresponding to the solution x^m. Since this is true for any iteration $m > 1$ we see that the objective function corresponding to x^m monotonically decreases with iteration m, $m \geq 1$. Since we are minimizing a convex objective function over the compact set $\mathcal{X}$ we are guaranteed convergence of the sequence of objective functions from solving the sequence of models in (15) to a stationary point in our original problem in (9).

Note that from equation (15) the only difference between solving the model at two different iterations is that the permutation vector defining the order statistics of the quantities may differ. The possible number of permutations is clearly finite and this implies that the sequence of permutation vectors generated from solving the sequence of models in (15) converges to a stationary point in a finite number of iterations. Thus $\exists \tilde{m} < \infty$ such that $\phi^{\tilde{m}} = \phi^{\tilde{m}+1}$. From equation (15) this implies that $x^{\tilde{m}} = x^{\tilde{m}+1}$ and we have the statement in our proof.

Q.E.D.

We presented a simple method in (13) that yields a stationary solution for the combinatorial problem in (9), given an initial solution. In the next section we study an experiment that tests this methodology.

6 Experimental Design and Results

Three questions arise in our research that need to be tested experimentally:

- What is the improvement gained by using historical patterns?
- What is the effect of aggregation of historical patterns on the benefit from using historical patterns?
- How effectively does the SO methodology in (13) use information derived from historical patterns?

The experiment we present utilizes real-world data representing the locomotive management problem of a large freight railroad. To measure the value of using historical patterns, we set up an experiment where we create a random sample of "unobservable" data. We then simulate human decision making by solving this problem optimally. The optimal solution is then aggregated to simulate the loss of information that occurs when polling information from a historical database. Thus in our laboratory experiment, we create our "history" by solving a function $\tilde{C}(x)$ optimally as a deterministic, flow-balancing network model with complete information as shown below:

$$x^{h*} = \arg\min_{x\in\mathcal{X}} \tilde{C}(x)$$

where the feasible region $\mathcal{X}$ is assumed to be compact. We do have access to an actual database of historical decisions but due to the complexity of actual operations we are unable to measure the quality of these solutions and hence we have to create our "history" artificially. It should be noted that in actual operations decisions are made suboptimally and our attempt to construct a "historical" database of decisions represented by x^{h*} obtained from solving $\tilde{C}$ is only an idealized characterization of the real-world.

Since the feasible region is identical when solving $\tilde{C}$ and C we have $N^h_s = N_s$ for each observation statistic $s \in \mathcal{S}$. We construct our pattern regression distribution ψ_s for each $s \in \mathcal{S}$ from the historical quantities generated by applying equation (1) to the optimal solution x^{h*}. We then use equation (7) to obtain the desired vector of quantities $\hat{q}^{h*} = \{\hat{q}^{h*}_{sk}\}_{s\in\mathcal{S},k\in\{1,2,\ldots,N_s\}}$ that represents our incomplete information in the pattern metric.

The set of observation statistics is chosen such that the numerical patterns are represented at a higher level of aggregation of the state and decision than is represented in either C or $\tilde{C}$. This makes our experiment realistic since we are unable to observe patterns from a historical database at the same level of detail that is captured in a cost function. The optimization model with the pattern metric is given by:

$$x^*(\theta) = \arg\min_{x\in\mathcal{X}} C(x) + \theta \sum_{s\in\mathcal{S}} \sum_{k=1}^{N_s} (\hat{q}_{s(k)}(x) - \hat{q}^{h*}_{s(k)})^2 \tag{18}$$

Solving the model in (18) using the traditional approach (that is $\theta = 0$) yields a suboptimal solution to the model $\tilde{C}$ since the objective $\tilde{C}$ consists of cost parameters that are not considered in C. We are interested in evaluating the improvement over this solution when the model is solved with a pattern metric (that is $\theta > 0$ in (18)).

Using the traditional method with just the cost function (that is setting $\theta = 0$ in equation (18)) we have:

$$x^*(0) \quad = \quad \arg\min_{x \in \mathcal{X}} C(x)$$

We solve a sequence of deterministic, flow-balancing models where at each index m of the sequence we solve:

$$x^m(\theta) \quad = \quad \arg \min_{x \in \mathcal{X}, \phi = \phi^{m-1}} C(x) + \theta \sum_{s \in \mathcal{S}} \sum_{k=1}^{N_s} \left(\hat{q}_{s\phi(k)}(x) - \hat{q}^{h*}_{s(k)}\right)^2, \quad m \in \{1, 2, \ldots\} \qquad (19)$$

We set ϕ^0 to the permutation pertaining to the order statistics of the solution $x^*(0)$. As we showed in proposition 5.2 the above sequence of optimal solutions converges to a stationary point. We denote the convergent solution to the sequence in (19) by $x^{SO,*}(\theta)$. We define our measure of model performance by the *improvement ratio* which is given by:

$$\eta^{IMP}(\theta) = \frac{\tilde{C}(x^*(0)) - \tilde{C}(x^{SO,*}(\theta))}{\tilde{C}(x^*(0)) - \tilde{C}(x^{h*})}$$

An improvement ratio equal to 1 means that for a particular value of θ we have obtained the optimal solution using the SO methodology of (19) . We claim that by solving the optimization model with a pattern metric as indicated in (19) we are able to get a positive improvement for a particular θ, that is we have $\eta^{IMP}(\theta) > 0$.

In the following sections we present the experimental setup detailing the generation of the vector x^{h*} and consequently the quantities that are represented in a pattern metric and the results of our research.

6.1 Experimental Setup

Our experimental setting is a locomotive management problem for a large railroad. The resource attribute state a^t for the train is as shown:

$$a^t \quad = \quad \{\text{Train ID, Origin, Destination, Tonnage, Train Type}\}, \quad a^t \in \mathcal{A}^t$$

$\mathcal{A}^t$ is the space of possible attribute vectors for a train resource. The resource attribute state of the locomotive denoted by a^l is as shown below:

$$a^l \quad = \quad \{\text{Locomotive Type, Location, Horsepower (HP)}\}, \quad a^l \in \mathcal{A}^l$$

where we use $\mathcal{A}^l$ to denote the attribute space of the locomotive attribute vector a^l. The set of decisions pertaining to a train resource with attribute vector $a \in \mathcal{A}^t$ is as shown below:

$$\mathcal{D}^t_a \quad = \quad \cup_{\{a^l \in \mathcal{A}^l : a^l_{Location} = a_{Origin}\}} \{\text{Assign train to locomotive with attribute vector } a^l\}$$

where $a^l_{Location}$ denotes the "Location" component of the attribute vector a^l. a_{Origin} denotes the attribute "Origin" of the train. Thus the decision set $\mathcal{D}^t_a$ consists of decisions where each decision means assigning to the train a locomotive with a specific attribute vector $a^l \in \mathcal{A}^l$ such that the location of the locomotive is same as the origin of the train. The set of decisions pertaining to a locomotive resource attribute vector $a \in \mathcal{A}^l$ is as shown below:

$$\mathcal{D}^l_a \quad = \quad \cup_{\{a^t \in \mathcal{A}^t : a_{Location} = a^t_{Origin}\}} \{\text{Assign the locomotive to train with attribute vector } a^t\}$$

The experimental data is obtained from a major railroad and typically represents the data for a week. The data consisted of 10557 train segments (each segment is a unique "Train ID") connecting 178 locations in the network, 12 train types and 8 locomotive types.

We consider two types of engineering costs, a positive cost denoted by the vector c^a (the cost of assigning locomotives to trains) and a negative cost denoted by the vector c^r (the reward of assigning power to a train). The objective function characterizing C has a quadratic form as shown below:

$$\begin{aligned} C(x) &= \sum_{a \in \mathcal{A}^l} \sum_{d \in \mathcal{D}_a^l} c_{ad}^a x_{ad} - \\ & \left(\sum_{a \in \mathcal{A}^t} \left[2 \left(\sum_{d \in \mathcal{D}_a^t} d_{HP} x_{ad} \right) c_{ad}^r - \left(\sum_{d \in \mathcal{D}_a^t} d_{HP} x_{ad} \right)^2 \frac{c_{ad}^r}{a_{Tonnage} * \beta} \right] \right) \end{aligned} \tag{20}$$

Please note that the decision set D_a^t is simply a set of locomotive states and hence we use d_{HP} to denote the horsepower of the locomotive. $\{c_{ad}^a\}_{a \in \mathcal{A}^l, d \in \mathcal{D}_a^l}$ and $\{c_{ad}^r\}_{a \in \mathcal{A}^t, d \in \mathcal{D}_a^t}$ characterize the vectors c^a and c^r respectively. β is a positive parameter characterizing the "horsepower per trailing ton" ratio.

The corresponding cost vectors $\tilde{c}^a$ and $\tilde{c}^r$ for the objective function $\tilde{C}$ characterizing the real-world are derived by perturbing the vectors c^a and c^r by random perturbations δ^a and δ^r in a biased sense, that is, the expectations of these random perturbations are not equal to 0. We also assume that the random vectors δ^a and δ^r are not observable to the modeler even after the fact.

Based on actual experience we believe that assigning power to a train just based on the tonnage of a train is inadequate because we do not consider the effect of the terrain over which the track is laid. This information is usually unobservable by the modeler. In our laboratory setting we attempt to capture this behavior by applying a uniformly distributed biased random multiplier on the parameter β for each train. We assume this random multiplier is completely unknown to the modeler and is denoted by δ_a^β. The objective function characterizing the real-world in our laboratory setting is as shown below:

$$\begin{aligned} \tilde{C}(x) &= \sum_{a \in \mathcal{A}^l} \sum_{d \in \mathcal{D}_a^l} \tilde{c}_{ad}^a x_{ad} - \\ & \left(\sum_{a \in \mathcal{A}^t} \left[2 \left(\sum_{d \in \mathcal{D}_a^t} d_{HP} x_{ad} \right) \tilde{c}_{ad}^r + \left(\sum_{d \in \mathcal{D}_a^t} d_{HP} x_{ad} \right)^2 \frac{\tilde{c}_{ad}^r}{a_{Tonnage} * \beta * \delta_a^\beta} \right] \right) \end{aligned} \tag{21}$$

We solve $\tilde{C}$ to generate our history x^{h*}. In our experiment the positive cost parameters characterizing $\tilde{C}$ are allowed to deviate uniformly between -100% and 200% of their respective values in C. The negative costs are allowed to deviate uniformly between -75% and 150% of their corresponding values in C. The random multiplier δ_a^β is uniform in the range $[0.7, 1.6]$. The difference between the parameters characterizing $\tilde{C}$ and the parameters characterizing C is assumed to be completely unknown to the modeler. In the next section we present how we capture the loss of information in representing numerical patterns through aggregation.

6.2 Aggregation

We consider aggregated train attribute states in our experiment to capture loss of information when representing the numerical patterns in our experiment. The level with the most detail represented by a train state $a^t \in \mathcal{A}^t$ is denoted by O-D-T-ID and consists of the attributes "Origin"(O), "Destination"(D), "Train Type"(T) and "Train ID"(ID). The aggregated states that we consider in our experiment are the O-D-T and O-D levels.

The modeler's estimate of the "horsepower per trailing ton" (HPT) for each train is given by β. However in the real-world the actual HPT may be different from β and this difference may be a function of a level of aggregation. If the HPT is specific to an individual train then δ_a^β is a function of the O-D-T-ID level. If the HPT is a function of the terrain and the train type then we capture this by letting δ_a^β be a function of the O-D-T level. If we assume the HPT is a function of only the terrain then the differences in HPT are reflected at the O-D level.

Thus when we generate our "history" if we generate patterns where the HPT information is a function of all the attributes at the O-D-T level the numerical patterns of interest are represented only at the O-D-T (complete information) and O-D level (loss of information). In the former each observation statistic refers to an O-D-T level and in the latter each observation statistic is at the O-D level. Thus in our methodology of capturing incomplete information contained in the real-world using numerical patterns we represent the numerical patterns at the same level or a higher level of aggregation than the level at which we generated the HPT data. This approach is encapsulated in the table 1. The quantity that we observe from our history and represented as numerical patterns is the total horsepower assigned to a train given by $\sum_{d \in \mathcal{D}_a^t} d_{HP} x_{ad}^{h*}$. We derive the order statistics for these quantities based on the

Data behavior	Numerical Patterns		
	O-D-T-ID	O-D-T	O-D
O-D-T-ID	Yes	Yes	Yes
O-D-T	-	Yes	Yes
O-D	-	-	Yes

Table 1: Experimental Scenarios

SO methodology for different data scenarios for different levels of aggregation			
Data behavior	(Improvement ratio,Scaling factor,Iterations)		
	O-D-T-ID	O-D-T	O-D
O-D-T-ID	(0.957,5000,1)	(0.178,50,2)	(0.131,50,11)
O-D-T	-	(0.940,5000,1)	(0.713,500,5)
O-D	-	-	(0.876,5000,1)

Table 2: Imperfect Engineering Costs

set of observation statistics which are generated at the same or higher level of aggregation with respect to level at which the HPT data is generated.

6.3 Summary of Results

All the models solved are quadratic programming problems and are solved on a SunOS5.8 platform using LOQO which utilizes interior-point methods to solve linear and quadratic programming problems (Vanderbei (1999)). We apply the SO methodology in (13), given an initial solution, in our experiment. Each model solved involved 95,109 variables and 12,077 constraints. In our experiment testing the SO methodology, convergence is achieved if the iteration-specific pattern metric given by:

$$H^m = \sum_{s\in\mathcal{S}}\sum_{k=1}^{N_s}\left(\hat{q}^m_{s\phi(k)} - \hat{q}^{h*}_{s(k)}\right)^2, \ \ m \in \{1,2,\ldots\}$$

where $\phi \in \Phi$ is the permutation vector that defines the order statistics of the solution vector $\{\hat{q}^m_{sk}\}_{s\in\mathcal{S},k\in\{1,2,\ldots,N_s\}}$, converges to within two significant digits.

We present the results of our experimental scenarios in table 2. We also performed an experiment where we assume the model is able to capture the engineering costs reflected in $\tilde{C}$ perfectly, that is, $c^a = \tilde{c}^a$ and $c^r = \tilde{c}^r$ and the only difference between C and $\tilde{C}$ is the

SO methodology for different data scenarios for different levels of aggregation			
Data behavior	(Improvement ratio,Scaling factor,Iterations)		
	O-D-T-ID	O-D-T	O-D
O-D-T-ID	(1.00,10000,1)	(0.144,50,1)	(0.126,50,12)
O-D-T	-	(1.00,10000,1)	(0.808,1000,6)
O-D	-	-	(0.981,5000,2)

Table 3: Perfect Engineering Costs

unknown vector of beta multipliers $\delta^{\beta} = \{\delta_t^{\beta}\}_{a \in \mathcal{A}^t}$. These results are shown in table 3. We show in the tables the best improvement ratio and the scaling factor corresponding to this ratio and number of iterations to convergence of the SO algorithm for this scaling factor.

As is indicated by the improvement ratio of 1 in the case where we use perfect engineering costs the SO methodology is able to capture all the incomplete information completely when θ is allowed to increase indefinitely and there is no aggregation and numerical patterns are generated at states O-D-T-ID and O-D-T as is indicated by the improvement ratio of 1. In the case where numerical patterns are generated at O-D level and represented with no aggregation along with perfect engineering costs the maximum improvement ratio is 0.981. It should be noted that since the SO methodology is not guaranteed to converge to the global optimum, these results speak well of the quality of the initial starting solution. These results show that the SO methodology as applied with our choice of initial feasible solution is successful in its ability to capture the combinatorial effects of the order statistics.

In figure 2 we represent the improvement ratio as a function of different data scenarios and the numerical patterns are represented at the O-D level. In the case of perfect engineering costs we plot this in figure 3. We see that the improvement ratio is not a monotone in the scaling factor as is indicated by its maximum at $\theta = 50$ when the numerical patterns are generated at the O-D-T-ID level. As is expected the improvement ratio at a particular value of θ is greater if the loss of information in pattern representation is less.

We plot the pattern metric as a function of the scaling factor in figure 4. We see that as $\theta \to \infty$ the pattern metric converges to 0 indicating that by increasing the scaling factor indefinitely we are able to put more emphasis on the historical patterns. In our experiment

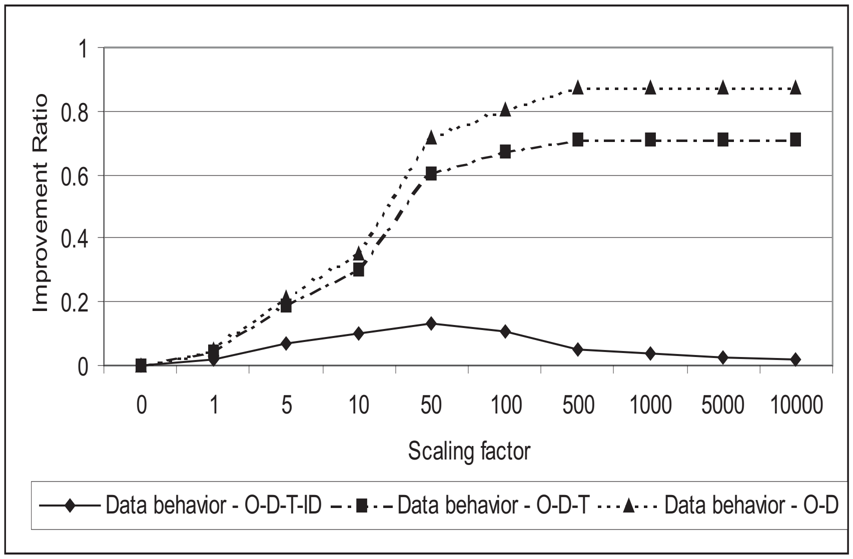

Figure 2: Improvement ratio where numerical patterns are represented at O-D level: Imperfect engineering costs

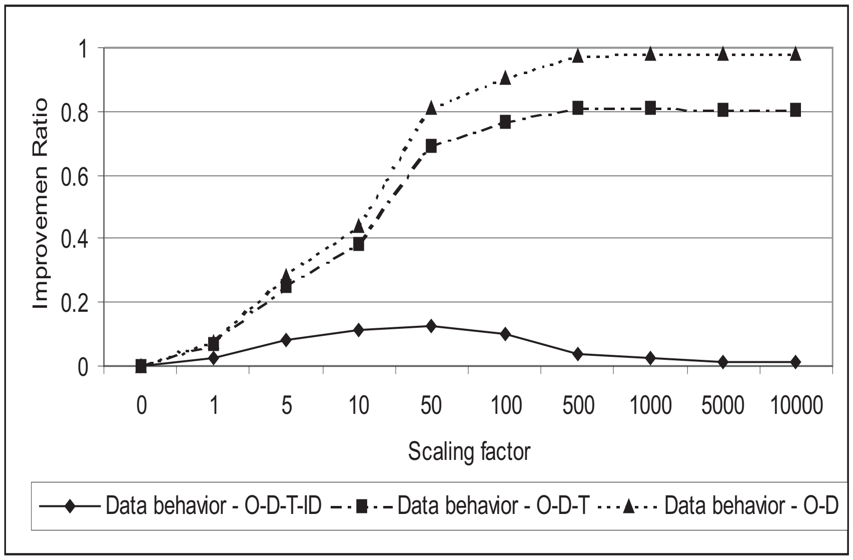

Figure 3: Improvement ratio where numerical patterns are represented at O-D level: Perfect engineering costs

we found that at a value of $\theta = 500$ the reduction in the pattern metric with respect to its value at $\theta = 1$ is around 99.9%.

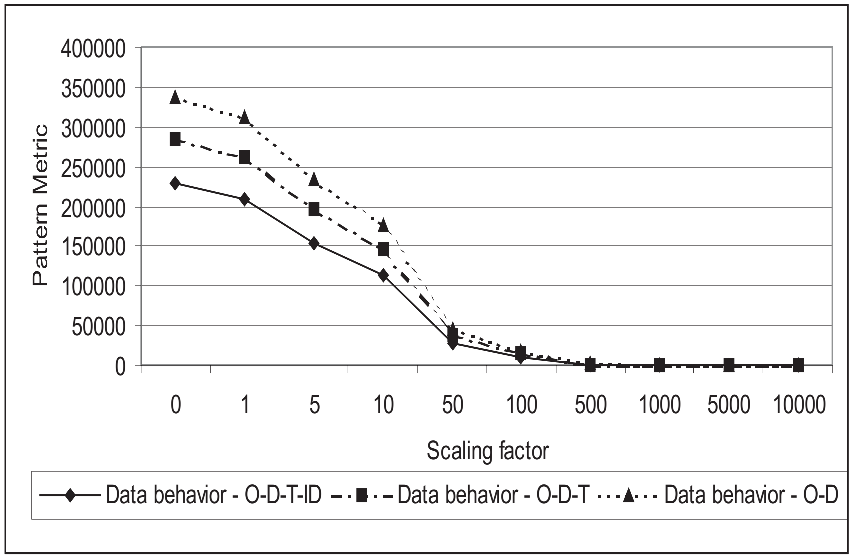

Figure 4: Pattern metric where numerical patterns are represented at O-D level: Imperfect engineering costs

7 Conclusion

We present a new methodology for representing information from regression trees in resource allocation models. Regression trees capture information from a historical database pertaining to flows or numerical quantities. Such problems arise in a freight logistics operation, for instance, representing information regarding locomotive power assigned to a train in a locomotive management operation.

We see that the problem of representing information pertaining to observed flows or quantities in a historical database is closely related to the empirical distribution function (EDF) of the decision variables. We adopt the Cramer-Von Mises goodness-of-fit test statistic for continuous distribution to develop our methodology of representing information in resource allocation models because of its separable nature in the number of instances of the observation statistic. The functional form known as the pattern metric that we use to represent information is combinatorial in nature due to the influence of the order statistics of the quantities. We solve this global optimization problem using the *sequential ordering* (SO) method that solves a sequence of models without any explicit use of constraints specifying the order statistics. We prove convergence of this algorithm to a stationary point in a finite number of iterations.

We tested our methodology on a real-world data set representing the operations of a large freight railroad. Preliminary results show that the SO algorithm has potential in representing information pertaining to numerical patterns as is validated by the positive improvement ratios for different levels of aggregation at which the numerical patterns are represented in the pattern metric. We also see that the performance of our methodology improves if we represent the numerical patterns with more detail in the pattern metric. A suggestion for future work is the application of our research in a real-world optimization model such as a locomotive scheduling model for a large freight railroad.

Acknowledgment

This research is supported by a consortium of three major railroads in United States and Canada, Norfolk Southern Corporation, Burlington Northern Santa Fe Corporation and Canadian National Railroad who are all corporate sponsors of CASTLE Laboratory, Department of Operations Research and Financial Engineering, Princeton University.

Appendix

The section relates to the content in section 5 of this paper. We state and prove the following lemma:

Lemma 7.1 *Given a vector* $\{x_k\}_{k\in\{1,2,\ldots,n\}}$ *whose elements are all strictly positive and a set* Φ_n *of all possible permutations of the vector* $\{1,2,\ldots,n\}$*, if we have a new vector of order statistics* $\{y_{(k)}\}_{k\in\{1,2,\ldots,n\}}$ *whose elements are also strictly positive, then:*

$$\max_{\phi\in\Phi_n}\sum_{k=1}^{n} y_{(k)}x_{\phi(k)} \quad = \quad \sum_{k=1}^{n} y_{(k)}x_{(k)}$$

where $\phi(k)$ *is the* k^{th} *element of the permutation vector* $\phi \in \Phi_n$ *and* $\{x_{(k)}\}_{k\in\{1,2,\ldots,n\}}$ *represents the vector of ordered statistics for* $\{x_k\}_{k\in\{1,2,\ldots,n\}}$.

Proof. Let $\phi \in \Phi_n$ be a generic permutation vector of $\{1, 2, \ldots, n\}$. We show our proof by induction. For the case where $m = 1$ the statement in the lemma is trivial. Consider the case where the number of instances is 2. In this case there are only two permutations possible for the vector x, $\{x_1, x_2\}$ and $\{x_2, x_1\}$. Without loss of generality we can set the vector of order statistics $\{x_{(1)}, x_{(2)}\}$ to $\{x_1, x_2\}$. We have:

$$\begin{aligned}(y_{(1)}x_{(1)} + y_{(2)}x_{(2)}) - (y_{(1)}x_2 + y_{(2)}x_1) &= y_{(1)}x_1 + y_{(2)}x_2 - y_{(1)}x_2 - y_{(2)}x_1 \\ &= y_{(1)}(x_1 - x_2) - y_{(2)}(x_1 - x_2) \\ &= (y_{(1)} - y_{(2)})(x_1 - x_2) \\ &\geq 0\end{aligned}$$

since $(y_{(1)} - y_{(2)}) \leq 0$ by definition of order statistics and $(x_1 - x_2) \leq 0$ by our assumption. Thus the lemma is true for the case $m = 2$, that is, we have:

$$\max_{\phi \in \Phi_2} \sum_{k=1}^{2} y_{(k)}x_{\phi(k)} = \sum_{k=1}^{2} y_{(k)}x_{(k)}$$

Thus we have shown that lemma is true for $m = 1, 2$. Using the induction hypothesis we assume that the lemma is true for $m = n$, that is,

$$\max_{\phi \in \Phi_n} \sum_{k=1}^{n} y_{(k)}x_{\phi(k)} = \sum_{k=1}^{n} y_{(k)}x_{(k)}$$

Consider the case $m = n + 1$. Without loss of generality we assume $y_{n+1} \geq y_j, \forall j \in \{1, 2, \ldots, n\}$ and $x_{n+1} \geq x_j, \forall j \in \{1, 2, \ldots, n\}$ so that $x_{(n+1)} = x_{n+1}$ and $y_{(n+1)} = y_{n+1}$. Consider the following quantity:

$$Z_{n+1}(\phi) = \sum_{k=1}^{n+1} y_{(k)}x_{\phi(k)}, \quad \forall \phi \in \Phi_{n+1}$$

We define the following function:

$$Z_{n+1,1}(\phi) = \sum_{k=1}^{n} y_{(k)}x_{\phi(k)} + y_{(n+1)}x_{(n+1)}, \quad \forall \phi \in \Phi_n \tag{22}$$

Consider swapping x_{n+1} with any $x_j, j \in \{1, 2, \ldots, n\}$ in equation (22). This procedure can be expressed using a function as shown below:

$$Z_{n+1,2}(\phi, j) = (\sum_{k=1}^{n} y_{(k)} x_{\phi(k)}) + y_{(n+1)} x_{\phi(j)} - y_{(j)} x_{\phi(j)} + y_{(j)} x_{(n+1)}, \quad \forall j \in \{1, 2, \ldots, n\}, \ \forall \phi \in \Phi_n$$

$Z_{n+1,2}(\phi, j)$ evaluates Z_{n+1} for all possible permutations of the set Φ_{n+1} such that Z_{n+1} does not contain the term $y_{(n+1)} x_{(n+1)}$. $Z_{n+1,1}(\phi)$ evaluates Z_{n+1} for all possible permutations of the set Φ_{n+1} such that Z_{n+1} consists of the term $y_{(n+1)} x_{(n+1)}$. We have:

$$\begin{aligned} Z_{n+1,1}(\phi) - Z_{n+1,2}(\phi, j) &= y_{(n+1)}(x_{(n+1)} - x_{\phi(j)}) - y_{(j)}(x_{(n+1)} - x_{\phi(j)}) \\ &= (y_{(n+1)} - y_{(j)})(x_{(n+1)} - x_{\phi(j)}) \\ &\geq 0, \qquad \forall j \in \{1, 2, \ldots, n\}, \ \forall \phi \in \Phi_n \end{aligned}$$

by definition of the order statistics. From the above equation we have:

$$\begin{aligned} \max_{\phi \in \Phi_{n+1}} Z_{n+1}(\phi) &= \max_{\phi \in \Phi_n} \{Z_{n+1,1}(\phi), \max_{j=\{1,2,\ldots,n\}} Z_{n+1,2}(\phi, j)\} \\ &= \max_{\phi \in \Phi_n} Z_{n+1,1}(\phi) \\ &= \max_{\phi \in \Phi_n} (\sum_{k=1}^{n} y_{(k)} x_{\phi(k)}) + y_{(n+1)} x_{(n+1)} \\ &= \sum_{k=1}^{n+1} y_{(n+1)} x_{(n+1)} \end{aligned}$$

Thus the lemma is true for $m = n + 1$ and by the induction hypothesis it is true for all $m = n$ where $n \in \{1, 2, \ldots\}$.

Q.E.D.

References

Bazaraa, M. S., Sherali, H. D. & Shetty, C. M. (1993), *Nonlinear Programming: Theory and Algorithms*, second edn, John Wiley & Sons, Inc., New York. 16

Birge, J. (1997), 'Stochastic programming computation and applications:', *INFORMS Journal on Computing* **9**(2), 111–133. 2

Bitran, G., Chandru, V., Sempolinski, D. & Shapiro, J. (1981), 'Inverse optimization: An application to the capacitated plant location problem', *Management Science* **27**(10), 1120–1141. 3

Breiman, L., Friedman, J., Olshen, R. & Stone, C. (1984), *Classification and Regression Trees*, Wadsworth International Group, Belmont, CA. 5, 6

Burton, D. & Toint, P. (1994), 'On the use of an inverse shortest paths algorithm for recovering linearly correlated costs', *Mathematical Programming* **63**, 1–22. 3

Burton, D., Pulleybank, B. & Toint, P. (1997), The inverse shortest paths problem with upper bounds on shortest paths costs, *in* P. Pardolos, D.W.Hearn & W.H.Hager, eds, 'Network Optimization', Vol. 450, pp. 156–171. 3

Chakravarthi, I., Laha, R. & Roy, J. (1967), *Handbook of Methods of Applied Statistics: Volume I*, John Wiley and Sons, New York. 11

Devroye, L., Gyorfi, L. & Lugosi, G. (1996), *A Probabilistic Theory of Pattern Recognition*, Springer. 4

Fayyad, U., Piatetsky-Shapiro, G. & Smyth, P. (1996), From data mining to knowledge discovery: An overview, *in* U. Fayyad, G. Piatetsky-Shapiro, P. Smyth & R. Uthurusamy, eds, 'Advances in Knowledge Discovery and Data Mining', AAAI Press, Menlo Park, CA, pp. 1–34. 4

Frank, M. & Wolfe, P. (1956), 'An algorithm for quadratic programming', *Naval Research Logistics Quarterly* **3**, 95–110. 16

Glover, F. (1990), 'Tabu search, part II', *ORSA J. Computing* **2**, 4–32. 16

Holland, J. (1975), *Adaptations in Natural and Artificial Systems*, Addison-Wesley Publishing Company. 16

Kirkpatrick, S., Gelatt, C. & Vecchi, M. (1983), 'Optimization by simulated annealing', *Science* **220**(4598), 671–680. 16

Morgan, J. & Sonquist, J. (1963), 'Problems in the analysis of survey data and a proposal', *J. Amer. Statist. Assoc.* **58**, 415–434. 6

Nilsson, N. (1990), *The Mathematical Foundations of Learning Machines*, Morgan Kaufmann, San Mateo, CA. 4

Pardalos, P. (1989), 'Parallel search algorithm in global optimization', *Applied Mathematics and Computation* **29**, 219–229. 16

Powell, W. B. & Carvalho, T. A. (1997), 'Dynamic control of multicommodity fleet management problems', *European Journal Of Operations Research* **98**, 522–541. 12

Quinlan, J. (1993), *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA. 5

Read, T. & Cressie, N. (1988), *Goodness-of-fit statistics for discrete, multivariate data*, Springer-Verlag, New York. 10

Sen, S. & Higle, J. (1999), 'An introductory tutorial on stochastic linear programming models', *Interfaces* **29**(2), 33–61. 2

Sokkalingam, P., Ahuja, R. & Orlin, J. (1999), 'Solving inverse spanning tree problems through network flow techniques', *Operations Research* **47**(2), 291–298. 3

Vanderbei, R. (1999), 'An interior point code for quadratic programming', *Optimization Methods and Software* **11**(2), 451–484. 28

Zhang, J. & Liu, Z. (1999), 'A further study on inverse linear programming problems', *Journal of Computational and Applied Mathematics* **106**(2), 345–359. 3