Solving Two-stage Robust Optimization Problems by A Constraint-and-Column Generation Method

Bo Zeng

Department of Industrial and Management Systems Engineering
University of South Florida, Email: bzeng@usf.edu

June, 2011

Abstract

We present a constraint-and-column generation algorithm to solve two-stage robust optimization problems. Compared with existing Benders style cutting plane methods, it is a general procedure with a unified approach to deal with optimality and feasibility. A computational study on a two-stage robust location-transportation problem shows that it performs an order of magnitude faster. Also, it reveals a linkage between two-stage robust optimization and stochastic programming, which may benefits our understanding on both of them.
Key words:
two-stage robust optimization, cutting plane algorithm, location-and-transportation

1 Introduction

Robust optimization (RO) is a recent optimization approach that deals with data uncertainty. Different from stochastic programming, another major approach to deal with data uncertainty, it does not assume probability distribution on the random data while just looks for solutions that are immune from any perturbation within a predefined uncertainty set,[2, 3, 4, 10, 6, 7]. Because it is derived to hedge against any perturbation in the input data, a solution to a (single-stage) RO model tends to be overly conservative. This disadvantage prevents the further application of RO, especially in the environments that decisions are made sequentially. To address this issue, *two-stage RO* and more general *multi-stage RO*, also known as robust adjustable or adaptable optimization, have been introduced and studied [5]. Similar to the case in two-stage stochastic programming, the second stage decision problem will derive recourse decisions after the first stage decisions are made and the uncertainty is revealed. Such a framework enables the decision makers to fully take advantage of the revealed information to make less conservative decisions. Due to the improved modeling ability, 2-stage RO quickly becomes a popular method for decision making with uncertainty. Typical applications include network/transportation problems [1, 14, 11], portfolio optimization [15] and power system control and scheduling [18, 12, 9].

However, 2-stage RO models are very difficult to compute. As shown in [5], even a simple 2-stage RO problem could be NP-hard and become intractable. To overcome the computational burden, two solution strategies have been studied. The first one is approximation algorithms, which assume that the second stage decisions are simple functions, such as affine functions, of the uncertainty, see examples presented in [8]. With this assumption, 2-stage RO formulations can be generally reduced to (single-stage) RO problems. The second type algorithms seek to derive exact solutions using some sophisticated procedures. Those algorithms gradually construct the value function of the first stage decisions using dual solutions of the second stage decision problem [17, 18, 9, 12, 11]. They are actually very similar to Benders' decomposition method in that constraints that approximate the value function are iteratively generated from a subproblem and then supplied to a master problem. We call them *Benders-dual cutting plane algorithms*.

Zhao and Zeng [18] develop a different cutting plane strategy to solve a power system scheduling problem with uncertain wind power supply. It does not create constraints using dual solutions of the second stage decision problem. Instead, it dynamically generates constraints with recourse decision variables for an identified scenario in the uncertainty set in each iteration, which is very different from the philosophy behind Benders-dual procedures. In fact, it is a constraint-and-column generation procedure. Observing in a preliminary computational study that this algorithm is very effective in solving 2-stage robust power system scheduling problems, we believe that it could be refined into a general solution procedure. So, in this paper, we develop and present this solution procedure in a general setting and benchmark with Benders-dual cutting plane procedure. In Section 2, we first give a general formulation of 2-stage RO problem and outline the existing Benders-dual cutting plane procedures. In Section 3, we describe the constraint-and-column generation procedure with some discussions. In Section 4, we apply this algorithm to solve a two-stage robust location-transportation problem and demonstrate its efficiency with respect to Benders-dual cutting plane procedure. We conclude this paper with a discussion on future research directions in Section 5.

We mention that the generated variables and constraints are very similar to those in a 2-stage stochastic programming model. Also, when the uncertainty set is discrete and finite, by enumerating variables and constrains for each scenario in the set, an equivalent regular optimization formulation can be constructed [15]. However, to the best of our knowledge, except the work in [18], no algorithm has been reported to utilize them within a cutting plane procedure, either with discrete or continuous uncertainty set, to solve 2-stage RO problems. So, it is the first time to present this cutting plane algorithm with the constraint-and-column generation strategy in a general setup and compare its computational performance with Benders-dual cutting plane method systematically.

2 Two-stage RO and Benders-dual Cutting Plane Method

Although the whole solution strategy can be easily extended to nonlinear formulations, we focus on linear formulations in this paper where both the first and second stage decision problems are linear optimization models and the uncertainty is either a finite discrete set or a polyhedron.

Let $\mathbf{y}$ be the first stage and $\mathbf{x}$ be the second stage decision variables respectively. Unless mentioned explicitly, no restriction will be imposed on them so that they can take either discrete or continuous values. The general form of 2-stage RO formulation is

$$\begin{aligned} \min_{\mathbf{y}} \quad & \mathbf{cy} + \max_{u \in \mathcal{U}} \min_{\mathbf{x} \in F(\mathbf{y},u)} \mathbf{bx} \\ \text{s.t.} \quad & \mathbf{Ay} \geq \mathbf{d}, \ \mathbf{y} \in \mathbf{S_y} \end{aligned} \tag{1}$$

where $F(\mathbf{y}, u) = \{\mathbf{x} \in \mathbf{S_x} : \mathbf{Gx} \geq \mathbf{h} - \mathbf{Ey} - \mathbf{M}u\}$ with $\mathbf{S_y} \subseteq \mathbf{R}^n_+$ and $\mathbf{S_x} \subseteq \mathbf{R}^m_+$. As a two-stage RO of this form is difficult to solve [5], a few papers solve it approximately by assuming that the value of $\mathbf{x}$ is a simple function of u [8]. However, when the second stage decision problem is not of any simple structure or it could be infeasible for some u, such approximation strategy may not be applicable. A few cutting plane based methods have been developed and implemented to derive the exact solution when $\mathbf{S_x} = \mathbf{R}^m_+$ [17, 18, 9, 12]. Because they are designed in the line of Benders decomposition and make use of the dual information of the second stage decision problem, we call them *Benders-dual cutting plane methods*, or *Benders-dual methods* for short, for 2-stage RO. We describe them as follows.

Note that once $\mathbf{y}$ and u are determined, the second stage decision problem is a LP problem in $\mathbf{x}$. We first take *the relatively complete recourse* assumption that this LP is feasible for any given $\mathbf{y}$ and u. Let π be its dual variables. Then, we obtain its dual problem and merge it with the maximization over the uncertainty set. As a result, we have the following problem, which is also the subproblem in Benders-dual method.

$$\begin{aligned} \mathbf{SP_1}:\ \mathcal{Q}(\mathbf{y}) = \quad & \max_{u,\pi}\ \pi^{\mathbf{T}}(\mathbf{h} - \mathbf{E}\mathbf{y} - \mathbf{M}u) \\ \text{s.t.} \quad & \mathbf{G}^{\mathbf{T}}\pi \leq \mathbf{b}^{\mathbf{T}} \\ & u \in \mathcal{U},\ \pi \geq \mathbf{0} \end{aligned} \tag{2}$$

Note that the resulting problem in (2) is a bilinear optimization problem, which is NP-hard in general. To solve it in 2-stage RO problems, several solution methods are developed for both relatively simple cardinality uncertainty sets and challenging polyhedral uncertainty sets, including outer approximation algorithm [9] and mixed integer linear reformulations [17, 18, 12, 11]. The first one can deal with a general polyhedral uncertainty set but only guarantees local optimum. The latter group fully utilizes the structure of the uncertainty set to convert the bilinear program into an equivalent mixed integer linear program. The conversion procedure relies on a critical property that if $\mathcal{U}$ is a polyhedron, u and π in one optimal solution will always take extreme points in their respective feasible sets.

In this paper, to deal with the general uncertainty set $\mathcal{U}$, we assume that an optimal solution of (2) for any $\mathcal{U}$ can be obtained from an oracle. Then, an optimal solution to 2-stage RO in (1) can be obtained by a two-level procedure that solves two problems, a master problem $\mathbf{MP_1}$ in (3) and a subproblem $\mathbf{SP_1}$ in (2), iteratively through a Benders style cutting plane procedure. Next, we describe the algorithm. In the description, LB denotes the lower bound, UB denotes the upper bound and k is the counter of Benders' iterations. Throughout this paper, $\epsilon \in \mathbf{R}_+$ denotes the tolerance of optimality.

Benders-dual Cutting Plane Algorithm

1. Set $LB = -\infty$, $UB = +\infty$ and $k = 0$.
2. Solve the following master problem.

$$\begin{aligned} \mathbf{MP_1}:\ \min_{\mathbf{y},\eta} \quad & \mathbf{c}\mathbf{y} + \eta \\ \text{s.t.} \quad & \mathbf{A}\mathbf{y} \geq \mathbf{d} \\ & \eta \geq \pi_l^*(\mathbf{h} - \mathbf{E}\mathbf{y} - \mathbf{M}u_l^*),\ \forall l \leq k \\ & \mathbf{y} \in \mathbf{S_y}, \eta \in \mathbf{R}. \end{aligned} \tag{3}$$

   Derive an optimal solution $(\mathbf{y}^*_{k+1}, \eta^*_{k+1})$ and update $LB = \mathbf{c}\mathbf{y}^*_{k+1} + \eta^*_{k+1}$.
3. Call the oracle to solve $\mathbf{SP_1}$ in (2), i.e. $\mathcal{Q}(\mathbf{y}^*_{k+1})$, and derive an optimal solution (u^*_{k+1}, π^*_{k+1}). Update $UB = \min\{UB, \mathbf{c}\mathbf{y}^*_{k+1} + \mathcal{Q}(\mathbf{y}^*_{k+1})\}$.
4. If $UB - LB \leq \epsilon$, return $\mathbf{y}^*_{k+1}$ and terminate. Otherwise, update $k = k + 1$ and go to Step 2 with addition of the following constraint to $\mathbf{MP_1}$.

$$\eta \geq \pi^{*\mathbf{T}}_{k+1}(\mathbf{h} - \mathbf{E}\mathbf{y} - \mathbf{M}u^*_{k+1}) \tag{4}$$

□

Note that, when $\mathcal{U}$ is a polyhedron, because u^*_k and π^*_k obtained in Step 3 are extreme points of their respective feasible sets, only a finite number of cuts in the form of (4) will be generated. Same observation holds when $\mathcal{U}$ is a finite discrete set. So, we have the following result regarding the algorithm's complexity.

Proposition 1. *Let p be the number of extreme points of $\mathcal{U}$ if it is a polyhedron or the cardinality of $\mathcal{U}$ if it is a finite discrete set. Let q be the number of extreme points of $\{\pi : \mathbf{G^T}\pi \leq \mathbf{b^T}, \pi \geq \mathbf{0}\}$. Then, Benders-dual cutting plane algorithm will generate an optimal solution to* (1) *in $O(pq)$ iterations.*

Compared with Benders Decomposition procedure [13], the generated cutting plane in (4) can be treated as a *optimality cut*. In the cases where the the relatively complete recourse assumption does not hold, Terry [16], Jiang et al. [12] discuss the *feasibility cut* issue.

3 The Primal Cut Algorithm

In this section, we present another cutting plane procedure to solve 2-stage RO problems. Because the generated cutting planes are defined by a set of created recourse decision variables that are primal to the decision maker, we call this procedure the *primal cut algorithm.*

To make our exposition simple, we first mention an observation when $\mathcal{U}$ is a finite discrete set. Let $\mathcal{U} = \{u_1, \ldots, u_r\}$ and $\{\mathbf{x}^1, \ldots, \mathbf{x}^r\}$ be the corresponding recourse decision variables. Then, the 2-stage RO in (1) can be reformulated as the following.

$$\min_{\mathbf{y}} \quad \mathbf{cy} + \eta \tag{5}$$

$$\text{s.t.} \quad \mathbf{Ay} \geq \mathbf{d} \tag{6}$$

$$\eta \geq \mathbf{bx}^l, \ l = 1, \ldots, r \tag{7}$$

$$\mathbf{Ey} + \mathbf{Gx}^l \geq \mathbf{h} - \mathbf{M}u_l, \ l = 1, \ldots, r \tag{8}$$

$$\mathbf{y} \in \mathbf{S_y}, \ \mathbf{x}^l \in \mathbf{S_x}, \ l = 1, \ldots, r \tag{9}$$

As a result, solving a 2-stage RO problem reduces to solve an equivalent (probably large-scale) mixed integer program, which is very close to a 2-stage stochastic programming (SP) model if the probability distribution over $\mathcal{U}$ is known. When the uncertainty set is large or is a polyhedron, enumerating all the possible uncertain scenarios in $\mathcal{U}$ is not feasible. In fact, constraints in (7) indicate that not all scenarios (and their corresponding variables and constraints) are necessary in defining the optimal value, or the optimal $\mathbf{y}^*$ of 2-stage RO consequently. Probably only a few important scenarios, a small subset of the uncertainty set, play the significant role in the formulation. Note that it is different from the 2-stage SP model where every single scenario in the scenario set actually contributes to the optimal value through its realization probability. So, it is necessary to consider the complete scenario set in deriving an optimal solution to the 2-stage SP model. With this observation in mind, we are motivated to design the *primal cut* algorithm, a constraint-and-column (variable) generation procedure, that just generates recourse decision variables and (7) - (8) for the significant scenarios on the fly.

Similar to Benders-dual method, our primal cut procedure is implemented in a master-subproblem framework. We assume that the previously mentioned oracle can solve the following max min problem, which is the subproblem in the procedure, for a given $\mathbf{y}$.

$$\begin{aligned} \mathbf{SP_2}: \quad & \mathcal{Q}(\mathbf{y}) = \max_{u \in \mathcal{U}} \min_{\mathbf{x}} \mathbf{bx} \\ \text{s.t.} \quad & \mathbf{Gx} \geq \mathbf{h} - \mathbf{Ey} - \mathbf{M}u \\ & \mathbf{x} \in \mathbf{S_x}. \end{aligned} \tag{10}$$

This oracle can either derive an optimal solution $(u^*, \mathbf{x}^*)$ with a finite optimal value $\mathcal{Q}(\mathbf{y})$ or identify some $u^* \in \mathcal{U}$ for which the second stage decision problem is infeasible. $\mathcal{Q}(\mathbf{y})$ in the latter case is set to $+\infty$ by convention. Next, we describe the algorithm.

Primal Cut Algorithm

1. Set $LB = -\infty$, $UB = +\infty$, $k = 0$ and $\mathbf{O} = \emptyset$.

2. Solve the following master problem.

$$
\begin{aligned}
\mathbf{MP_2}: \min_{\mathbf{y},\eta} \quad & \mathbf{c}\mathbf{y} + \eta \\
\text{s.t.} \quad & \mathbf{A}\mathbf{y} \geq \mathbf{d} \\
& \eta \geq \mathbf{b}\mathbf{x}^l, \ \forall l \in \mathbf{O} \\
& \mathbf{E}\mathbf{y} + \mathbf{G}\mathbf{x}^l \geq \mathbf{h} - \mathbf{M}u_l^*, \ \forall l \leq k \\
& \mathbf{y} \in \mathbf{S_y}, \ \eta \in \mathbf{R}, \ \mathbf{x}^l \in \mathbf{S_x} \ \forall l \leq k
\end{aligned} \tag{11}
$$

Derive an optimal solution $(\mathbf{y}_{k+1}^*, \eta_{k+1}^*, \mathbf{x}^{1*}, \ldots, \mathbf{x}^{k*})$ and update $LB = \mathbf{c}\mathbf{y}_{k+1}^* + \eta_{k+1}^*$.

3. Call the oracle to solve subproblem $\mathbf{SP_2}$ in (10) and update $UB = \min\{UB, \mathbf{c}\mathbf{y}_{k+1}^* + \mathcal{Q}(\mathbf{y}_{k+1}^*)\}$.

4. If $UB - LB \leq \epsilon$, return $\mathbf{y}_{k+1}^*$ and terminate. Otherwise, do

   (a) if $\mathcal{Q}(\mathbf{y}_{k+1}^*) < +\infty$, create variables $\mathbf{x}^{k+1}$ and add the following constraints

$$\eta \geq \mathbf{b}\mathbf{x}^{k+1} \tag{12}$$

$$\mathbf{E}\mathbf{y} + \mathbf{G}\mathbf{x}^{k+1} \geq \mathbf{h} - \mathbf{M}u_{k+1}^* \tag{13}$$

   to $\mathbf{MP_2}$ where u_{k+1}^* is the optimal scenario solving $\mathcal{Q}(\mathbf{y}_{k+1}^*)$. Update $k = k + 1$, $\mathbf{O} = \mathbf{O} \cup \{k + 1\}$ and go to Step 2.

   (b) if $\mathcal{Q}(\mathbf{y}_{k+1}^*) = +\infty$, create variables $\mathbf{x}^{k+1}$ and add the following constraints

$$\mathbf{E}\mathbf{y} + \mathbf{G}\mathbf{x}^{k+1} \geq \mathbf{h} - \mathbf{M}u_{k+1}^* \tag{14}$$

   to $\mathbf{MP_2}$ where u_{k+1}^* is the identified scenario for which $\mathcal{Q}(\mathbf{y}_{k+1}^*) = +\infty$. Update $k = k + 1$ and go to Step 2. □

Note that constraints (12)-(13) generated in Step 4.(a) serve as *optimality cuts* and constraints (14) generated in Step 4.(b) serve as *feasibility cuts*. In fact, because constraint (12) with $\mathbf{x}^{k+1}$ for an infeasible scenario is also valid, we can simply generate both (12) and (13) for any identified scenario. Therefore, it yields a unified approach to deal with optimality and feasibility. Also, from the above description, this algorithm dynamically constructs a formulation ($\mathbf{MP_2}$) with recourse decisions for each identified scenario, whose structure is very similar to that of a 2-stage SP model. Under the relatively complete recourse assumption, our algorithm can be completed in a finite number of iterations.

Proposition 2. *Let p be the number of extreme points of $\mathcal{U}$ if it is a polyhedron or the cardinality of $\mathcal{U}$ if it is a finite discrete set. Then, the primal cut algorithm will generate an optimal solution to* (1) *in $O(p)$ iterations.*

Comparing Benders-dual algorithm and the primal cut algorithm, we note some significant differences in the following aspects.

(i) Decision variables in the master problem. The primal cut algorithm increases the dimensionality of the solution space by introducing a set of new variables in each iteration while Benders-dual algorithm keeps working with the same set of variables.

(ii) The numbers and structures of the generated constraints. Both algorithms generate a constraint providing the lower bound to the worst case cost in each iteration. In the primal cut algorithm, such constraint is defined by primal recourse variables along with a set of constraints to restrict them. In Benders-dual algorithm, such constraint is constructed by the first stage primal decision variables with optimal values of the second stage dual variables. So, the primal cut algorithm generates a set of constraints while Benders-dual algorithm just creates a single constraint in each iteration.

(*iii*) Feasibility cut. The primal cut algorithm provides a general approach to deal with the feasibility issue of the second stage decision problem while current approaches for Benders-dual algorithm are problem specific.

(*iv*) The computational complexities. Compared with Benders-dual algorithm, the primal cut algorithm solves the master program with a larger number of variables and constraints. However, under the relatively complete recourse assumption, according to Proposition 1 and 2, the number of iterations in the primal cut algorithm is reduced by the order of $O(q)$, if the second stage decision problem is a LP. Actually, as the number of extreme points is exponential with respect to numbers of variables and constraints (in the second stage), such a reduction is very significant. The computational study on a power system scheduling problem [18] and on a location-transportation problem presented in Section 4 confirms this point.

(*v*) Solution capability. Different from Benders-dual algorithm that requires the second stage problem to be a LP problem, the primal cut algorithm is indifferent to the variable types in the second stage.

As we mentioned, the primal cut algorithm converts a 2-stage RO problem into a mixed integer linear formulation that has a structure very close to a 2-stage SP problem. We believe that the connection between 2-stage RO and 2-stage SP revealed in the primal cut algorithm opens a few opportunities to perform the analytical research on these two subjects in both theoretical and computational aspects. The cross-subject study may greatly improve our understanding and solution ability on these two subjects.

In the next section, we apply the primal cut method to solve the location-transportation problem that has been modeled and studied with a 2-stage robust optimization formulation by Atamturk and Zhang [1]. It is also recently investigated by Gabrel et al. [11] with a Bender-dual implementation.

4 A Case Study: Robust Location-transportation Problem

We consider the following location-transportation problem. To supply some commodity to customers, it will be first stored at m potential facilities, and then be transported to n customers. The fixed cost of building facilities at site i is f_i and unit capacity cost is a_i for $i = 1, \dots, m$. The demand is d_j for $j = 1, \dots, n$ and the unit transportation cost between i and j is c_{ij} for $i - j$ pair. The maximal allowable capacity of the facility at site i is K_i and we have $\sum_i K_i \geq \sum_j d_j$ to ensure feasibility. Let $y_i \in \{0, 1\}$ be the facility location variable, $z_i \in \mathbf{R}_+$ be the capacity variable, $x_{ij} \in \mathbf{R}_+$ be the transportation variable. Then, the nominal formulation of this location-transportation problem is as follows:

$$\min_{\mathbf{y},\mathbf{z},\mathbf{x}} \sum_i f_i y_i + \sum_i a_i z_i + \sum_i \sum_j c_{ij} x_{ij} \tag{15}$$

$$\text{s.t. } z_i \leq K_i y_i, \forall i \tag{16}$$

$$\sum_j x_{ij} \leq z_i, \forall i \tag{17}$$

$$\sum_i x_{ij} \geq d_j, \forall j \tag{18}$$

$$y_i \in \{0, 1\},\ z_i \geq 0\ \forall i, \quad x_{ij} \geq 0\ \forall i, j \tag{19}$$

The objective function in (15) is to minimize the overall cost, including the fixed cost, capacity cost and transportation cost. Constraints in (16) and (17) require that capacity can only be installed in a site with a facility built and the supply cannot exceed the capacity. Constraints in (18) guarantee that the demand is satisfied.

In practice, the demand is unknown before any facility is built and capacity is installed. Its uncertainty can be captured by a polyhedral set

$$\mathbf{D} = \{\mathbf{d} : d_j = \underline{d}_j + g_j \tilde{d}_j,\ g_j \in [0,1],\ \sum_j g_j \leq \Gamma,\ j = 1, \ldots, n\} \tag{20}$$

where $\underline{d}_j$ is the basic demand, $\tilde{d}_j$ is the maximal deviation, and Γ, a predefined integer value, is introduced to define the constraint of *budget of uncertainty* to control the conservative level. Such way to define the uncertainty set is very popular among current applications of 2-stage RO, see [8, 12, 1] for examples.

With the uncertainty set on the demand, the whole decision process can be decomposed into two stages that are implemented before and after the realization of demands. So, in the first stage, facility and capacity will be determined and established; in the second stage, transportation will be determined to meet customers' demands. To minimize the total cost in the worst situation, a 2-stage robust counterpart of the nominal formulation can be obtained as follows. Similar to the nominal model, we assume that $\sum_i K_i \geq \max\{\sum_j d_j : \mathbf{d} \in \mathbf{D}\}$ to ensure the existence of feasible solutions.

$$\begin{aligned}
\min_{(\mathbf{y},\mathbf{z}) \in \mathbf{S_y}} \quad & \sum_i f_i y_i + \sum_i a_i z_i + \max_{\mathbf{d} \in \mathbf{D}} \min_{\mathbf{x} \in \mathbf{S_x}} \sum_i \sum_j c_{ij} x_{ij} \\
\text{s.t.} \quad & \mathbf{S_y} = \{(\mathbf{y}, \mathbf{z}) \in \{0,1\}^m \times \mathbf{R}_+^m : (16)\} \\
& \mathbf{S_x} = \{\mathbf{x} \in \mathbf{R}_+^{m \times n} : (17-18)\}
\end{aligned}$$

4.1 Primal Cut Algorithm

As the general description of the algorithm has been presented in Section 3, we focus on the solution method for $\mathbf{SP_2}$ of the location-transportation problem. We also provide the dynamically constructed master problems for both the primal cut and Benders-dual algorithms for comparison.

We first consider the infeasibility detection for the aforementioned 2-stage model. Note that the feasibility issue happens only if the installed total capacity is strictly less than the total demand. So, it is sufficient to consider a scenario with the largest total demand. It can be easily obtained by first sorting $\underline{d}_j + \tilde{d}_j$ for all j; then identifying the largest Γ out of them and setting their d_j to $\underline{d}_j + \tilde{d}_j$ and setting remaining ones to $\underline{d}_j$. In our implementation, we generate constraints (12) and (13) for that scenario.

To generate general optimality cuts, for a fixed $(\mathbf{y}^*, \mathbf{z}^*)$, we obtain the dual problem of the second stage decision problem and combine it with the maximization part. The combined problem is

$$\begin{aligned}
\max_{\mathbf{d},\lambda,\pi} & \sum_j \underline{d}_j \lambda_j + \sum_j \tilde{d}_j \lambda_j g_j - \sum_i \pi_i z_i^* \\
\text{st. } & \lambda_j - \pi_i \leq c_{ij}, \forall i, j \\
& \sum_j g_j \leq \Gamma \\
& 0 \leq g_j \leq 1,\ \forall j \\
& \lambda_j \geq 0,\ \forall j,\ \pi_i \geq 0,\ \forall i
\end{aligned}$$

It is easy to see that we always have $g_j \in \{0,1\}$ for all j in one optimal solution. So, we can apply the standard linearization technique to obtained the following mixed integer linear program. Note that w_j is introduced to replace $\lambda_j g_j$ and M' is a big number. The same strategy is also used in [11].

$$
\begin{aligned}
\max_{\mathbf{d},\lambda,\pi,\mathbf{w}} \quad & \sum_j \underline{d}_j \lambda_j + \sum_j \tilde{d}_j w_j - \sum_i z_i^* \pi_i \\
\text{st. } & \lambda_j - \pi_i \le c_{ij}, \forall i, j \\
& \sum_j g_j \le \Gamma \\
& w_j \le \lambda_j, \forall j \\
& w_j \le M' g_j, \forall j \\
& w_j \ge \lambda_j - M'(1 - g_j), \forall j \\
& \lambda_j,\ w_j \ge 0,\ g_j \in \{0,1\}\ \forall j,\ \pi_i \ge 0,\ \forall i
\end{aligned}
\tag{21}
$$

Then, a mixed integer program solver can be used as the oracle to derive an optimal solution to the above problem. Assuming that an optimal solution, $(\mathbf{d}^k, \lambda^k, \pi^k, \mathbf{w}^k)$, is derived by a solver in the k-th iteration. The master problem defined in (11) in the k-th iteration will be of the following form where $\mathbf{x}^k$ are the created recourse variables in iteration k.

$$
\begin{aligned}
\min_{\mathbf{y},\mathbf{z},\mathbf{x},\eta} \quad & \sum_i f_i y_i + \sum_i a_i z_i + \eta \\
\text{s.t. } & (16) \\
& \eta \ge \sum_i \sum_j c_{ij} x_{ij}^l \ \forall i, j, \text{and } 1 \le l \le k \\
& \sum_i x_{ij}^l \ge d_j^l, \forall j, \text{and } 1 \le l \le k \\
& \sum_j x_{ij}^l \le z_i, \forall i, \text{and } 1 \le l \le k \\
& z_i \ge 0, y_i \in \{0,1\}, \forall i,\ \eta \in \mathbf{R},\ x_{ji}^l \ge 0, \forall i, j, \text{and } 1 \le l \le k
\end{aligned}
$$

If we adopt Benders-dual algorithm, we have the following master problem in the k'-th iteration where k' is the counter of Benders iterations.

$$
\begin{aligned}
\min_{\mathbf{y},\mathbf{z}} & \sum_i f_i y_i + \sum_i a_i z_i + \eta \\
\text{s.t. } & (16) \\
& \eta \ge \sum_j \lambda_j^l d_j^l - \sum_i \pi_i^l z_i, 1 \le l \le k' \\
& z_i \ge 0, y_i \in \{0,1\}, \forall i,\ \eta \in \mathbf{R}
\end{aligned}
$$

In next section, we apply these two algorithms to solve the same set of instances. Our numerical results demonstrate the effectiveness of the primal cut algorithm.

4.2 Computation Results

To provide a basis for comparison, our numerical study is performed on instances that are randomly generated in a fashion used in [11]. The demand $\underline{d}_j$ is obtained from $[10, 500]$, the deviation $\tilde{d}_j$ is $\alpha \underline{d}_j$ with $\alpha \in [0.1, 0.5]$, the maximal allowable capacity K_i is drawn from $[200, 700]$ with the feasibility guarantee, the fixed cost is generated from $[100, 1000]$, the unit capacity cost is selected from $[10, 100]$, and the transportation cost is in interval $[1, 1000]$. With the aforementioned setup, 20 instances are randomly generated with 10 for the case

$m \times n = 30 \times 30$ and 10 for the case $m \times n = 70 \times 70$. Also, to investigate the impact of Γ, we set its value to $10\%, 20\%, \ldots, 100\%$ of m. So, overall, we have two sets of 100 testing problems. We also use the method presented in [11] to set values for M' in (21). In all of our experiments, CPLEX 12.1 is used as the solver to the master problem and the oracle to the linearized subproblem. For both the master and the subproblems, the optimality tolerance is set to 10^{-4}. We implement both primal cut and Benders-dual algorithms in C++ and perform computation experiments on a desktop Dell OPTIPLEX 760 (Intel Core 2 Duo CPU, 3.0GHz, 3.25GB of RAM) under Windows 7 environment.

All the original computation results are presented in Table A-1–A-4 in the appendix. We summarize those results in Table 1 and Table 2 where the average performance over every 10 instances under different Γ is displayed. In those tables, *BD* stands for Benders-dual algorithm, *PC* stands for the primal cut algorithm, *Ratio* represents the ratio of their performance.

Our results on Benders-dual algorithm generally agree with those presented in [11]. The computational time for $\Gamma \in [20\%, 80\%]$ is typically more than that of other cases. This is different from the results presented in [1], where the computational times are negatively correlated with Γ. One explanation is that the problem is solved approximately in [1] while exact solutions are derived by Benders-dual algorithm. For the primal cut algorithm, we first observe that it performs an order of magnitude faster than Benders-dual algorithm in all experiments. Such improvement is more significant when the problem size is large. Besides the reduction in the computational time, it generally can complete within a small number of iterations, very different from Benders-dual method that may need hundreds of iterations. We believe that the performance improvement could be explained by two reasons. First, the primal cut algorithm strictly identifies another significant scenario by its subproblem in every iteration, which drastically increases the convergence rate. On the contrary, Benders-dual method use many iterations to obtain the value function for a particular first stage decision under the same scenario. Second, the primal cut algorithm produces a (large-scale) mixed integer program as its master problem, which largely keeps the network structure of the nominal model. So, the solver can fully make use of that structure in their computation while the generated cutting planes by Benders-dual method prevent it from identifying and utilizing that structure. The latter point is also supported by the fact that the reduction in the computational time is more significant than the reduction in the number of iterations.

We also observe that, unlike the computation time, the number of iterations in the primal cut algorithm is insensitive to the problem sizes. A similar result is also found in solving robust power system scheduling problems [18]. Those results indicate that the number of significant scenario defining the worst case cost is relatively stable and small, regardless of the problem size. So, a method to quickly identify the significant scenarios, along with an efficient algorithm for the resulting master problem, can greatly improve our solution capability on 2-stage RO problems. Actually, it would be interesting to implement decomposition methods used in stochastic programming to the master problem, which has a clear decomposable structure, to solve large-scale instances. Certainly, this observation is related to the description of the uncertainty set. We will consider other applications with more general uncertainty sets in future.

5 Conclusion

In this paper, we present a cutting plane algorithm, the primal cut algorithm, in a general setting to solve 2-stage robust optimization problems. Different from existing Benders-dual cutting plane methods, it is a constraint-and-column generation method. It can solve general problems, does not restrict variable types, and provide a unified approach to deal with feasibility and optimality issues. In particular, this algorithm can derive an optimal solution with a much less computational expense. From our systematical study on 2-stage robust location-transportation problem, as well as a preliminary study on power system scheduling problems in [18], we observe that it performs an order of magnitude faster than Benders-dual cutting plane algorithm.

Besides the computational improvement, we mention that the primal cut algorithm reveals a linkage between 2-stage robust optimization and 2-stage stochastic programming. It may open a few opportunities in studying these two subjects, from both methodological and computational aspects. One possible direction is how to take advantage of existing results on one subject to advance the study on the other one. As both problems are NP-hard in general, such a cross-subject study may greatly improve our understanding and solution ability on them.

References

[1] A. Atamturk and M. Zhang. Two-stage robust network flow and design under demand uncertainty. *Operations Research*, 55(4):662–673, 2007.

[2] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23(4):769–805, 1998.

[3] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Operations Research Letters*, 25(1):1–14, 1999.

[4] A. Ben-Tal and A. Nemirovski. Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical Programming*, 88(3):411–424, 2000.

[5] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Mathematical Programming*, 99(2):351–376, 2004.

[6] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming*, 98(1):49–71, 2003.

[7] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[8] D. Bertsimas, D. B. Brown, and C. Caramanis. Theory and applications of robust optimization. *to appear in SIAM Review*, 2010.

[9] D. Bertsimas, E. Litvinov, X.A. Sun, J. Zhao, and T. Zheng. Adaptive robust optimization for the security constrained unit commitment problem. Technical report, 2011.

[10] L. El Ghaoui, F. Oustry, and H. Lebret. Robust solutions to uncertain semidefinite programs. *SIAM journal of optimization*, 9:33–52, 1998.

[11] V. Gabrel, M. Lacroix, C. Murat, and N. Remli. Robust location transportation problems under uncertain demands. Technical report, submitted to Discrete Applied Mathematics, Lamsade, Universite Paris-Dauphine, 2011.

[12] R. Jiang, M. Zhang, G. Li, and Y. Guan. Two-stage robust power grid optimization problem. Technical report, 2011.

[13] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* Wiley, 1988.

[14] F. Ordonez and J. Zhao. Robust capacity expansion of network flows. *Networks*, 50 (2):136–145, 2007.

[15] A. Takeda, S. Taguchi, and RH Tutuncu. Adjustable robust optimization models for a nonlinear two-period system. *Journal of Optimization Theory and Applications*, 136 (2):275–295, 2008.

[16] T.L. Terry. *Robust Linear Optimization with Recourse: Solution Methods and Other Properties*. PhD thesis, University of Michigan, 2009.

[17] A. Thiele, T. Terry, and M. Epelman. Robust linear optimization with recourse. Technical report, available in *optimization-online*, 2010.

[18] L. Zhao and B. Zeng. Robust unit commitment problem with demand response and wind energy. Technical report, available in *optimization-online*, University of South Florida, 2010.

Table 1: Performance of Benders-dual and Primal Cut Algorithms on 30×30 Instances

Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Avg.
BD (CPU sec.)	16.77	18.05	21.21	17.62	18.47	19.09	19.80	21.69	22.07	19.15	19.39
PC (CPU sec.)	1.17	1.89	2.28	1.99	1.33	1.61	1.41	1.68	1.08	0.34	1.48
Ratio	**14.31**	**9.57**	**9.31**	**8.86**	**13.89**	**11.83**	**14.07**	**12.92**	**20.51**	**57.08**	**17.24**
BD (# iter.)	62.70	58.70	58.50	47.30	46.20	43.30	43.20	44.70	43.80	40.60	48.90
PC (# iter.)	4.20	5.70	6.50	5.40	5.00	5.60	4.60	5.40	4.10	2.00	4.85
Ratio	**14.93**	**10.30**	**9.00**	**8.76**	**9.24**	**7.73**	**9.39**	**8.28**	**10.68**	**20.30**	**10.86**

Table 2: Performance of Benders-dual and Primal Cut Algorithms on 70×70 Instances

Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Avg.
BD (CPU sec.)	689.32	1581.90	1725.21	1391.96	837.04	703.60	657.46	548.40	396.61	205.31	873.68
PC (CPU sec.)	22.57	26.52	76.55	70.19	39.76	50.76	12.03	11.59	6.59	1.15	31.77
Ratio	**30.54**	**59.65**	**22.54**	**19.83**	**21.05**	**13.86**	**54.67**	**47.31**	**60.18**	**178.71**	**50.84**
BD (# iter.)	198.50	153.50	137.70	122.70	143.00	133.20	138.70	131.40	139.20	133.30	143.12
PC (# iter.)	7.00	5.30	5.40	5.10	5.20	5.80	4.50	5.00	4.60	2.00	4.99
Ratio	**28.36**	**28.96**	**25.50**	**24.06**	**27.50**	**22.97**	**30.82**	**26.28**	**30.26**	**66.65**	**31.14**

Appendix

A-1 Computational Results for 30×30 Instances

Table A-1: Computation Time (sec.) of Primal Cut Algorithm

ID\Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
1	0.57	0.96	1.54	2.21	1.26	1.49	1.23	1.33	0.49	0.34
2	0.64	1.17	1.73	0.88	1.16	1.66	0.98	2.24	0.90	0.31
3	1.38	3.32	2.98	2.02	1.50	1.71	2.99	3.21	2.67	0.36
4	1.20	2.03	2.21	2.36	1.09	2.11	3.00	3.05	1.25	0.32
5	0.61	1.04	0.79	1.08	1.80	2.22	0.82	1.73	0.67	0.35
6	2.16	5.13	7.48	7.34	2.36	1.10	0.98	0.71	1.01	0.35
7	0.64	0.83	0.86	1.04	1.18	1.89	0.57	0.57	0.82	0.33
8	0.88	2.44	2.34	1.63	0.86	0.71	0.46	0.67	0.63	0.32
9	2.05	1.19	2.07	0.90	0.64	0.68	0.40	0.83	0.68	0.33
10	1.59	0.77	0.79	0.43	1.45	2.57	2.66	2.45	1.65	0.36
Avg.	1.17	1.89	2.28	1.99	1.33	1.61	1.41	1.68	1.08	0.34

Table A-2: Computation Time (sec.) of Benders-dual Algorithm

ID\Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
1	11.51	14.53	17.57	15.97	13.21	11.47	12.41	13.61	9.58	7.78
2	19.23	13.80	18.82	12.49	16.68	17.33	14.77	21.70	17.69	13.45
3	38.77	41.20	51.69	36.19	47.52	57.77	56.16	68.43	76.89	66.09
4	11.70	13.97	14.49	13.51	13.30	13.19	15.67	12.57	8.63	7.49
5	19.82	17.71	17.87	16.63	15.64	14.04	15.97	16.45	15.44	12.94
6	12.39	20.87	27.84	21.97	23.02	20.58	20.66	17.07	17.19	19.43
7	11.06	16.87	16.87	14.51	17.22	17.75	18.34	26.37	22.39	21.51
8	10.59	14.04	16.31	13.45	10.14	7.17	7.51	5.50	6.37	7.01
9	11.83	11.30	12.95	8.94	4.28	4.12	2.47	1.85	2.42	3.05
10	20.84	16.24	17.69	22.54	23.68	27.47	34.01	33.39	44.15	32.72
Avg.	16.77	18.05	21.21	17.62	18.47	19.09	19.80	21.69	22.07	19.15

Table A-3: Number of Iterations in Primal Cut Algorithm

ID\Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
1	3	4	6	6	5	6	5	5	2	2
2	3	5	7	4	5	6	4	8	4	2
3	5	9	9	6	5	6	8	8	7	2
4	5	6	6	7	4	5	6	7	5	2
5	3	5	4	5	7	7	4	6	3	2
6	6	9	13	12	7	4	4	3	4	2
7	3	4	4	4	5	7	3	3	4	2
8	2	7	7	5	3	3	2	3	3	2
9	6	4	5	3	3	3	2	4	3	2
10	6	4	4	2	6	9	8	7	6	2
Avg.	4.2	5.7	6.5	5.4	5	5.6	4.6	5.4	4.1	2

Table A-4: Number of Iterations in Benders-dual Algorithm

ID\Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
1	50	56	56	40	39	33	33	33	25	22
2	77	50	60	38	44	40	35	52	43	36
3	116	107	113	81	98	96	84	95	108	94
4	49	40	39	37	34	30	33	26	22	22
5	74	67	61	52	38	36	40	42	38	33
6	49	69	62	49	55	53	54	44	41	49
7	48	57	58	47	50	48	51	66	54	52
8	40	48	51	35	29	22	23	17	19	23
9	42	30	26	25	14	14	9	7	9	13
10	82	63	59	69	61	61	70	65	79	62
Avg.	62.7	58.7	58.5	47.3	46.2	43.3	43.2	44.7	43.8	40.6

A-2 Computational Results for 70×70 Instances

Table A-5: Computation Time (sec.) of Primal Cut Algorithm

ID\Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
1	22.67	54.57	53.98	14.69	12.61	13.52	12.14	7.58	6.08	1.14
2	33.00	80.38	62.90	19.43	48.44	331.60	9.71	9.56	4.19	1.19
3	38.74	28.05	15.82	52.30	189.82	43.56	18.73	7.91	4.92	1.16
4	70.97	16.08	9.97	4.85	8.55	10.61	4.51	9.12	10.39	1.22
5	16.56	5.88	21.79	60.19	21.57	7.21	4.61	5.71	4.30	1.10
6	3.07	6.90	12.17	23.42	29.33	29.98	13.43	13.83	6.02	1.11
7	1.27	3.92	11.08	12.75	14.80	10.26	6.92	6.34	3.93	1.05
8	6.47	26.26	8.70	4.62	22.83	16.70	21.13	24.35	20.73	1.17
9	12.70	29.34	563.15	496.86	28.87	23.54	18.46	27.40	1.88	1.22
10	20.24	13.81	5.97	12.84	20.74	20.56	10.61	4.14	3.48	1.12
Avg.	22.57	26.52	76.55	70.19	39.76	50.76	12.03	11.59	6.59	1.15

Table A-6: Computation Time (sec.) of Benders-dual Algorithm

ID\Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
1.0	486.1	660.8	800.7	218.9	235.7	230.7	141.3	133.7	103.9	68.3
2.0	1083.7	8798.8	7863.6	2429.9	1124.3	1246.4	593.0	129.1	104.2	119.5
3.0	1960.4	2751.9	1768.3	1644.5	1095.6	1063.1	1239.5	1163.4	1263.8	449.0
4.0	709.9	385.2	286.3	278.5	290.0	371.4	311.3	381.5	467.5	444.1
5.0	458.7	230.0	404.8	400.7	910.2	505.5	440.8	625.0	189.7	208.2
6.0	89.3	87.6	133.0	593.2	1014.7	892.3	980.2	749.1	523.0	72.4
7.0	345.4	364.3	462.0	403.9	697.5	619.2	656.5	236.0	139.6	151.1
8.0	194.2	490.0	358.6	263.3	924.4	922.2	1017.5	1116.8	282.3	169.8
9.0	1007.1	1592.0	4767.6	7235.0	1712.4	802.4	539.9	505.6	301.7	279.1
10.0	558.5	458.4	407.2	451.7	365.7	382.6	654.5	443.8	590.5	91.7
Avg.	689.3	1581.9	1725.2	1392.0	837.0	703.6	657.5	548.4	396.6	205.3

Table A-7: Number of Iterations in Primal Cut Algorithm

ID\Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
1	5	4	6	4	3	4	4	4	5	2
2	8	5	5	3	4	12	6	5	4	2
3	9	7	4	5	7	7	4	4	4	2
4	15	7	6	4	6	6	4	5	7	2
5	7	5	4	6	5	4	4	4	4	2
6	3	4	5	6	6	6	4	6	5	2
7	2	3	6	5	5	4	4	4	4	2
8	5	6	4	3	5	4	6	6	7	2
9	7	7	11	10	6	6	5	8	2	2
10	9	5	3	5	5	5	4	4	4	2
Avg.	7	5.3	5.4	5.1	5.2	5.8	4.5	5	4.6	2

Table A-8: Number of Iterations in Benders-dual Algorithm

ID\Γ	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
1	97	77	81	59	51	58	46	44	42	34
2	254	147	156	92	107	121	130	108	121	127
3	321	353	230	158	196	197	214	224	257	246
4	247	139	90	89	96	109	81	85	109	108
5	178	86	88	72	146	113	131	138	149	154
6	66	45	65	127	168	134	182	149	135	116
7	180	150	167	118	160	138	149	157	147	141
8	150	151	125	72	166	146	167	172	163	154
9	276	222	263	326	254	226	154	136	125	122
10	216	165	112	114	86	90	133	101	144	131
Avg.	198.5	153.5	137.7	122.7	143	133.2	138.7	131.4	139.2	133.3