Coalescing Data and Decision Sciences for Analytics

Yunxiao Deng
Daniel J. Epstein Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, 90089, yunxiaod@usc.edu

Junyi Liu
Daniel J. Epstein Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, 90089, junyiliu@usc.edu

Suvrajeet Sen
Daniel J. Epstein Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, 90089, s.sen@usc.edu

Abstract The dream of analytics is to work from common, clean, and consistent data sources in a manner that all of its facets (descriptive, predictive, and prescriptive) are supported via a coherent vision of data and decision sciences. To the extent that data and decisions sciences work within logically/mathematically consistent frameworks, and that these paradigms operate within computationally realistic structures, analytics will help organizations thrive. Done right, analytics promises to plant more OR/MS resources at C-suites where major corporate decisions are made. Done poorly, the OR/MS community will look back at these days as a lost opportunity to make a long-term difference to both theory and practice of analytics. With this tutorial, we invite the OR/MS community to join an intellectually vibrant endeavor to integrate seemingly disparate pillars, namely predictive and prescriptive analytics. Just as importantly, we show how OR/MS strengths in modeling, algorithms, and applications can facilitate coalescing the data and decision sciences. We draw upon several examples which pair data science tools such as filtering and regression, to decision science tools such as Monte Carlo tree search and stochastic programming. The marriage of these two paradigms (data and decision sciences) is what we refer to as Learning Enabled Optimization (LEO). This tutorial covers fundamental concepts for the fusion of statistical and optimization modeling, statistically approximate optimality, sampling-based algorithms, and finally, model assessment and selection.

Keywords Statistical Learning, Optimization, Model Assessment and Selection

1. An Integrated View of Analytics

In a recent survey (KPMG 2014), over one hundred CFOs and CIOs of large organizations (over a billion dollars in annual turnover) were interviewed regarding the effectiveness of business analytics. Over 96% of those surveyed acknowledged that they could do better with big data, and make better use of analytics. Two of the top three significant questions which emerged from the KPMG survey relate to: a) How will predictive analytics drive future decision making? and b) What technology will be required to operationalize data and analytics within the organization? From an OR/MS perspective, these questions ask: a) what "thought- leadership" will we provide so that we can build a bridge between predictive and prescriptive analytics? and b) what technological leadership will we provide so as to "operationalize data and analytics"? Motivated by these two questions, this tutorial coalesces concepts from data and decision sciences to help identify innovations to address these questions.

A class of applications which come under the umbrella of Analytics of Things (Tom Davenport, Deloitte Insights, 2014) builds upon the "sense-and-respond" paradigm in which the Internet is used to communicate data for the purposes of Analytics. However, transforming insight into action is not as straightforward as one may be led to believe. To recognize the hurdles, consider a common view of analytics, as summarized by Figure 1 (source: Gartner Analytics). While the mapping of different forms of analytics to "hindsight, insight, foresight", is itself very insightful, it also highlights the challenges of integration. Because of disparate managerial levels, data types, sources, and frequency of updates, as well as disparate mathematical assumptions, the specific areas identified in Figure 1 tend to operate within their own silos. This tutorial is intended to help readers bridge the silos at the higher end of the value-skills spectrum. The fundamentals of data science draw upon statistical (or

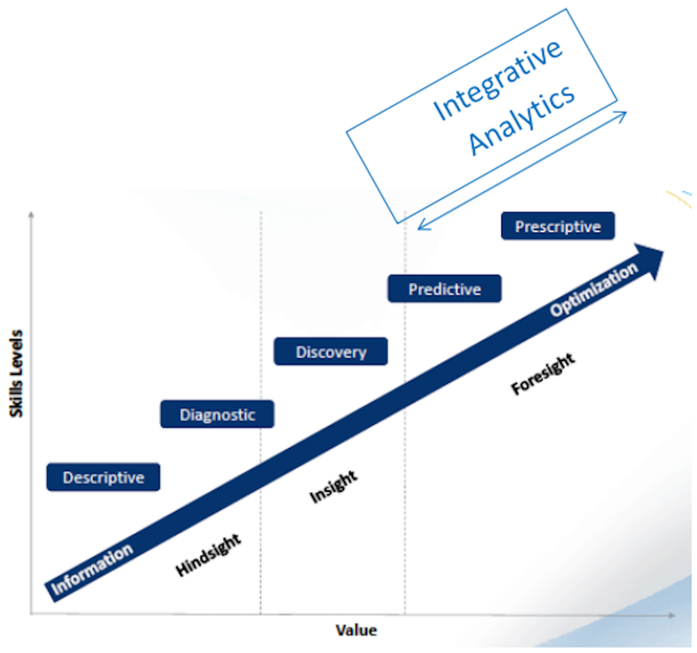


FIGURE 1. Types/Phases of Analytics. Source: Gartner Analytics. Integrative Analytics box added by the authors.

machine) learning (SL/ML) and optimization. In addition, there is a wealth of data that is being collected, curated and disseminated via the world-wide-web. If this growing deluge of data can be harnessed by powerful decision models, then the OR/MS community will not only provide answers to questions raised in the KPMG report, it will also be legitimate in laying claims to thought-leadership in this emerging area. One of the goals of this tutorial is to help researchers recognize the potential for exploiting the interdependence between data and decisions. While researchers in both these areas are making important strides within each silo, there are many opportunities for advances in the interstices of data and decision sciences. From modeling to algorithmic frameworks, and model validation to model selection, there are a host of open questions that remain unresolved.

Some of the early work in the area started with research on operations management (OM) models with special structures, such as newsvendor problems. For instance, Liyanage and Shanthikumar [36] and more recently Rudin and Vahn [41] studied the integration of optimization and learning to identify optimal inventory ordering policies. Both papers use the specialized form of the newsvendor model to illustrate the potential for joint models of learning and optimization (see also Cooper et al. [10]). Other OM applications motivating joint estimation and optimization models consider problems arising from combining learning and pricing (Besbes and Zeevi [7],[Cooper et al. [10], Harsha et al.[20]). Another avenue of application-specific use of learning within optimization arises from the introduction of specialized regularization in the context of portfolio optimization in Ban et al. [2]. Finally, for a broad personal perspective on data-driven OM, we refer to Simchi-Levi[48].

As for algorithmic frameworks which tie data and decision sciences, an early paper is the work on Directed Regression (Kao et al. [29]). In this setup, an empirical optimization method was developed in which regression coefficients are chosen based on a combined objective consisting of an empirical loss function associated with the solution chosen based on a quadratic cost function with the regression coefficients. Based on a generative model with missing features, it was shown that the regression coefficients for minimizing the risk of quadratic cost function is a convex combination of ordinary least square estimation and coefficient estimation by empirical optimization method. Subsequently, Kao and Van Roy [30] extended these ideas to another quadratic optimization model whose optimum solution depends on an unknown Variance-Covariance Matrix of a Gaussian random variable. A more recent study (Davarnia and Cornuéjols [11]) works on parameter estimation for optimization via shrinkage technique. In an optimization problem with the unknown parameters, they construct a solution using the maximum likelihood estimation (MLE) with shrinkage as the parameter estimation and shows that such construction dominates the MLE solution estimator for a quadratic cost function. Another viewpoint at the interface between predictive and prescriptive analytics is presented in Bertsimas and Kallus [6]. Their work demonstrates that in the presence of dependencies among random variables, approaches which accommodate learning perform much better than those which do not (e.g., optimization with sample average approximation).

Despite much interest in the two fundamental building blocks, i.e., statistics and optimization, there are gaps in the fundamental questions which these areas serve. In general, the algorithmic aspects of both building blocks derive from common variational principles (see Rockafellar and Uryasiev [39]). However, there is a fair amount of separation between other aspects such as modeling, model validation, and model selection. In the world of SL/ML, objectives are typically stated in the form of "goodness of fit". This leads to a specific form of optimization objective. For instance, a lower bound of zero is often valid for optimization models arising in SL/ML and one can take advantage of such a priori knowledge within optimization. On the other hand, model assessment and selection are not as common in the optimization literature. Recent exceptions to this comment are the works of Den Boer and Sierag [13] and Sen and Deng[43]. The first of these formalizes the notion of decision-based-model-selection (DBMS), whereas, the latter provides an "end-to-end" covering modeling, algorithms, model assessment and selection. Because of its "end-to-end" emphasis, it is convenient to include a description of DBMS within the LEO framework (see section 4).

The main sections of this tutorial pertain to three classes of decision models:

- Section 2: Deterministic optimization models for which cost parameters are estimated via predictive analytics. To date, most practical interfaces between data and decision sciences have been studied via a sequential process, sometimes referred to as "Predict-then-Optimize" (PO) (Elmachtoub and Grigas [15]). We begin our presentation using this basic idea for two examples which highlight how modern data sources have altered the landscape of decisions associated with some traditional questions.
- Section 3: Stochastic optimization models in which cost as well as constraint coefficients (including right-hand side parameters) are modeled using random variables. While this approach falls under traditional stochastic programming (SP) (Birge and Louveaux [8]), we will address certain non-traditional aspects such as model validation. These models are also viewed in the PO setting with the emphasis on model validation which highlights the impact of data science on SP. We refer to Wallace and Ziemba [50] for a more comprehensive collection of applications.
- Section 4: Learning enabled optimization (LEO) models in which statistical learning is used to capture dependencies resulting from *latent* random effects. The mathematical structure of LEO models derives from a fusion of statistical/machine learning (SL/ML) and SP. Again, we discuss procedures for validating such models, as well as issues related to model selection. Such issues, which are somewhat novel for the optimization literature, and are motivated by connections with data science.

This tutorial also provides links to data sources which may be used for the pedagogical examples discussed below.

2. Predict-then-Optimize with Deterministic Constraints: Uncertainty in the Objective

We begin with two applications for which the set of feasible decisions are deterministic, whereas, the objective function for the models are not known precisely. The first example is a thoroughly revamped version of the famous Diet Problem (DP), and the second one is the ubiquitous Dynamic Routing Problem (DRP). In the first model (DP, see subsection 2.1), the objective function represents an ambiguous quantity i.e., ratings. As the model is exercised many times, by many individuals, the errors of prediction reduce, and hence the description of such a learning system is "Collaborative Filtering". For each round in this setting, the objective function is observed once, and no other prediction is made until the decision model has been solved. Thus, it is a static estimate of the objective function. In case of DRP (see subsection 2.2), the objective function changes during the decision process, as updated sensor data becomes available. The updates for DRP are based on "Particle Filtering" and only a subset of decisions are implemented, while changes to other (future) decisions are possible as the cost vector evolves. In both cases, the data science aspect deals with the concept of "filtering", which refers to the act of prediction with reduced errors as larger amounts of data become available to the prediction method.

After presenting the above examples, we turn to some very recent theory in subsection 2.3 where we summarize an approach known as Smart "Predict then Optimize" (SPO) [15]. This concept focuses on convex optimization problems with a linear objective over a compact set. In this sense, it is more general than many previous ideas in the literature (i.e., either special-purpose models such as newsvendors, or unconstrained optimization models). The SPO setup measures the loss of optimality from implementing an incorrect decision induced by an error-prone predicted cost function. As of this writing however, the SPO approach is not yet general enough to accommodate either mixed-integer programming (as in DP) or sequential predictions with only part of the decision vector being implemented in each round (as in DRP). Nevertheless, we include SPO in this tutorial to motivate future work which might address more general models.

2.1. Static Predict-then-Optimize

Example 1 (Meal Planning for the New Millennium (MnM Problem)).

The famous Diet Problem was first stated by George Stigler (Stigler(1945) [49]) who also used a heuristic to find an approximate solution for a data set used in the 1930s and 40s. While there is continued pedagogical value of the Diet Problem for LP formulations, there has always been the criticism that in the absence of a reasonable objective function, a diet based on minimizing cost or a similar objective may not provide particularly tasty meals. Moreover, tasty meals result from recipes which are targeted to individual tastes. Thanks to Internet-based feedback for recipes, it is now possible to formulate such optimization models by using ratings on various recipes available on the web[1]. These recipes not only provide the specific quantities of various foods and the required ingredients, they also tend to provide nutrition content, cooking time estimates etc. Note that one can also estimate costs of meals from online groceries (e.g., Walmart or even the neighborhood groceries). Finally, note that modern python-based programs[2] can be used to automatically download content using web-scraping programs. As a result, data for the MnM Problem no longer reside in library archives (as they did in the days of G. Stigler).

Consider two roommates who are planning their meals for the following week, and they are friendly enough to share cooking and shopping responsibilities but like many students, they are strapped for time and money. These roommates have well-trained taste buds, and are interested in eating healthy, tasty meals, as well as eating within budgetary and dietary restrictions. The MnM problem focuses on meal planning with an OR/MS twist: the roommates will share shopping, cooking, and meal selection responsibilities, and will also respect each other's schedules and budgets. They will use social media (and websites) to choose recipes. This shift from buying food-types in the Diet Problem to choosing recipes for the MnM results in the shift of decision modeling paradigm from LP to mixed-integer programming (MIP), including mixed-binary variables. We omit presenting the MIP model for a couple of reasons: a) those familiar with a first course in OR/MS should be able to formulate such a decision model without much trouble, and b) we do not wish to curb the creativity of students in OR/MS courses since they might find this effort to be intellectually interesting. Nevertheless, the "proof is in the pudding", and as a result, we feel obliged to report that students in our Analytics course at USC have created week-long dinner menus which include "Grilled brined chicken with chimichurri sauce" on Monday, "Beef tenderloin with red wine sauce, creamed spinach and french fries" on Tuesday, ..., "Country bread stuffing with smoked ham, goat cheese and dried cherries" for Friday. Not only do these recommendations fit the roommates' tastes, the students also seemed to be satisfied with their cooking schedules which met their time and budget constraints. Such meal plans are a "far cry" from the traditional diet problem in which certain food-types, especially broccoli, appear to be favorites! With this motivation, we proceed to discuss the data science aspects for this application.

Data Science: Collaborative Filtering.

As for the data science side, one of its more interesting contributions in the meal-planning effort is the ability to include diet preferences. Although it may not be easy to get personalized ratings in the early weeks of training a meal planning model, one might be able to eventually infer personal ratings by using modern tools such as matrix completion or other

[1] https://www.kaggle.com/hugodarwood/epirecipes/data

[2] https://medium.freecodecamp.org/how-to-scrape-websites-with-python-and-beautifulsoup-5946935d93fe

collaborative filtering techniques. These data science tools are commonly used in recommendation systems by online retailers (e.g., Amazon, Netflix etc.) to recommend products and services inferred from prior individual ratings, as well as overlapping tastes with other users. Note that personalization only works after several trials have been undertaken to populate a ratings matrix with some user data. Nevertheless, one may use average ratings as a surrogate objective function before sufficient personal data becomes available.

The goal of matrix completion is to provide item recommendations or "predictions" based on the opinions of other users with similar "tastes" (Koren et al.(2009) [32]). Based on the input data of customers and their interested products, one can generate personalized coefficients for a certain user and then use the set of coefficients to predict the user's taste according to which we can recommend products with higher ratings. Specifically, suppose we are given a rating matrix M in which each entry (i,j) represents the rating of product j by user i if user i has rated product j. Otherwise, the entry is missing. Since we would like to predict the taste of users based on the given rating matrix, the problem is converted into predicting the remaining missing entries of M in order to make good recommendations to the users.

Note that matrix completion usually assumes that a low-rank matrix is preferred, since over-fitting may result if there are no restrictions placed on the number of degrees of freedom. (Therefore, we assume the rank of completed matrix is given, say r, and we would like to find a matrix $\hat{M}$ with rank r which matches the known entries.) Based on this idea, one formulates the following mathematical model.

$$\min_X \quad \frac{1}{2}||P_\Omega(M-X)||_F^2 + \lambda||X||_* \tag{1}$$

where Ω represents the positions of entries which are already known in the rating matrix M and $P_\Omega(\cdot)$ then represents the orthogonal projection onto the span of matrices vanishing outside of Ω, so that the (i,j)th component of $P_\Omega(X)$ is equal to X_{ij} if $(i,j)\in\Omega$ and zero otherwise. Let $\|\cdot\|_F$ denote the Frobenius norm which is the sum of square of all entries and $\|\cdot\|_*$ denote the nuclear norm, which is the sum of the singular values of the matrix. Such a norm provides a convex relaxation of rank in (1). Therefore we obtain a convex-optimization problem for matrix completion of M. In addition, there exist cases where the entries are provided with a small amount of noise. For example, a noisy model can have $Y_{ij} = M_{ij} + \varepsilon_{ij}$ where ε_{ij} denotes a noise term and Y_{ij} is the observed rating entry. From Hastie et al.(2015) [21], we could summarize the algorithm of completing M as follows.

At iteration i,

(1) The estimated matrix at iteration i is denoted as $\hat{X}_i$. Calculate

$$\hat{M} = P_\Omega(M) + P_\Omega(\hat{X}_i).$$

(2) Compute the soft-thresholded singular value decomposition of $\hat{M}$ represented as $UDV^\top$.
(3) Construct a diagonal matrix $\hat{D}_\lambda$ based on D. If $D_{ii} > \lambda$, let $(\hat{D}_\lambda)_{ii} = (D_{ii} - \lambda)$. Otherwise, $(\hat{D}_\lambda)_{ii} = D_{ii}$.
(4) Update $\hat{X}_{i+1}$ by $U\hat{D}_\lambda V^\top$.
(5) Repeat from step 1 to step 5 until convergence.

2.2. Evolving Predict-then-Optimize

Unlike the previous example, sensor data updates speed predictions, which then requires updating the route (i.e the decision) itself. Even though the constraints in the optimization model remain deterministic, the fact that only part of the decisions are implemented changes the way that data and decisions interact within this setting. Hence the decision problem in this example includes a "look-ahead" feature which was absent from the previous example.

Example 2 (Dynamic Routing Problem). Consider a situation where we are planning to find the best route (with least travel time) from node s to node t in a network, whose links are divided into several pieces by speed sensors (see Figure 2). For link i, there are n^i speed sensors located at $d_1, ..., d_{n^i}$, while sensors provide noise-corrupted data $z(d_1, \tau), ..., z(d_{n^i}, \tau)$ at time τ for all locations. We will use particle filtering to use these noise-corrupted observations to infer state variables estimates (denoted w) throughout the network. Suppose we plan to start traveling at time T_0 on a particular weekday. Traditional

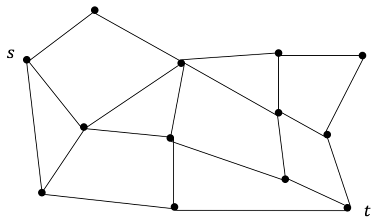


FIGURE 2. Network Routing

approaches might use historical speed data from previous weeks as a way to estimate travel time. Because of the availability of sensor data in today's traffic network, more current data on speeds can be inferred from the speed data. In this example we use filtering techniques as a prediction tool (Chen et al. [9]) for all sensor locations in Figure 2. As for the decision problem, we study a specialized shortest-path Monte Carlo Tree Search algorithm to find the best route.

Data Science: Particle Filtering. Perhaps, the most commonly used paradigm for sequential state estimation is the Kalman filter which is applicable in the context of linear dynamic systems subject to Gaussian noise. Since the setting of urban traffic flow is known to be nonlinear, and the noise is usually not Gaussian, we will work with a more modern filtering scheme known as the Particle Filter. This scheme, also known as a Sequential Monte Carlo (SMC) method, assumes that the state space of the dynamic system can be partitioned into many small segments over which the density of particles approximate the probability distribution of the state of the system. Together the ensemble of particles will represent the distribution of the state at the most recent (past) time instant. The method will use a Bayesian framework to obtain a posterior distribution which is predicted using past observations of the stochastic process. In this mechanism, each particle will be assigned its own likelihood (weight) which indicates the probability of being sampled. This framework of particle filters is illustrated in Figure 3.

To describe the specific update, suppose that the traffic states are estimated consecutively at discrete time instants $t_1, t_2, \ldots, t_k, \ldots$, possibly asynchronously, based on all the incoming information up to the current time T_0 transmitted by speed sensors to the filter. Let $\mathbf{w}_k$ denote the state vector of traffic speed at time t_k, and $\mathbf{z}_k$ be the observation vector of traffic speed at time t_k at sensor locations. Thus the observation vector $\mathbf{Z^k} = \{\mathbf{z}_1, \ldots, \mathbf{z}_k\}$, will be used to infer the state vector w_k (speed) for each link-segment of the network. This will be accomplished using a recursive Bayesian update in which the conditional density function $p(\mathbf{w}_k|\mathbf{Z}^k)$ of the state $\mathbf{w}_k$, given a set of measurements $\mathbf{Z}^k$, can be updated according to

$$p(\mathbf{w}_k|\mathbf{Z}^{k-1}) = \int p(\mathbf{w}_k|\mathbf{w}_{k-1})\ p(\mathbf{w}_{k-1}|\mathbf{Z}^{k-1}) d\mathbf{w}_{k-1} \tag{2}$$

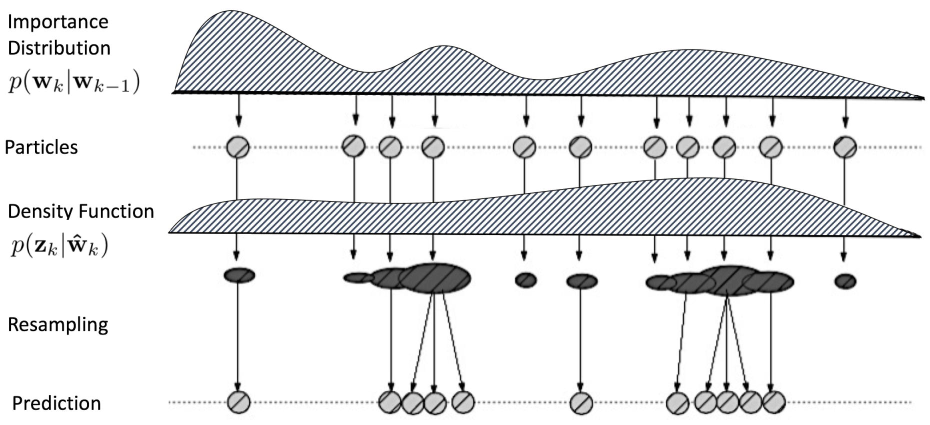

Importance Distribution $p(\mathbf{w}_k|\mathbf{w}_{k-1})$
Particles
Density Function $p(\mathbf{z}_k|\hat{\mathbf{w}}_k)$
Resampling
Prediction

FIGURE 3. Particle Filtering Steps

$$p(\mathbf{w}_k|\mathbf{Z}^k) = \frac{p(\mathbf{z}_k|\mathbf{w}_k)\ p(\mathbf{w}_k|\mathbf{Z}^{k-1})}{p(\mathbf{z}_k|\mathbf{Z}^{k-1})} \propto\ p(\mathbf{z}_k|\mathbf{w}_k)\ p(\mathbf{w}_k|\mathbf{Z}^{\mathbf{k-1}}) \tag{3}$$

However, it is computationally expensive to evaluate (2) and (3). Particle Filtering provides an approximation of the dynamic process by a discrete-time recursive update of the posterior probability distribution $p(\mathbf{w}_k|\mathbf{Z}^k)$ corresponding to a collection of particles $\{\mathbf{w}_k^{(i)}\}_{i=1}^N$. Specifically, in the main step of the algorithm (see step 2 in the framed summary below) at iteration k, by sampling the state space with respect to certain probabilities, we are able to get a set of particles $\{\mathbf{w}_{k-1}^{(i)}\}_{i=1}^N$ as an approximation of the evolving probability distribution function $p(\mathbf{w}_{k-1}|\mathbf{Z}^{k-1})$. However, the real posterior density model $p(\mathbf{w}_k|\mathbf{w}_{k-1})$ is usually unknown. Therefore it is common to choose an approximation density function of the true posterior, which is also known as an *importance distribution*. Then we sample the particles $\{\hat{\mathbf{w}}_{\mathbf{k}}^{(i)}\}_{i=1}^N$ using the importance distribution $p(\mathbf{w}_k|\mathbf{w}_{k-1})$ given the particles $\{\mathbf{w}_{k-1}^{(i)}\}_{i=1}^N$. The weights $\{\beta_k^{(i)}\}_{i=1}^{(N)}$ of the particles can thus be updated by the sampled results $p(\mathbf{z}_k|\hat{\mathbf{w}}_k^{(i)})$ with normalization. One could discard the particles with low normalized importance weights, and then resample a new set of particles $\{\mathbf{w}_k^{(i)}\}_{i=1}^N$ using updated weights. After sufficiently many iterations, only a few particles have non-zero importance weights and the weights will converge. A generic algorithm of particle filtering using transition prior density can be summarized as follows.

For steps $k = 0, 1, 2, ...$

(1) Initialization: for $i = 1, ..., N$, generate samples $\{\mathbf{w}_0^{(i)}\}_{i=1}^N$ from the initial probability distribution $p(\mathbf{w}_0)$. Set initial weights $\beta_0^i = \frac{1}{N}$ for $i = 1, \dots, N$.
(2) Importance Sampling: for the sample $i = 1, ..., N$, draw a set of particles $\hat{\mathbf{w}}_k^{(i)}$ from an importance distribution $p(\mathbf{w}_k|\mathbf{w}_{k-1}^{(i)})$.
(3) Weights Update: calculate the updated weights $\beta_k^i = p(\mathbf{z}_k|\hat{\mathbf{w}}_k^{(i)})$ for $i = 1, \dots, N$.
(4) Weights Normalization: calculate the normalized importance weights

$$\hat{\beta}_k^i = \frac{\beta_k^i}{\sum_{j=1}^N \beta_k^j}.$$

(5) Resampling: discard the particles with low normalized importance weights, then resample to generate N new particles from the set $\{\hat{\mathbf{w}}_k^{(i)}, 1 \le i \le N\}$ according to the importance weights $\{\hat{\beta}_k^i\}_{i=1}^N$.
(6) Repeat from step 2 to step 5 until the weights converge. $p(\mathbf{z}_k|\hat{\mathbf{w}}_k)$

Decision Science

For a given location d_j and an initial time τ, we are able to train a particle filter by past dataset, to predict the expected future speed $\{\mathbb{E}[v(d_j,\tau+\tau_0)],\mathbb{E}[v(d_j,\tau+2\tau_0)],...,\mathbb{E}[v(d_j,\tau+m\tau_0)],\}$, where the cycle τ_0 is decided by the past dataset, and m denotes the number of prediction period. Once the predicted speed values are obtained, we are able to calculate and update travel time of each link. Note that in this example the travel time of each link is time-varying, because the speed prediction values are changing over time. As a result, to obtain the route with shortest travel time, we consider using an algorithm called Monte Carlo Tree Search (MCTS), in which the travel time of each link is updated during each time period, hence the algorithm becomes more responsive to the changing traffic conditions (Bertsimas et al.(2014) [5], Jiang et al.(2017) [28]). In the context of SPO, the decision vector evolves with each prediction round of the particle filter.

Monte Carlo Tree Search (MCTS) is a search algorithm, which has already made an impact on Artificial Intelligence (AI), and other cases in which the domain of the problem can be represented as sequence of decisions of trees (Fu(2017) [17]). For each iteration, we use a certain policy (tree policy) to find the most efficient node of the current tree. Then one can update the current tree by running a simulation from the selected node. The standard MCTS algorithm is summarized in the following.

For each iteration, there are mainly four steps.

(1) **Selection.** Starting from root node, select a child node by a given tree policy to reach an optimal node. Notice that in this case, the selected node is required to be expandable, which represents that the selected node has at least one unvisited child.
(2) **Expansion.** Since the selected node is expandable, add all the child nodes of the selected node to expand the tree.
(3) **Simulation.** For the newly added child nodes, run a simulation according to the given policy and calculate the outcome.
(4) **Backpropagation.** Based on the simulation result in step 3, backpropagate the current tree through the selected nodes, and update their values.

In this example, the routing algorithm is performed by using one-step ahead MCTS to find the current optimal route. Then the policy used for updating the statistics of newly created tree nodes is the predicted travel time by particle filtering. The shortest path implementation of MCTS is summarized as follows.

For time slots $j = 0, 1, 2, ...,$

(1) **Initialization.** Initialize the vertex set Q to include all the nodes. For each node v in the network, set $time[v]$ =Infinity, which represents a unknown travel time from source s to v in the initialization state. For the source node s, set $time[s] = 0$.
(2) **Selection.** Choose a node in Q with minimum distance from source s to it and denote the node as u. Remove node u from set Q.
(3) **Expansion.** For each neighbor v of u, update the current travel-time to be $time[u] + \tau_j(u,v)$, where $\tau_j(u,v)$ represents the predicted travel time from u to v in time slot j provided by particle filtering.
(4) **Simulation.** If the current travel-time $time[u] + \tau_j(u,v) < time[v]$, then we update the state of node v: set $time[v] = time[u] + \tau_j(u,v)$, and update the previous node of v in the current optimal path: $prev[v] = u$.
(5) **Backpropagation.** If Q is not empty, repeat from step 2 to step 4.

In this DRP experiment, the traffic travel time data was obtained from PeMS (the Caltrans Performance Measurement System)[3], in which the traffic data is collected in real-time using individual detectors. We run particle filtering as well as extended Dijkstra algorithm based on a network of the freeway-system across all major areas of Los Angeles. The predicted travel time of the optimal path using dynamic routing algorithm is 56.10 minutes, and we validate this result by using the real travel time data for the calculated optimal path. The validated travel time is 58.04 minutes, which is close to the predicted travel time. For the purposes of comparisons with a case of using the deterministic Dijkstra algorithm with the cost of each link being estimated by expected historical travel-time data, we find that the estimated travel time of an optimal path is 49.87 minutes, which is better than the result from dynamic routing algorithm. However, when we validate the calculated path by real travel time data, the resulting simulated time is 65.84 minutes, which indicates that there exists a mismatch of actual travel-time and estimated travel-time for deterministic routing. Moreover, comparing the validated results, the travel time of calculated path using particle filtering is significantly smaller.

2.3. Smart Predict-then-Optimize: Concepts

We now proceed to a recent generalization in which estimation and deterministic optimization are considered within one setting. In the meal planning example, the decision making problem is formulated as a mixed-integer program which has a cost vector $(r_1, \ldots, r_K)$ defined as the prediction of ratings of K recipes coming from the matrix completion model. Note that because of the prediction of ratings in the model, the objective function is somewhat uncertain. However, it would be appropriate to measure the loss of optimality because of deviation from presumed "true" vector. Unfortunately, this remains an open question for MIP models thus far. Nevertheless, we present a new approach known as Smart "Predict then Optimize" (SPO) as it is among the few approaches which do accommodate somewhat general optimization models, under assumptions of convexity and compactness of the feasible set [15]. Given these assumptions on a set $\mathbf{X}$, consider the following model.

$$P(w): \quad v^*(w) \triangleq \min_x \; w^\top x \quad \text{s.t. } x \in \mathbf{X} \tag{4}$$

where $x \in \mathbb{R}^{n_1}$ are decision variables, $w \in \mathbb{R}^{n_1}$ is the unknown uncertainty describing the linear objective function, and $\mathbf{X} \subseteq \mathbb{R}^{n_1}$ is a fixed, nonempty, compact and convex set. Suppose the uncertain vector w can be modeled as a dependent variable on an external feature vector $z \in \mathbb{R}^p$ from a hypothesis class $\mathcal{H}$ of functions $m : \mathbb{R}^p \to \mathbb{R}^{n_1}$. Given the loss function $\ell(\cdot,\cdot)$ and the training data $\{(z_i, w_i)\}_{i \in T}$ where T is the index set of training data set, we choose a prediction model $m^* \in \mathcal{H}$ by solving an empirical risk minimization problem.

$$\min_{m \in \mathcal{H}} \frac{1}{|T|} \sum_{i=1}^{|T|} \ell(m(z_i), w_i)$$

One common choice of the loss function is the mean squared error loss function, i.e. $\ell(\hat{w}, w) = \frac{1}{2} \parallel \hat{w} - w \parallel_2^2$. In the SPO approach of Elmachtoub and Grigas [15], a new loss function is constructed to measure the excess cost incurred due to an imprecise prediction of the cost vector. Let $x^*(w)$ be the optimal solution and $S^*(w)$ represent the optimal solution set of $P(w)$. The so-called SPO loss function is defined as,

$$\ell_{SPO}(\hat{w}, w) \triangleq \max_{x \in S^*(\hat{w})} \{w^\top x\} - v^*(w). \tag{5}$$

[3] https://pems.dot.ca.gov/.

Accordingly the empirical risk minimization problem for the prediction model can be constructed as,

$$\min_{m\in\mathcal{H}} \frac{1}{|T|}\sum_{i=1}^{|T|} \ell_{SPO}(m(z_i), w_i). \tag{6}$$

However, solving program (6) could be computationally challenging since $x^*(\hat{w})$ may not even be continuous and the loss function ℓ_{SPO} may not be convex in $\hat{w}$. Hence, a surrogate loss function $\ell_{SPO+}(\cdot,\cdot)$ is constructed as the upper bound of the true loss function $\ell_{SPO}(\cdot,\cdot)$.

$$\ell_{SPO+}(\hat{c}, c) \triangleq \max_{w\in S}\{c^\top w - 2\hat{c}^\top w\} + 2\hat{c}^\top w^*(c) - v^*(c).$$

According to the construction, the SPO+ loss function is convex in $\hat{c}$. It is then justified to be an approximation of ℓ_{SPO} (Proposition 4 and Proposition 5 in Elmachtoub and Grigas [15]). Based on the construction of the SPO+ loss function, there are some preliminary results on the consistency of the surrogate loss problem to the true loss problem. Since the empirical risk minimization of surrogate loss function is a convex problem, many state-of-the-art algorithms are available to solve that problem, such as stochastic gradient descent algorithm and the variance-reduction/acceleration algorithms. Specific algorithms should be considered when dealing with some special structures. For instance, when the hypothesis class is the class of linear functions and the feasible set is a polyhedron, the SPO+ loss problem can be reformulated as a linear program by using the dual formulation Elmachtoub and Grigas [15].

There are several works, in different settings, which combine parameter prediction with optimization in order to make better decisions. Donti et al [14], also consider a stochastic program with an uncertain cost vector in the objective function. This uncertain cost vector is then assumed to be generated from a parametric probabilistic model conditioned on an external feature vector. Similar probabilistic conditions may also happen with inequality constraints. However, the parameter of the probabilistic model is usually unknown. As a result, it may be appropriate to choose a parameter by minimizing the expected cost of making a suboptimal decision using an inaccurate parametric distribution of uncertainty. This idea is different from SPO approach in two respects. First, it assumes a parametric distribution of the uncertainty which is then involved in making the decision. In contrast, the SPO approach takes a deterministic prediction from historical data for the purposes of decision-making. Second, the parametric distribution is chosen by the measure of the cost value whereas, in the SPO approach the prediction is chosen by a measure of the excess cost. Moreover, some interesting examples are illustrated in Donti et al.[14] including the inventory stock problem and generator scheduling problems with the promising results.

3. Stochastic Programming (SP): Cost and Constraint Uncertainty

From its earliest version by George Dantzig in 1955, to the most recent class of SP models, uncertainty has been modeled in SP using random variables in both objectives and constraints. It is usually stated in the setting of an evolving sequence of random variables, although for our purposes, we will focus on its simplest two-stage version which may be stated as follows.

$$\min \quad f_1(x_1) := c_1(x_1) + \mathbb{E}\left[h_2(x_1, \tilde{\omega_1})\right] \tag{7a}$$
$$\Pr(g_2(x_1, \tilde{\omega}_1) \le 0) \ge p_1 \tag{7b}$$
$$x_1 \in \mathbf{X_1}, \tag{7c}$$

where for a particular realization ω_1 of $\tilde{\omega_1}$, yields $h_2(x,\omega_1)$ defined by

$$h_2(x_1,\omega_1) = \min \quad f_2(x_2; x_1, \omega_1) \tag{8a}$$
$$x_2 \in \mathbf{X_2}(x_1, \omega_1). \tag{8b}$$

The subscripts for the functions c, h, g, f are intended to convey the time period when the information to evaluate them is revealed. However, whenever there can be no confusion, we will drop the subscripts to ease the notation. It is important to note that the random variables in SP accommodate uncertainty in constraints as well as the objective function. In some instances, one could use the data directly as in empirical risk minimization or sample average approximation (SAA), and in others, a distribution is assumed to have been chosen.

The above statement of SP covers specific formulations of "SP with recourse" as well as "SP with chance constraints". This structure is general enough to allow risk measures (e.g. Downside Risk, Conditional Value at Risk, and others). Indeed, with an appropriately nested collection of random variables, one can also state a multi-stage decision model using extensions of the above mathematical statement. We refer to Birge and Louveaux [8] for SP models and algorithms, Shapiro et al. [47] for mathematical structures and properties, and to Sen [42] for shorter encyclopedic overviews. For the most part, SP models have focused on convex optimization models, although there have been significant advances in stochastic mixed-integer programming in recent years (see Küçükyavuz and Sen [33]). If one wishes to guard against misspecified distributions, we recommend models which are distributionally robust (e.g., Delage and Ye [12]). Our focus on the fusion of data and decision science limits our ability to focus on many aspects of modeling uncertainty, including the work on chance-constrained problems. It is also worth reiterating that there are a variety of applications in the volume edited by Wallace and Ziemba [50] which is entirely devoted to applications of SP.

For the purposes of this tutorial we use SP in dual roles: a) as a prototypical decision model (which accommodates uncertainty in both the objective and constraints) to make the transition from PO (or SPO) to Learning Enabled Optimization (LEO) presented in the next section, and b) as a vehicle to illustrate an often-overlooked aspect of SP modeling, namely, model validation. Because model validation and selection are an integral part of data science (and SL/ML) we present some ideas underlying SP model validation in this section. We begin with a brief commentary on the character of solutions from SP.

3.1. SP as a Bridge and the Structure of Hedged Solutions

In order to highlight how SP provides a bridge between data and decision sciences, we first present the well known SVM model as a SP. To do so, we will first admit the possibility of using an empirical distribution into the SP formulation. In this case, the notion of empirical risk minimization of SL/ML translates to a Sample Average Approximation (SAA) commonly used of SP.

Example 3 (Support Vector Machines). Consider a binary classification problem. Suppose that we have a training set consisting of N objects which belong to one of two categories: I^+ and I^- (and $I = I^+ \cup I^-$). With each of these categories we have a collection of objects $i \in I^+$ whose features are quantified in a vector $Z_i \in \mathbb{R}^n$. For instance, the number of features n may be two (say, weight and volume), and therefore Z_i will denote the weight and volume of object $i \in I^+$. Similarly for each i in I^-, we also have feature vectors $Z_i \in x \in \mathbb{R}^n$. It is common to represent the membership in I^+ by setting a scalar $W_i = +1$, whereas, the membership in I^- is recorded via a scalar $W_i = -1$. For the binary classification problem, we state the classification problem as one of identifying a vector $\beta \in \mathbb{R}^{n_1}$ and a scalar β_0 such that the function

$$h(\beta, \beta_0; z) = \beta^\top z + \beta_0 \tag{9}$$

assumes positive values for $i \in I^+$, and analogously assumes negative values when $i \in I^-$. In case of affine functions such as the one shown in (9), the goal of the SVM problem is to find a vector $(\beta_0, \beta) \in \mathbb{R}^{n+1}$ such that the points in I are classified correctly, to the extent possible.

Note that because the threshold zero is common to both inequalities, there is the likelihood of ambiguities in the decision-making when the value $(\beta^\top z + \beta_0)$ turns out to be relatively small. It might actually be more effective to compare the expression in (9) to $+1$ or -1 because there is a separation. Thus we should check whether $\beta^\top z + \beta_0 \geq 1$ or $\beta^\top z + \beta_0 \leq -1$. As a result, we seek a vector $\beta \in \mathbb{R}^n$ (where n is the number of features) and a scalar β_0 such that the points in I^+ and I^- are properly classified. One way to do this is to solve the following SAA approximation using the empirical distribution induced by the sample $\{w_i, Z_i\}_{i\in I}$. Then, denoting the expectation with respect to the empirical distribution $\hat{\mathbb{E}}$, we have

$$\begin{aligned} \min \quad & \frac{1}{2}\beta^\top \beta + \gamma \quad \hat{\mathbb{E}}_Z[h(\beta, \beta_0, Z)] \\ \text{where } h(\beta, \beta_0, Z = z) \quad = \quad & \min \quad \sum_{i\in I} (\delta_i^+ + \delta_i^-) \\ & \beta^\top z + \beta_0 + \delta_i^+ \geq 1, \quad i \in I^+ \\ & \beta^\top z + \beta_0 - \delta_i^- \leq -1, \quad i \in I^- \\ & \delta_i^+, \delta_i^- \geq 0, \ \forall i. \end{aligned}$$

Notice in this formulation the data (representing uncertainty) appears in the constraints. When there are a few data points, the above problem can be solved using deterministic quadratic programming (QP). However for a large number of data points, a QP becomes very unwieldy, and decomposition approaches of SP have been studied (e.g., PEGASOS Shalev-Shwartz et al. [46]). It is also interesting to note that the regularizer (quadratic term) in SVM also appears within SP, especially in the Stochastic Decomposition algorithm (Higle and Sen [23] and Sen and Liu [45]). In this sense SP provides an excellent vehicle for bridging the fields of Data and Decision Sciences[4].

Example 4 (Hedging when Networks Links Fail). The act of "hedging" is a common strategy in financial applications and for this reason, SP has made significant in-roads into that discipline. The same design principles are very useful for designing resilient engineering systems. Such a principle essentially states that the number of non-zero elements of an optimal solution of a stochastic LP is usually larger than that of an instance in which the random variables are replaced by their expected values (Murphy and Sen [38]). The following example illustrates this principle for a small network planning problem.

Consider an undirected network with K links, and assume that some of these links are failure prone. When a link fails, it cannot carry any flow; however, if the link is up, it allows a flow within the range $[0, x_i]$, where $x_i > 0$ denotes the link capacity chosen for the link with index i. In this example, there are two types of links: i) those with zero failure rate, and therefore, these have a fixed capacity. However, such links can be relatively expensive, with the cost denoted by c_0. ii) link with failure rate $r \in (0, 1)$; in this example r is considered to be a constant among all the links which have non-zero failure rate. Therefore, the link size of a given non-zero failure rate link can be considered as a random variable with binomial distribution (i.e., size of link i is 0 with probability r, and is x_i with probability $1 - r$). We also assume that link failures are independent of each other.

For the sake of illustration consider the network shown in figure 4. For simplicity we refer to node pairs A-B, B-C and A-C by an index i for $i = 1, 2, 3$ respectively, and ω_1, ω_2 and ω_3 denote the number of requests (demand) for the three node-pairs. The demand values of pair 1 and 2 are labeled as "high" level, which means that the demands obey the distribution in Table 1, while the demand of pair 3 is"low" and the distribution is shown in Table 2.

[4] For SVM data sets see http://archive.ics.uci.edu/ml/index.php

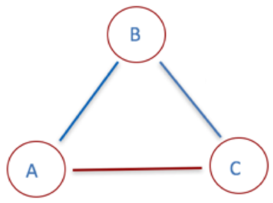


FIGURE 4. Network of Example 3.1

Value	0	1	2	3	4
Probability	0.05	0.2	0.5	0.2	0.05

TABLE 1. High Demand Distribution.

Value	0	1	2	3
Probability	0.1	0.4	0.4	0.1

TABLE 2. Low Demand Distribution.

These demands are served by flows on links AB, BC, and AC. Suppose that link AB and BC have a failure rate $r = 0.3$, and the cost $c_0 = \$1$. On the other hand, link AC has a zero failure rate link with cost \$4. Given a total budget of \$10, the goal is to find the best plan of link capacities. If we adopt a deterministic approach, then we might be tempted to replace the random variables by their expected values. If we do that, the standard LP formulation would be as follows.

Let f_{IJ} = value of flow directly from I to J, where $I, J \in \{A, B, C\}$, $I \neq J$

f_{IJK} = value of flow from I to J, and ends at K, where $I, J, K \in \{A, B, C\}$, $I \neq J \neq K$

r = failure rate of non-zero failure rate link

s_i = number of rejects corresponding to node-pair i, $i = 1, 2, 3$

x_i = size of the link joining node-pair i, where $i = 1, 2, 3$

the problem can be formulated as follows:

$$\begin{aligned}
\min_{x_i, s_i, i=1,2,3} \quad & \sum_{i-1}^{3} s_i \\
\text{s.t.} \quad & s_1 + f_{AB} + f_{ACB} = \omega_1,\ s_2 + f_{BC} + f_{BAC} = \omega_2,\ s_3 + f_{AC} + f_{ABC} = \omega_3 \\
& f_{AB} + f_{BAC} + f_{ABC} \leq (1-r)x_1, \quad f_{ACB} + f_{BC} + f_{ABC} \leq (1-r)x_2 \\
& f_{ACB} + f_{BAC} + f_{AC} \leq x_3,\ x_1 + x_2 + 4x_3 \leq 10 \\
& x_1, x_2, x_3 \geq 0, s_1, s_2, s_3 \geq 0
\end{aligned}$$

The solution of the LP formulation indicates that the link sizes of A-B and B-C are both 5, while the link size of A-C is zero, which means A-C is disconnected. It is not difficult to show that for this decision, the expected number of rejects in this network would be 2. Note also that since the LP optimum can be found at an extreme point, the associated network will have the structure of a tree (no-cycles) as in figure 5.

Now suppose that we wish to use SP to capture the uncertainty of demands ω_1, ω_2 and ω_3. Furthermore, the failure rate r becomes a binonmial random variable. In this case, we minimize the expectation of $h(x, \omega)$ with respect to the random variables $\{\omega_1, \omega_2, \omega_3, r\}$. Therefore the SP formulation is as follows.

$$\min \quad \mathbb{E}_{\tilde{\omega}, \tilde{r}}\, [h(x, \tilde{\omega}, \tilde{r})]$$

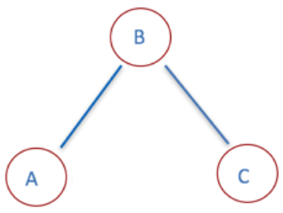


FIGURE 5. Solution of Deterministic LP

$$\begin{aligned} \text{s.t.} \quad & x_1 + x_2 + 4x_3 \leq 10, \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

where for each given outcome (ω, r) and any x, we define

$$\begin{aligned} h(x,\omega,r) = \min \quad & \sum_{i-1}^{3} s_i \\ \text{s.t.} \quad & s_1 + f_{AB} + f_{ACB} = \omega_1, \ \ s_2 + f_{BC} + f_{BAC} = \omega_2, \ \ s_3 + f_{AC} + f_{ABC} = \omega_3, \\ & f_{AB} + f_{BAC} + f_{ABC} \leq (1-r)x_1, \quad f_{ACB} + f_{BC} + f_{ABC} \leq (1-r)x_2, \\ & f_{ACB} + f_{BAC} + f_{AC} \leq x_3, \\ & s_1, s_2, s_3 \geq 0 \end{aligned}$$

For this SP formulation, the solution of the two-stage SP is $\{x_1, x_2, x_3\} = \{3, 3, 1\}$ with expected number of rejects $= 1.8$. Thus, we expect a 10% improvement in performance by using the SP model, provided the distribution is correct. In any event, the notion of hedging mentioned earlier should be clear from the structure of the solution: The LP formulation provided a network design which was a tree, whereas, the SP formulation provides a cycle in the design. The presence of a cycle in Figure 6 creates the opportunity for hedging. Thus for cases in which one of the failure-prone links fails, all three nodes are still able to communicate. On the other hand, the solution provided by the deterministic model, being a tree, is such that even if one link breaks down, there can be no communications in the network, thus leading to poor performance.

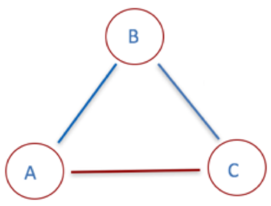


FIGURE 6. Solution of Stochastic LP

3.2. SP Model Validation

Depending on the size of the sample space associated with the random variables in an SP formulation, one can adopt direct methods (as in solving SVM problems by QP solvers), or using decomposition schemes which are designed to take advantage of the decomposable structure of the SP. Both kinds of algorithms are available to SAA (or empirical risk minimization). If sampling is carried out in an "outer loop", the methodology gets the label of "external" sampling because the numerical solution of the approximation is carried out in an "inner loop", separately from sampling. For most SP applications, this is the more popular approach (see Mak et al. [37], Bayraksan and Morton [3]) and has been used for some problems of practical size in Linderoth et al [34]). For "internal" sampling algorithms, every iteration of optimization is followed by a sampling phase which enlarges the sample size according to some rule (Higle and Sen [24], Royset and Szechtman [40]). In either case, replications are essential for reduced variance estimates of both objective value as well as solutions.

One intrinsic difference between replication for simulation studies, and replications in SP is that the latter creates the possibility of generating several candidate solutions. Indeed, Linderoth et al. [34] report wide disparity of solutions of sampled instances of SSN, even with a sample size of 5000. (For further information on SSN, please see [44] as well as the subsection on "Verification" in this paper.) In such cases, the question is: which decision should be recommended? The traditional idea is one where a preliminary objective function estimate is obtained for solutions from each replication, and then one successively prunes them, and the objective value estimates are refined, until the subset of solutions is small enough to recommend a final decision ([34]). Such procedures are better formalized in terms of ranking and selection methods in simulation optimization. Nevertheless, they can be extremely time and resource intensive. In order to overcome these problems, Sen and Liu [45] recommend the "Compromise Decision". This idea provides a relatively fast and effective technique for identifying a low-variance solution from among several candidates. Although it was originally tested for the SD algorithm, it is applicable to any scheme where each sampled approximation is solved by using a stochastic proximal (sub)gradient algorithm (e.g., Atchade et al. [1]).

To be specific, let ν denote an index of the replications ($\nu = 1,\ldots,M$) and the sample average approximation (SAA) for replication ν will have a sample size N^ν. Such an SAA function can be defined by

$$\hat{f}^\nu(x) = c_1(x) + \sum_{i}^{N^\nu} h_1(x,\omega_i). \tag{10}$$

The right hand side of the above function have subscripts as in (7a), although we omit the subscript on the left hand side to ease the notational burden. Let $f^\nu(x)$ denote a piecewise linear lower bounding approximation of $\hat{f}^\nu(x)$, using finitely many subgradients. Given $\rho^\nu > 0$, suppose that

$$\mathbf{x}^\nu \in \operatorname*{Min}_{x\in X} \quad f^\nu(x) + \frac{\rho^\nu}{2}\|x - \mathbf{x}^\nu\|^2. \tag{11}$$

The above setting is also the standard instance at the culmination of a replication in SD [23] where ρ^ν is updated during the course of the algorithm. Let $\bar{\rho}$ denote the sample average of $\{\rho^\nu\}_{\nu=1}^M$, and consider the following problem known as the "Compromise Problem."

$$\operatorname*{Min}_{x\in X} \quad \frac{1}{M}\sum_{\nu=1}^{M}\left[f^\nu(x) + \frac{\bar{\rho}}{2}\|x - \mathbf{x}^\nu\|^2\right]. \tag{12}$$

Problem (12) may be interpreted as a "grand" SAA approximation, and as a result, the approximation has many more samples which produces a decision with much lower variance than any of the individual solutions, and moreover, the bias from SP is also reduced. To formalize this result, let $\mathbf{x}^c$ denote an optimal solution to (12), and we refer to it as the compromise decision. Define $\bar{\mathbf{x}} = \frac{1}{M}\sum_\nu \mathbf{x}^\nu$, and define $\bar{F}_M(x) = \frac{1}{M}\sum_{\nu=1}^M f^\nu(x)$. The following theorem is presented in Sen and Liu [45].

Theorem 1. Assume that the SP model is a convex program and let $N = \min_\nu N^\nu$. For replications $\nu = 1, \dots, M$, let x^ν denote an optimal solution of (11). Assume that for each ν, at least one subgradient of f^ν is a subgradient of $\hat{f}^\nu$ at the point $\mathbf{x}^\nu$, while all others subgradients (finite in number) are $\epsilon-$subgradients of $\hat{f}^\nu$ at $\mathbf{x}^\nu$. Then the following hold true.
a) If $\mathbf{x}^c = \bar{\mathbf{x}}$, then $\mathbf{x}^c$ solves

$$\underset{x \in X}{\text{Min}} \quad \bar{F}_M(x) := \frac{1}{M} \sum_{\nu=1}^{M} f^\nu(x), \tag{13}$$

and b)

$$|f(\mathbf{x}^c) - \bar{F}_M(\mathbf{x}^c)| = O_p((NM)^{-\frac{1}{2}}). \blacksquare \tag{14}$$

The most important consequence of the above result is that it creates an easily implemented stopping rule even when the solutions $\mathbf{x}^\nu$ are very disparate. Moreover, the right hand side of (14) implies that for M reasonably large, the objective function estimate of the compromise decision has lower variance than solutions $\mathbf{x}^\nu$ for any individual replication (which are of the order of $O_p(N^{-1/2})$.

The condition that $\mathbf{x}^c = \bar{\mathbf{x}}$ might seem somewhat strict, but as it turns out, it can be relaxed to the case in which $\|\mathbf{x}^c - \bar{\mathbf{x}}\| = \varepsilon$, whence one can infer an error tolerance on the sub-optimality of $\mathbf{x}^c$, depending on the value ε. This result, which is also proven in (Sen and Deng [43]) is also the basis of “Statistical Optimality” presented in the next section.

3.3. SP Model Verification

In very large scale (or difficult) instances, where validation may be too onerous, the verification process becomes a “zero-th” (as opposed to a first-order) test. This subsection therefore revolves around large scale and difficult instances. Unlike the validation process which seeks optimality due to replicated runs, the verification process is intended to only test whether the predicted objective lies within the 95% confidence interval, and hence less stringent than the optimality tests presented in the previous subsection. *The intuition behind this is that an SP solver produces biased (low) objective estimates until the sample size is large enough. Thus if one can ascertain that the 95% Confidence Interval obtained using a Monte Carlo process already contains the objective estimate predicted by the SP solver, then this lends credence to the view that the sample size is large enough.* However, it is important to recognize that the above process may not be acceptable if the width of the 95% C.I is so large that unattractive decisions become acceptable.

Example 5 (Verification of Statewide Electricity Dispatch). This example is from a case study of realistic power grid. The economic dispatch model for the state of Illinois was part of a study reported in Gangammanavar et al [18]. Because the study included a significant number of wind farms (12), the electricity dispatch problem was required to be solved in 10 minute intervals, with each planning period covering the next 60 minutes in the second stage model. This two stage SP (with seven 10-min. intervals) included 84 correlated wind random variables together with the other thermal generators in the state’s portfolio. If we were to use an “external” sampling approach with only 100 scenarios representing 84 correlated random variables, we would have to solve an LP with 4 million columns and 3 million rows within a few minutes (on the order of 5-7 mins). This was out of the question because our preliminary experiments suggested that using an external sampling approach (using an LP solver) with only 5 scenarios would exceed 10 hours of computational time with a state-of-the art LP solver. We decided to use the SD algorithm as mentioned above. We were able to get the SD solver to produce results within the 20 min. on ordinary desktop machine. We concluded that we could easily scale up to about 5 minutes per SD solve, if we used a more powerful computing platform. However, replications requiring several runs would be far too stringent for application that runs every 5-10 minutes. So, we decided to verify (using Monte Carlo Sampling) whether the predicted objective value is reliable.

The verification results in Table 3 are reproduced from Gangammanavar et al [18]. These computational experiments were based on the electrical grid of the Illinois system, and the table provides both predicted value as estimated by SD, and a validated 95% C.I., for which an external simulator generates a verification dataset. Note that the predicted value in row 2, column 2, is within the 95% C.I. reported in row 2, column 3. This is in line with the comments of the previous paragraph[5]. The point of this example is to illustrate that when the SP algorithm is able to predict an objective function estimate within a 95% C.I (produced using Monte Carlo sampling), it may be a reasonably good decision.

Algorithm	Sample Size	Predicted value	95% C.I.
External Sampling (LP)	5	12,811,076	[13,722,876;13,797,483]
Internal Sampling (SD)	138	13,721,268	[13,660,228;13,742,965]

TABLE 3. Verification of Electricity Dispatch (10% Wind Penetration) (Gangammanavar et al. [18])

Example 6 (Verification of a Predecessor to SSN). SSN is a notoriously difficult SLP model which has been circulated within in SP community for over 20 years. The most recent computational results with SSN, presented in Sen and Liu [45] are a clear testament to the progress that SLP methodology has made in recent years. The data set which we discuss below is the original version of SSN which could not be circulated because of confidentiality agreements. That specific instance (which was called SSBN) was associated with a study to introduce private line services in a network had the same basic character as SSN (86 demand pairs, 89 links and 706 potential routes, Sen et al [44]). Neither the computers of that era, nor the SP methodology of that era were capable of solving the SSN instance multiple times to carry out the type of Model Validation that we recommend for today's procedures. As a result, the authors of SSN created an alternative discrete-even simulation which was then used as a proxy for an alternative model. The plots shown in Figure 7 are intended to verify whether the population of predicted objectives match with a simulated (verification) dataset. Note that Figure 8a plots the estimated mean value of rejected calls associated with plans corresponding to certain iterations of the algorithm, whereas Figure 8b plots the estimated value of percent blocking within a discrete-event simulation for the same plans of Figure 8a. Figure 8c can be obtained by normalizing the data for estimated mean values and percent blocking values so that the two sets of values can be compared. The correlation between the objectives of the SP model, and the discrete-event simulation appears to be striking because the data from the two alternative approaches in Figure 8c lie approximately near the diagonal. This indicates that the iterative objective value improvements reported at various SD iterations fit the simulated results reasonably well. Again, this is not a validation of optimality, but a verification of objective value estimates via comparisons to discrete-event simulation.

4. Learning Enabled Optimization: Models and Concepts

As discussed above SL/ML provides support for predictive analytics, whereas, optimization forms the basis for prescriptive analytics, and the methodologies for these are built (somewhat) independently of each other. The process recommended for SL/ML is summarized in Figure 8a in which the entire data set is divided into two parts (Training and Validation),

[5] Over a 10 min. interval, estimated savings by SD over LP is \$58,583 for the state

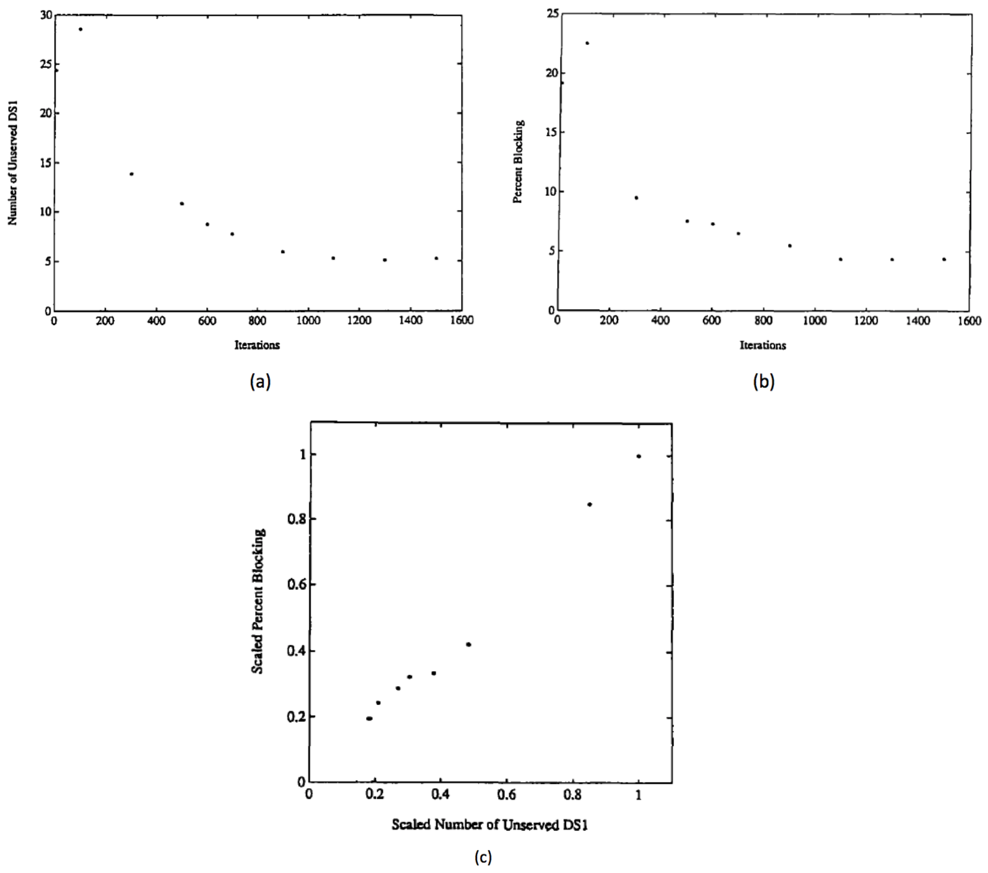


FIGURE 7. SP Validation for SSN Problem (Sen et al.(1994) [44])

with the former being used to learn model parameters (for a predictive model), and the latter data set used for model assessment and selection. Once a model is selected, it can be finally tested by either simulation or using an additional "test data set" for trials before adoption. The LEO framework greatly enhances the potential for fusion as shown in Figure 8.

Example 7 (Wyndor-Uncertainty with Latent Random Effects). Consider a well known example called "The Wyndor Glass Co." In this example, Hillier and Lieberman [25] address resource utilization questions arising in the production of high quality glass doors: some with aluminum frames (A), and others with wood frames (B). These doors are produced by using resources available in three plants, named 1, 2, and 3. The data associated with this problem is shown in Table 4. The product mix will not only be decided by production capacity, but also the potential of future sales. Sales information, however, is uncertain and depends on the marketing strategy to be adopted. Given 200 advertising time slots, the marketing strategy involves choosing a mix of advertising outlets through which to reach consumers. Exercising some "artistic license" here, we suggest that the advertising data set in James et al. [27] reflects sales resulting from an advertising campaign undertaken by Wyndor Glass. That is, the company advertises both types of doors through one campaign

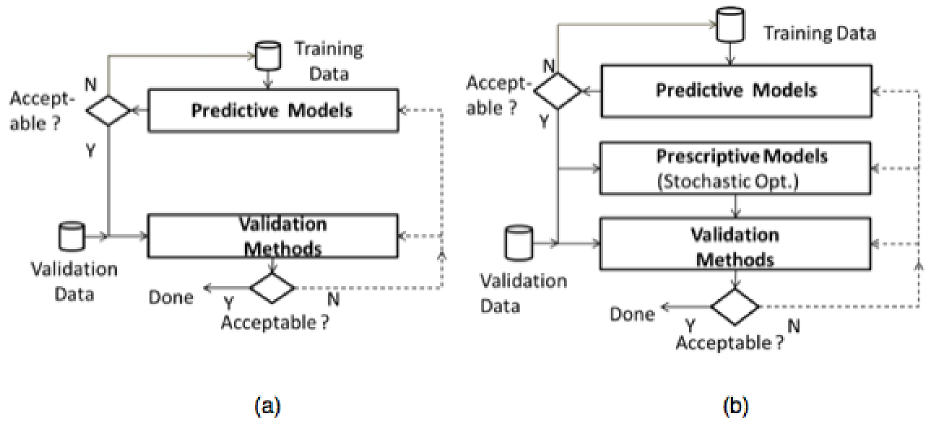


FIGURE 8. Statistical Learning and Learning Enabled Optimization

Plant	Prod. time for A (Hours/Batch)	Prod. time for B (Hours/Batch)	Total Hours Available
1	1	0	4
2	0	2	12
3	3	2	18
Profit per Batch	\$3,000	\$5,000	

TABLE 4. Data for the Wyndor Glass Problem (Hillier and Lieberman [25])

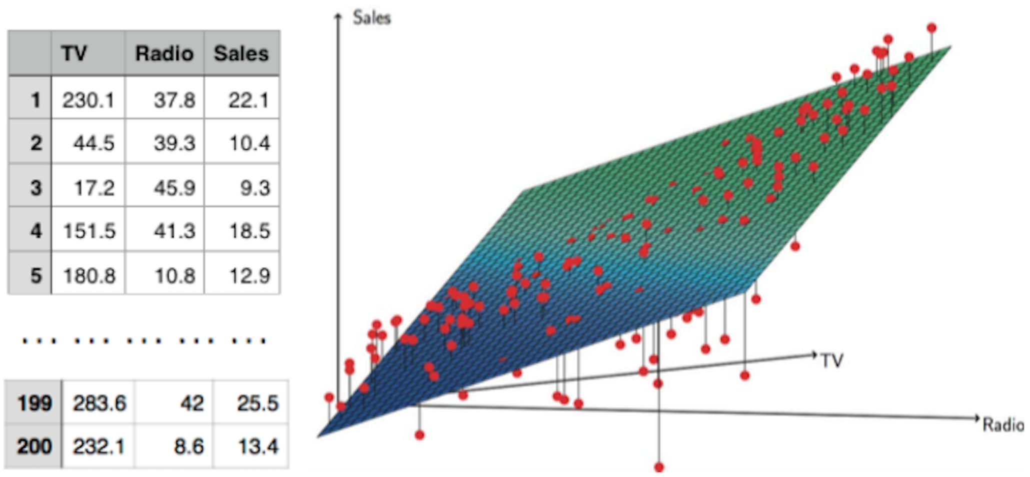
which uses two different media, namely, TV and radio[6]. Note that in the original data set advertising strategy is represented as budgeted dollars, whereas, we have revised it to represent the number of advertising time slots. Thus in our statistical model, sales predictions are based on the number of TV and radio advertising time slots[7]. In our interpretation, product-sales reflect total number of doors sold ($\{W_i\}$) when advertising expenditure for TV is $Z_{i,1}$ and that for radio is $Z_{i,2}$, in number of advertising time slots. (This data set has 200 data points, that is, $i = 1, \ldots, 200$). Let x_1 denote the number of TV advertising time slots, and x_2 denote the number of radio advertising time slots.

Remark 1 (Comments Regarding a Standard SP Approach). The above situation includes some *latent* random effects, and without statistical modeling, it is not clear how these effects can be introduced into a decision model. If one considers an entirely data-driven SP model (i.e., without using a statistical model), then connection between sales and advertising is not represented within the SP approach, and this gap makes the SP model untenable. Our approach to this issue is an combination of PO and SP. In general, we will refer to this approach combining statistical learning and stochastic optimization as Learning Enabled Optimization (LEO). Further comments regarding this difficulty appear at the end of this subsection.

This is a case where a linear regression model reveals the relationship between advertising and sales. When we include an SL model into the production planning model, we are able

[6] The actual data set discussed in James et al.[27] also includes newspapers. However we have dropped it here to keep the example very simple.

[7] The numbers are the same as in James et al [27]



	TV	Radio	Sales
1	230.1	37.8	22.1
2	44.5	39.3	10.4
3	17.2	45.9	9.3
4	151.5	41.3	18.5
5	180.8	10.8	12.9
...	...	...	...
199	283.6	42	25.5
200	232.1	8.6	13.4

FIGURE 9. The Advertising Data Set (Source: James et al [27].

to accommodate the relationships discovered by a learning model. Because of the following assumptions of linear regression, one can justify the fusion of regression and SP.

Data Science We have the following assumptions underlying this statistical estimation model.

(1) Linearity: A linear fit must be deemed appropriate for any Multiple Linear Regression (MLR) model.
(2) Independence: The data (W_i, Z_i) are assumed to be i.i.d.
(3) Normality: Another standard assumption in regression is that the error (i.e., difference between the "least-squares" estimate, and the observed data) is approximately normally distributed.
(4) Error Distribution: This refers to the property that the distribution of the error ε does not vary with the choice of x.

If all these assumptions hold, it can be shown that the variance $\sigma^2 = \mathbb{E}[\frac{\sum_i^N \varepsilon_i^2}{N-n-1}]$. Hence with data points indexed by i, one can estimates σ^2 as follows.

$$s^2 = \frac{\sum_i^N \varepsilon_i^2}{N-n-1}, \tag{15}$$

where, N is the number of data points, and $n+1$ is the total number of parameters $\{\beta_j\}_{j=0}^n$ in MLR.

Ordinarily, for somewhat large data sets, one should be able to justify that the parameters yield stable estimates of the estimated mean $\beta_0 + \sum_j \beta_j x_j$, and the random variable $\tilde{\varepsilon}$ can be approximated by a normal distribution when the number of degrees of freedom is large enough. In the statistical literature, it is customary to suggest that when the degrees of freedom (i.e. $N-n-1$) exceeds about 60, the parameters $\{\beta_j\}_{j=0}^n$ are quite stable, and the errors ε are approximately $\mathbf{N}(0, s^2)$. Thus, for the example with the advertising data set, the number of data points $N = 100$ (for the training data set), and the number of parameters being used is $n+1=3$. Hence, the number of degrees of freedom far exceeds the suggested 60, and as a result, ε in such examples may be reasonably approximated by using normal random variates using $\mathbf{N}(0, s^2)$.

Decision Models The linear regression model for sales is shown in Figure 9, and will be represented by $m_q(x, \xi)$. We consider the following statistical models.

(1) (DF) For deterministic forecasts (DF) we simply use the sales given by $\hat{m}_1(z) = \hat{\beta}_0 + \hat{\beta}_1 z_1 + \hat{\beta}_2 z_2$. Thus, for deterministic predictions, we do not account for errors in forecast, whereas the remaining cases below use some form of error distributions.
(2) (NDC) Another approximation of the sales forecast is $m_2(z,\xi) = (\hat{\beta}_0 + \xi_0) + (\hat{\beta}_1 + \xi_1) z_1 + (\hat{\beta}_2 + \xi_2) z_2$, where (ξ_0, ξ_1, ξ_2) are <u>correlated and normally distributed</u> according to the variance-covariance matrix reported by MLR.
(3) (EAE) This is the empirical additive error model, where $m_3(z,\xi) = \hat{\beta}_0 + \hat{\beta}_1 z_1 + \hat{\beta}_2 z_2 + \xi_0$, and ξ_0 denotes a rvs whose outcomes are $W_i - \hat{m}_3(Z_i)$. We refer to this model as <u>empirical additive errors</u>.

The corresponding first stage is given as follows, where the index q refers to the alternative error models (DF, NDC and EAE).

<u>Index Sets and Variables</u>

$i \equiv$ index of product, $i \in \{A, B\}$.

$y_i \equiv$ number of batches of product i produced.

$$x_q \in \delta - \text{argmax } \{-0.1x_1 - 0.5x_2 + \mathbb{E}[\text{Profit}(x,\ \tilde{\xi}_q \mid Z = z = x)]\} \tag{16a}$$
$$\text{s.t.} \quad x_1 + x_2 \le 200 \tag{16b}$$
$$x_1 - 0.5x_2 \ge 0 \tag{16c}$$
$$L_1 \le x_1 \le U_1, L_2 \le x_2 \le U_2 \tag{16d}$$

where

$$\text{Profit}(x, \xi_q \mid Z = z = x) \quad = \quad \text{Max} \quad 3y_A + 5y_B \tag{17a}$$
$$\text{s.t.} \quad y_A \le 4 \tag{17b}$$
$$2y_B \le 12 \tag{17c}$$
$$3y_A + 2y_B \le 18 \tag{17d}$$
$$y_A + y_B \le m_q(z, \xi_q) \tag{17e}$$
$$y_A, y_B \ge 0 \tag{17f}$$

The model shown in (16-17) closes the gap that we mention in **Remark 1**. In the absence of the statistical model $m_q(z,\xi_q)$, the second stage has a gap in representing the right-hand side (17e) as well as the condition that $Z = x$. While one might consider a penalty formulation for a data-driven SP, such models can be difficult to validate if the penalties are not well-justified. Most penalties in SP models reflect some aspect of reality (e.g., salvage value, or spot market prices in electricity markets, or overtime payments in scheduling labor resources etc.) The lack of a model to describe a relationship between advertising (Z) and sales (W) may lead to a poor choice of penalties which may penalize deviations indiscriminately.

4.1. General LEO Model

As one might expect, this framework of LEO consists of two major parts: the SL/ML piece and the Stochastic Optimization (SO) piece. We begin by stating a regression model in its relatively standard form. Towards this end, let m denote an arbitrary regression model for a training set of observations $\{W_i, Z_i\}$, indexed by $i \in T$, the training data. For notational simplicity we assume that $W_i \in \mathbb{R}$, $Z_i \in \mathbb{R}^p$ and the upper-case letters (W/Z) represent the random variable while the lower-case letters represent the realization (w/z). Given the training data, a class of models $\mathcal{M}$, and a loss function ℓ, the regression is represented as follows:

$$\hat{m} \in \text{argmin}\left\{ \frac{1}{|T|} \sum_{i \in T} \ell(m(Z_i), W_i) \mid m \in \mathcal{M} \right\}. \tag{18}$$

After the SL/ML model is selected, LEO framework uses the error terms of the SL/ML piece to build the SP model so that the model fidelity may be enhanced. We will present alternative ways to build error models.

Empirical additive errors. Suppose that an error random variable has outcomes defined by $\xi_i := W_i - \hat{m}(Z_i)$. Since all the outcomes are associated equal weights, the error random variable is with discrete empirical distribution. In addition, the distribution is assumed to not depending on any specific choice of $Z = z$, therefore the random variable is said to be homoscedasticity.

Multi-dimensional errors. Define a function $m(z, \xi) = \sum_\tau \tilde{\beta}_\tau \phi_\tau(z)$, where $\phi_0 = 1$, and $\phi_\tau(\cdot)$, are deterministic functions, but the parameters $\tilde{\beta}_\tau$ are elements of a multivariate random variable. Let $\hat{m}(z) = \sum_\tau \bar{\beta}_\tau \phi_\tau(z)$, where $\bar{\beta}_\tau = \mathbb{E}(\beta_\tau)$. In this case, a vector of errors associated with an outcome of random coefficients $\{\tilde{\beta}_\tau\}$ is given by the difference $\tilde{\xi}_\tau = \tilde{\beta}_\tau - \mathbb{E}(\beta_\tau)$. The coefficients $\{\tilde{\beta}_\tau\}$ may be correlated, but the multivariate distribution of these coefficients $\{\tilde{\beta}_\tau\}$ are assumed to not depend on any specific choice of $Z = z$ (homoscedasticity).

Assumption 1 (A1). $\{W_i, Z_i\}$ are assumed to be i.i.d. observations of the data process $\{W, Z\}$. Moreover we assume that the errors are homoscedastic.

Assumption 2 (A2). We will assume that decisions in the SP model, denoted x, have no impact on the continuing data process $\{(W, Z)\}$ to be observed in the future.

We will present one of the alternative structures for the SO part of LEO: a model with "Shared Spaces". Another model structure named "LEO with Disjoint Spaces", is provided by Sen and Deng [43]. This class of models is the simplest version of a LEO model in which the values of the statistical inputs (denoted Z), do not assume values in the space of optimization variables (x) of the SP model,

4.1.1. LEO Models with Shared Spaces. Let $x \in \mathbf{X} \subseteq \mathbb{R}^{n_1}$ denote the optimization variables, and suppose that the predictors Z have p elements indexed by the set $\mathcal{J} = \{1, ..., p\}$. SL/ML models are assumed to be given as $m_q(z, \xi_q)$ indexed by q. Consider an SP model whose decisions x, have a subset of variables $(x_j, j \in J \subset \mathcal{J})$ which take values in the same space as the predictor data. Hence these models will be referred to as "models with Shared Spaces", and the objective function we formulate is as follows.

$$f_q(x) := c(x) + \mathbb{E}_{\xi_q}[H(x, \tilde{\xi}_q | Z = z, z_j = x_j, j \in J)]\}, \tag{19}$$

where, H is a convex function over the feasible set $\mathbf{X}$, and the outcomes h are defined as follows.

$$\begin{aligned} h(x, \xi_q | Z = z, z_j = x_j, j \in J) := \min \quad & d(y) \\ \text{s.t.} \quad & g(x, y) - m_q(z, \xi_q) \le 0 \\ & z_j = x_j, j \in J \\ & y \in \mathbf{Y} \subseteq \mathbb{R}^{n_2} \end{aligned} \tag{20}$$

Note that we assume that $h(x, \cdot)$ defined in (20) is a convex function in x. In the form of a model with shared spaces, the response function value is a random variable $H(x, \tilde{\xi}_q | Z = z, z_j = x_j, j \in J)$. The value function h defined in (20) goes by several alternative names such as "recourse" functions in SP or "cost-to-go" function in Dynamic Programming. Note that H is a random cost-to-go function, and h are its outcomes, and $\mathbb{E}_{\tilde{\xi}}[H]$ is the expected cost-to-go function.

Assumption 3 (A3). We assume that c, d, g are convex functions, the sets $\mathbf{X}, \mathbf{Y}$ are convex sets, $h(x, \cdot)$ is a convex function in x, and $m_q(z, \xi_q)$ is concave in $z_j, j \in J$ for all ξ except on a set of $\mathcal{P}_q$-measure zero.

Assumption 4 (A4). When decisions x are allowed to assume values in a subspace of observations of the rv Z, we assume that $\mathbf{X}$ is a subset of $\Pi_J\,(conv\ \{Z_i\}_{i \in T})$, where the notation $\Pi_J(\cdot)$ denotes the projection on to the subspace of variables indexed by J. If the number of decision variables are the same as those in Z, then set of feasible solutions ($\mathbf{X}$) is compact as well.

4.2. Statistical Optimality

In order reflect the aspirations of a modeler (in search of decisions with performance guarantees), we state the LEO model in terms of a probabilistic guarantee. Let $\mathcal{P}_q$ denote the probability distribution of a random variable $\tilde{\xi}_q$. For LEO models, we will seek a pair (x_q, γ_q) such that for a pre-specified accuracy tolerance $\delta > 0$, we have

$$\gamma_q := \mathcal{P}_q \; (x_q \in \delta - \arg\min\{f_q(x) | x \in \mathbf{X}\}), \tag{21}$$

with $\gamma_q \geq \underline{\gamma}$. In our experiment, we choose $\delta = 1\%$ and $\underline{\gamma} = 0.95$. We refer to the above requirement as Statistical Optimality. As the reader might notice, (21) states our aspirations for a LEO model in such a manner that we can report the following critical quantities: $\delta,\ \gamma_q,\ x_q$ for a model indexed by q.

Unlike most deterministic algorithms, both SP and LEO call for a requirement of reliability, which leads us to the new concept of computational statistical optimality. However, most of the SP literature reports computations using deterministic algorithms. With a few exceptions, the task of validation is mainly undertaken via statistical optimality bounds such as those in (e.g., Higle and Sen [22], Higle and Sen [24], Mak et al. [37], Kleywegt et al. [31], Bayraksan and Morton [4], Glynn and Infanger [19]). A complete mathematical treatment of these concepts appears under the banner of "Statistical Validation" in Shapiro et al.[47], and a detailed tutorial for SAA appears in Homem-de Mello and Bayraksan [26]. The current subsection includes an extended result in which the statistical optimality of a compromise decision can be computed and verified to a given accuracy. Our approach combines concepts from external sampling (SAA), as well as internal sampling (Stochastic Decomposition) within one common framework.

Before getting into the details of the theorem, we impose the assumptions required for the asymptotic convergence of the standard Stochastic Decomposition (SD) method. Let $\nu = 1, ..., M$ to denote the index of replications, and for each ν, the SD algorithm is assumed to run for $K_\nu(\varepsilon)$ samples, to produce a terminal solution $\mathbf{x}^\nu(\varepsilon)$, and a terminal value f_ε^ν, where ε is the stopping tolerance used for each replication. The grand-mean approximation is defined to be $\bar{F}_M(x) := (1/M)\sum_{\nu=1}^M f^\nu(x)$, where $\{f^\nu\}_{\nu=1}^M$ denotes terminal value function approximations for each replication ν. In addition, $\bar{\mathbf{x}} = (1/M)\sum_\nu x^\nu$, and the LEO compromise decision $\mathbf{x}^c$ is defined as the optimal solution of the analog of (12) for the LEO model in (19). Then following theorem gives the probability bound of $\mathbf{x}^c \in S(\delta_u)$.

Theorem 2. Define $S(\delta_u) = \{x \in \mathbf{X} \mid \bar{F}_M(x) \leq f^* + \delta_u\}$ and let $F(x, \tilde{\xi}) := c(x) + H(x, \tilde{\xi})$ denote the objective rv in (19). Suppose for each outcome ξ, $\kappa(\xi)$ satisfies $|F(x', \xi) - F(x, \xi)| \leq \kappa(\xi)||x' - x||$. We define the Lipschitz constant of $\mathbb{E}_\xi[F(x, \tilde{\xi})]$ as $L = \mathbb{E}_\xi[\kappa(\tilde{\xi})]$. Suppose $\mathbf{X} \subseteq \mathbb{R}^n$ has a finite diameter D, M stands for the number of replications in solving the SP, N denotes the minimum of sample size of all the replications, and let the tolerance level $\delta_u > \delta$, with $\delta = \bar{\rho}||\mathbf{x}^c - \bar{\mathbf{x}}||^2$. Then we have the following inequality:

$$\Pr(\mathbf{x}^c \in S(\delta_u)) \geq 1 - \exp\left(- \frac{NM(\delta_u - \delta)^2}{32L^2D^2} + n \ln\left(\frac{8LD}{\delta_u - \delta}\right)\right). \blacksquare \tag{22}$$

One of the more interesting consequences of the above result, is that when the second stage model is a linear program, all quantities used in the above probability calculation can be observed during the course of a solution algorithm (e.g., Stochastic Decomposition), and as a result we can provide good reliability estimates of the optimality of the proposed solution (for the given distribution).

4.3. Statistical Tests for Validation

The three standard statistical tests are summarized as follows. If the null hypothesis is rejected in any of these tests, the corresponding LEO model will not be selected.

<u>χ^2 Test for Error Terms and Cost-to-go Objectives.</u> The data-set is expected to have two parts, and we test the null hypothesis that both parts of the data-set share a common distribution. The dataset is allocated into B bins. For the i^{th} bin, denote E_{1i} as the observed frequency for bin i from one sample set, and E_{2i} as the observed frequency for bin i from the other sample set. Then the χ^2 statistic for this data is estimated as:

$$\hat{\chi}^2 = \sum_{i=1}^{B} \frac{(E_{1i} - E_{2i})^2}{E_{1i} + E_{2i}} \tag{23}$$

We calculate the χ^2 distribution with B degrees of freedom which provides the standard value $\chi^2(B)$, and the probability $p = Prob(\chi^2(B) > \hat{\chi}^2)$. If $p \leq \alpha$, in which α represents a significance level, the null hypothesis is rejected; otherwise, we do not reject the null hypothesis.

<u>Cost-to-go Hypothesis test for the Mean value using T-statistic.</u> For cost-to-go objectives, we introduce the null hypothesis test for the mean value. Suppose we have two independent samples which reflect the training and validation sets of size m_1 and m_2 respectively, the mean values of them are h_1, h_2, and the standard deviations are s_1, s_2. To determine whether two sample means are significantly different with $\alpha = 0.05$, then T-statistic of two group means is $t = \frac{|h_1 - h_2|}{\sqrt{s_1^2/m_1 + s_2^2/m_2}}$. We compare the calculated T-value with $t_0 = 1.96$; if $t > t_0$, we reject the null hypothesis because it indicates that the mean values of two samples are significantly different. If this hypothesis is rejected, the objectives of training and validation set are considered to be different on the basis of the first moment.

<u>Cost-to-go Hypothesis test for the Variance value using F-statistic.</u> Besides the hypothesis test on the first moment level, we also perform a test for the variance value based on the F distribution. The F-test is often used to test if the variances of two samples are consistent. The null hypothesis H_0 is defined as: the variances of two samples are equal. Suppose we denote the observed variances of two samples as s_1^2 and s_2^2, then the F statistic is the following $f = s_1^2/s_2^2$. Suppose we choose the significance level α, sample 1 has sample size N_1, and sample 2 has sample size N_2, then the critical region is decided by two values from F-distribution: $F_{\alpha/2, N_1-1, N_2-1}$ and $F_{1-\alpha/2, N_1-1, N_2-1}$. If $F_{\alpha/2, N_1-1, N_2-1} \leq f \leq F_{1-\alpha/2, N_1-1, N_2-1}$, we do not reject the null hypothesis.

4.4. Confidence and Prediction Intervals

The 95% Confidence Intervals estimates the sample mean of the validated cost-to-go function (using the validation data set) for a given decision x_q (see (21)). This is calculated using cross-validation which is similar to that used in SL or simulation, and moreover, it has the same interpretation.

While the interpretation of 95% confidence intervals are very similar to that used in regression (and other SL models), prediction intervals are interpreted differently, in our setting: we use the shortest interval which covers 95% of the validated cost-to-go population as the 95% prediction interval (95% PI). As a result, the 95% PI may not be symmetric (See Frey [16]). We obtain the 95% PI by solving a combinatorial optimization problem as described below. While this problem can be stated as an extension of a knapsack problem, we propose a somewhat tighter formulation (with additional valid inequalities) which is proposed in Sen and Deng [43].

4.5. Model Selection

In SL/ML pieces of LEO model, multiple classes of models will be available to choose from. Therefore, we need to design a criterion based on the decisions to choose the best model class among multiple classes. Den Boer and Sierag [13] proposes a decision based model selection (DBMS) method that evaluates models based on the quality of the decisions they produce. The key idea of DBMS method is using resampling technique to evaluate the performance of the decision generated by using different models. Our approach to model selection is similar to the DBMS methodology with an important distinction which we note subsequently. Specifically, one considers a decision making problem

$$\begin{aligned} \min \quad & \mathbb{E}[f(x, W(x))] && (24) \\ \text{s.t.} \quad & x \in \mathbf{X} && (25) \end{aligned}$$

where $f : \mathbf{X} \times \mathcal{W} \to \mathbb{R}$ is a cost function and $\{W(x), x \in \mathbf{X}\} \subset \mathcal{W}$ is a collection of random variables. The distribution of $W(x)$ is unknown to the decision maker but a data set $T^0 = \{(X^i, W^i)\}_{i=1}^n$ is available. Suppose there is a set $\mathcal{H} \triangleq \{\mathcal{M}^{(0)}, \mathcal{M}^{(1)}, \ldots, \mathcal{M}^{(Q)}\}$ which includes the class of parametric models of the probability distributions we consider. In example 7, the total number of models is $Q = 3$. Each class of models $\mathcal{M}^{(q)} = \{F^q_{x,\theta} \in \mathbb{F} \mid x \in \mathbf{X}, \theta \in \Theta^{(q)}\}$ where $\mathbb{F}$ is the set of cumulative distribution functions on $\mathcal{W}$, and $\Theta^{(q)}$ is a nonempty parameter set. Suppose for each q, the estimator is a function $\varphi^{(q)} : (\mathbf{X} \times \mathcal{W})^n \to \Theta$ which maps the data to parameter values and an optimization algorithm is a mapping $\chi^{(q)} : \Theta \to \mathbf{X}$ such that $\chi^{(q)}(\theta)$ generates an exact/inexact solution which optimizes

$$\int_{w \in \mathcal{W}} f(x, w) \; dF^{(q)}_{x,\theta}(w). \tag{26}$$

Therefore, the decision maker first picks a class of models $\mathcal{M}^{(q)} \in \mathcal{H}$, then estimates the parameter $\theta^q = \varphi^{(q)}(T^0)$, and finally makes a decision $x^{(q)} = \chi^{(q)}(\theta^q)$. In the development by Den Boer and Sierag [13], it is assumed that $\mathcal{M}^0$ is the class of models containing the true model F^0_{z,θ^*} and an $x^{(0)}$ can be obtained from this class. Specifically, let $\hat{\theta} = \varphi^{(0)}(T^{(r)})$ where $T^{(r)}$ is the resampled data such that $T^r = \{(z^i, w^{\tau_i})\}$ and $\tau = \{\tau_1, \tau_2, \ldots, \tau_n\}$ is a permutation of the set $\{1, 2, \ldots, n\}$. The recommendation of Den Boer and Sierag [13] is to choose the model for which the following minimum is attained.

$$\min_{0,\ldots,Q} \int_{w \in \mathcal{W}} f(x^{(q)}, w) dF^0_{x^{(q)},\hat{\theta}}(w). \tag{27}$$

This strategy is shown to have the consistency property such that optimal cost value with the model chosen by DBMS method converges in probability to the optimal cost value with the true model as the data size goes to infinity. Correspondingly, Den Boer and Sierag [13] show consistency in the sense that optimal cost value using the DBMS method converges in probability to the optimal cost value with the chosen model. Moreover, the assumption of the true model and true parameter value can be relaxed to be any one in which the model and the parameter that provide the best explanation of the data.

Our approach adopts a somewhat relaxed set of requirements, namely a) instead of solving (26) exactly, we find an approximate solution through (21), b) we estimate the objective value in (27) using the *validation data set* representing F^0 without assuming that $\mathcal{M}^0$ is available and c) instead of (27), we use the Kruskal-Wallis non-parametric analysis of variance for pairwise comparisons between alternative models. To formally state the process for model selection, let $\tilde{f}_q$ denote the entire population vector of the validated objectives for model q, and let $\bar{f}_q$ denote the sample average of *validated objective* for model q. Then the model selection procedure is as follows. Let $\mathcal{Q} = \{1, \ldots, Q\}$, and choose small scalars $\alpha, \varepsilon_1, \varepsilon_2 > 0$, and do the following for $r \in \mathcal{Q}$:

$$\text{if } \exists \; q \in \mathcal{Q} \;\text{ such that } \Big(\Pr\left(||\tilde{f}_q - \tilde{f}_r|| > \varepsilon_1\right) \geq 1 - \alpha \;\&\; \bar{f}_q \leq \bar{f}_r - \varepsilon_2\Big), \mathcal{Q} \leftarrow \mathcal{Q} \setminus \{r\}. \tag{28}$$

Letting $f^* = \min \; \{\bar{f}_q \mid q \in \mathcal{Q} \}$, we define the optimization error to be the difference $\bar{f}_q - f^*$ for any $q \in \{1, \ldots, Q\}$. The probability estimates above are provided by the non-parametric Kruskal-Wallis test which does not require normality as an assumption. As for details regarding the generalization error for cost-to-go estimates, we refer to Sen and Deng [43]. The final choice of the model should be based on a combination of both optimization and generalization errors of the models.

4.6. Computational Results for LEO-Wyndor

The instances discussed below are developed using existing data-sets and existing optimization models. We include one example for the Shared Spaces LEO structure. Each model requires two aspects: one is the SL/ML aspect, and the other is the decision/optimization aspect. In case of the SL/ML part of the model, we assume that measures that are necessary to create acceptable SL/ML models are undertaken. For the following model, in LEO-Wyndor model with cross-sectional data, outliers in the data should be identified and removed.

The decisions and various metrics discussed earlier are shown in Table 5. Although the prediction interval is listed in a symmetric way, the actual data points are asymmetric with respect to the estimated mean. The last two rows report the probability γ and the corresponding tolerance level δ, which are provided by SD algorithm based on the theorems of statistical optimality. We choose 1% of the mean value of validated objective to be δ_u in Theorem 2.

Notice that the optimization errors appear to be significant, therefore NDC/SD does not yield the most profitable decisions. For the Kruskal-Wallis tests of DF/LP with other methodologies, the p-values are all smaller than 0.01, which implies that there are significant differences between the median ranks of DF/LP and each of the other four approaches. The stepped curve in Figure 10 illustrates the ordering discovered by the Kruskal-Wallis test. The optimization errors are the difference between the estimated mean of the validated objectives of EAE/SD and each of the other two models. The specific optimization errors are listed in the last row of Table 5. EAE/SD leads to more profitable decisions since they stochastically dominate the others. The value of f^* (which is the validated mean 95% C.I) is different from the mid point of the validated 95% P.I. because the latter is asymmetric (see the last column of Table 5).

<u>Capstone Course and Software Platform</u>

We have used some of the material in this tutorial for a capstone course in the Masters of Analytics program at USC. This course requires a Masters level pre-requisite in both predictive and prescriptive analytics. In order to offer this course, we have implemented the LEO protocol presented in Figure 8 through a collection of open source statistical and optimization packages. In addition, the procedures for model validation and selection are included within this software framework. This software is available for non-commercial public use, and can be accessed through `https://sites.google.com/site/uscdatadrivendecisions/home` and `https://github.com/USC3DLAB`.

Models	DF/LP	NDC/SD	EAE/SD
x_1	173.48	181.40	191.40
x_2	26.52	18.60	8.60
Predicted Obj.	\$41,391 (±601)	\$41,492 (±272)	\$42,045 (±465)
Validated Obj. 95% C.I.	\$39,869 (±692)	\$41,865 (±302)	\$42,274 (±493)
Validated Obj. 95% P.I.	\$39,856 (±1,302)	\$41,841 (±639)	\$42,279 (±973)
Probability (γ)		0.9698	0.9872
Tolerance (δ)		694	842
Optimization Error	2405	409	f^*= \$42,274

Table 5. LEO-Wyndor: Comparison of Solutions for Alternative Models

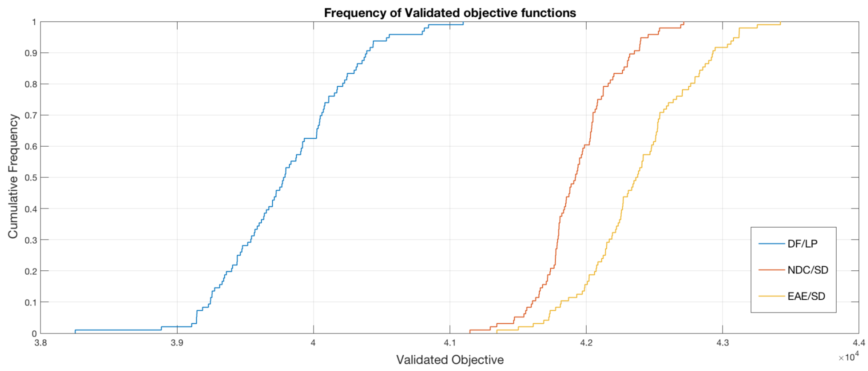


FIGURE 10. LEO-Wyndor: Stochastic Dominance of Validated Objectives under Alternative Models

Models	DF/LP	NDC/SD	EAE/SD
T-statistics($t < 1.96$)	$t = 2.18$	$t = 0.84$	$t = 0.49$
Cost-to-go Test(Mean)	rejected	not rejected	not rejected
F-statistics($0.67 < f < 1.49$)	$f = 1.23$	$f = 1.29$	$f = 1.16$
Cost-to-go Test(Variance)	not rejected	not rejected	not rejected
χ^2 Test p-value ($p > 0.05$)	$p = 0.038$	$p = 0.32$	$p = 0.42$
Cost-to-go Test(Distribution)	rejected	not rejected	not rejected

TABLE 6. LEO-Wyndor: Hypothesis Test Results for Alternative Models

Models	EAE/SD	NDC/SD
DF/LP	2.76×10^{-8}	1.12×10^{-7}
NDC/SD	2.05×10^{-7}	

TABLE 7. LEO-Wyndor: Kruskal-Wallis Test Results ($p < \alpha = 0.01$ implies one dominates another)

Acknowledgments

This research was supported by AFOSR Grant FA9550-15-1-0267, NSF Grant CMMI 1822327, ECCS 1548847 and NSF Grant CMMI 1538605. We also thank Gabe Hackebeil for extending PySP functionality to allow SMPS output which is necessary for our SD code (available from the authors as well as the Github repository).

References

[1] Yevs F Atchade, Gersende Fort, and Eric Moulines. On stochastic proximal gradient algorithms. *arXiv preprint arXiv:1402.2365*, 23, 2014.

[2] Gah-Yi Ban, Noureddine El Karoui, and Andrew E. B. Lim. Machine learning and portfolio optimization. *Management Science*, 2017.

[3] Güzin Bayraksan and David P Morton. Assessing solution quality in stochastic programs via sampling. *Tutorials in Operations Research*, 5:102–122, 2009.

[4] Güzin Bayraksan and David P Morton. A sequential sampling procedure for stochastic programming. *Operations Research*, 59(4):898–913, 2011.

[5] Dimitris Bertsimas, J Daniel Griffith, Vishal Gupta, Mykel J Kochenderfer, Velibor V Mišić, and Robert Moss. A comparison of monte carlo tree search and mathematical optimization for large scale dynamic resource allocation. *arXiv preprint arXiv:1405.5498*, 2014.

[6] Dimitris Bertsimas and Nathan Kallus. From predictive to prescriptive analytics. *arXiv preprint arXiv:1402.5481*, 2014.

[7] Omar Besbes and Assaf Zeevi. On the (surprising) sufficiency of linear models for dynamic pricing with demand learning. *Management Science*, 61(4):723–739, 2015.

[8] John R Birge and Francois Louveaux. *Introduction to Stochastic Programming*. Springer Science Business Media, 2011.

[9] Zhe Chen et al. Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics*, 182(1):1–69, 2003.

[10] William L Cooper, Tito Homem-de Mello, and Anton J Kleywegt. Learning and pricing with models that do not explicitly incorporate competition. *Operations research*, 63(1):86–103, 2015.

[11] Danial Davarnia and Gérard Cornuéjols. From estimation to optimization via shrinkage. *Operations Research Letters*, 45(6):642–646, 2017.

[12] Erick Delage and Yinyu Ye. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations research*, 58(3):595–612, 2010.

[13] Arnoud V Den Boer and Dirk D Sierag. Decision-based model selection, 2016.

[14] Priya L Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning. *arXiv preprint arXiv:1703.04529*, 2017.

[15] Adam N Elmachtoub and Paul Grigas. Smart "predict, then optimize". *arXiv preprint arXiv:1710.08005.*, 2017.

[16] Jesse Frey. Data-driven nonparametric prediction intervals. *Journal of Statistical Planning and Inference*, 143(6):1039–1048, 2013.

[17] Michael C Fu. Markov decision processes, alphago, and monte carlo tree search: Back to the future. In *Leading Developments from INFORMS Communities*, pages 68–88. INFORMS, 2017.

[18] Harsha Gangammanavar, Suvrajeet Sen, and Victor M Zavala. Stochastic optimization of subhourly economic dispatch with wind energy. *IEEE Transactions on Power Systems*, 31(2):949–959, 2016.

[19] Peter W Glynn and Gerd Infanger. Simulation-based confidence bounds for two-stage stochastic programs. *Mathematical Programming*, 138(1-2):15–42, 2013.

[20] Pavithra Harsha, Ramesh Natarajan, and Dharmashankar Subramanian. A data-driven, distribution-free, multivariate approach to the price-setting newsvendor problem. 2015.

[21] Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *Journal of Machine Learning Research*, 16:3367–3402, 2015.

[22] Julia L Higle and Suvrajeet Sen. Stochastic decomposition: An algorithm for two-stage linear programs with recourse. *Mathematics of Operations Research*, 16(3):650–669, 1991.

[23] Julia L Higle and Suvrajeet Sen. Finite master programs in regularized stochastic decomposition. *Mathematical Programming*, 67(1-3):143–168, 1994.

[24] Julia L Higle and Suvrajeet Sen. *Stochastic Decomposition*. Springer, 1996.

[25] Frederick S Hillier and G J Lieberman. *Introduction to operations research.* Tata McGraw-Hill Education, 2012.

[26] Tito Homem-de Mello and Güzin Bayraksan. Monte carlo sampling-based methods for stochastic optimization. *Surveys in Operations Research and Management Science*, 19(1):56–85, 2014.

[27] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 6. Springer, 2013.

[28] Daniel R Jiang, Lina Al-Kanj, and Warren B Powell. Monte carlo tree search with sampled information relaxation dual bounds. *arXiv preprint arXiv:1704.05963*, 2017.

[29] Y-H. Kao, Benjamin V. Roy, and Xiang Yan. Directed regression. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 889–897. Curran Associates, Inc., 2009.

[30] Yi-Hao Kao and Benjamin Van Roy. Directed principal component analysis. *Operations Research*, 62(4):957–972, 2014.

[31] Anton J Kleywegt, Alexander Shapiro, and Tito Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002.

[32] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8), 2009.

[33] Simge Küçükyavuz and Suvrajeet Sen. An introduction to two-stage stochastic mixed-integer programming. *INFORMS TutORials in Operations Research*, 2017.

[34] Jeff Linderoth, Alexander Shapiro, and Stephen Wright. The empirical behavior of sampling methods for stochastic programming. *Annals of Operations Research*, 142(1):215–241, 2006.

[35] Junyi Liu and Suvrajeet Sen. Asymptotic results for two-stage stochastic quadratic programming. *SIAM Journal on Optimization (submitted)*, 2017.

[36] Liwan H Liyanage and J George Shanthikumar. A practical inventory control policy using operational statistics. *Operations Research Letters*, 33(4):341–348, 2005.

[37] Wai-Kei Mak, David P Morton, and R Kevin Wood. Monte carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters*, 24(1):47–56, 1999.

[38] Frederic H Murphy and Suvrajeet Sen. Qualitative implications of uncertainty in economic equilibrium models. In *Decision Making Under Uncertainty*, pages 135–151. Springer, 2002.

[39] R Tyrrell Rockafellar and Stan Uryasev. The fundamental risk quadrangle in risk management, optimization and statistical estimation. *Surveys in Operations Research and Management Science*, 18(1-2):33–53, 2013.

[40] Johannes O Royset and Roberto Szechtman. Optimal budget allocation for sample average approximation. *Operations Research*, 61(3):762–776, 2013.

[41] Cynthia Rudin and Gah-Yi Vahn. The big data newsvendor: Practical insights from machine learning. *DSpace*, 2014.

[42] Suvrajeet Sen. Stochastic programming. In *Encyclopedia of Operations Research and Management Science*, pages 1486–1497. Springer, 2013.

[43] Suvrajeet Sen and Yunxiao Deng. Learning enabled optimization: Towards a fusion of statistical learning and stochastic programming. *INFORMS Journal on Optimization (submitted)*, 2018.

[44] Suvrajeet Sen, Robert D Doverspike, and Steve Cosares. Network planning with random demand. *Telecommunication systems*, 3(1):11–30, 1994.

[45] Suvrajeet Sen and Yifan Liu. Mitigating uncertainty via compromise decisions in two-stage stochastic linear programming: Variance reduction. *Operations Research*, 64(6):1422–1437, 2016.

[46] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

[47] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczynski. Lectures on stochastic programming. *SIAM, Philadelphia*, 10, 2009.

[48] David Simchi-Levi. Om forumom research: From problem-driven to data-driven research. *Manufacturing & Service Operations Management*, 16(1):2–10, 2013.

[49] George J Stigler. The cost of subsistence. *Journal of farm economics*, 27(2):303–314, 1945.

[50] Stein W Wallace and William T Ziemba. *Applications of Stochastic Programming.* SIAM, 2005.