Distributionally Robust Disaster Relief Planning under the Wasserstein Set

Mohamed El Tonbari George Nemhauser Alejandro Toriello
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, Georgia, USA 30332

Abstract

We study a two-stage natural disaster management problem modeled as a stochastic program, where the first stage consists of a facility location problem, deciding where to open facilities and pre-allocate resources such as medical and food kits, and the second stage is a fixed-charge transportation problem, routing resources to affected areas after observing a disaster. Our model has binary variables present in both stages. Due to the lack of data, classical stochastic programming approaches may be ill-suited, and we propose a two-stage distributionally robust formulation with a Wasserstein ambiguity set, where we consider distributions consistent with historical data and a tunable parameter to control the level of risk aversion. We develop a tailored column-and-constraint generation (CCG) algorithm to solve an extensive reformulation, where scenarios are iteratively generated. We handle the presence of binary variables in the second stage by leveraging the structure of our support set and second-stage problem, and provide conditions under which the optimal value of the latter is concave with respect to the intensity of the disaster, leading to an efficient scenario generation procedure. We also show that our results extend to the case where the second stage is a fixed-charge network flow problem. We perform extensive computational experiments demonstrating the computational advantage of our method over classical CCG implementations on synthetic instances, and illustrate the benefits of our approach on a popular case study from the literature of hurricane threats on the Gulf of Mexico in the United States.

1 Introduction

Natural disasters can have a significant impact on the lives of many people. It is estimated that about two major hurricanes make landfall around the Gulf or Atlantic coast of the United States every three years which can lead to very large costs [9]. Hurricane Katrina and Sandy were among the costliest hurricanes, leading to over a hundred billion dollars in costs each. In natural disaster management, planners seek to alleviate the consequences of a disaster ahead of time. Common long-term planning involves readiness, such as pre-positioning resources and potentially deciding on the location of new facilities to more easily serve affected areas, and an efficient response after a disaster [27]. Other decisions and challenges include evacuation planning and its uncertainties [2, 12]. Although optimization is a powerful tool to solve such decision-making problems, it can lead to arbitrarily poor solutions if one assumes all parameters are known in advance. There is a natural sequence of decisions in disaster relief operations, namely before and after a disaster occurs. To account for the uncertainty in the intensity or location of the disaster, such a decision-making problem naturally lends itself to a two-stage stochastic programming model, where decisions are split into two stages, before and after observing the uncertainty. We are specifically interested in a two-stage network design problem arising in natural disaster management, where the first stage is a facility location problem, deciding where to open facilities and pre-allocate resources such as medical or food kits, and the second stage is a fixed-charge transportation problem, routing resources from facilities to affected areas after a disaster.

In general two-stage stochastic programs, the goal is to minimize the sum of the first stage cost and the expected second-stage cost, where the expectation is taken over the probability distribution $\mathbb{P}$ of the underlying uncertainty. Its general form can be written as

$$\min_{\boldsymbol{x} \in X} \quad \boldsymbol{c}^\top \boldsymbol{x} + \mathbb{E}_{\mathbb{P}}\left[Q(\boldsymbol{x}, \boldsymbol{\xi})\right] \tag{SP}$$

where $\boldsymbol{x}$ represents the first-stage decisions of opening a facility and pre-allocating resources, $\boldsymbol{\xi}$ is a random vector belonging to a support set Ξ with associated probability distribution $\mathbb{P}$ representing uncertain demands of resources and/or costs resulting from a disaster, and $Q(\boldsymbol{x}, \boldsymbol{\xi})$ is the optimal value of the second-stage routing problem given first-stage decisions $\boldsymbol{x}$ and a realization of the random vector $\boldsymbol{\xi}$. We also refer to $Q(\boldsymbol{x}, \boldsymbol{\xi})$ as the second-stage value function.

As in many applications, the underlying distribution $\mathbb{P}$ is not known, and historical samples are often used to approximate the expected value and the empirical average is minimized instead. This is known as the Sample Average Approximation (SAA) method [33, 21]. Alternatively, a more conservative approach is to minimize the worst-case cost with respect to the random variables, where a distribution need not be known. This is referred to as robust optimization (RO) [4]. Although it can be shown that SAA converges to the optimal solution of (SP) under certain conditions as the number of samples goes to infinity, given a finite training dataset, the SAA solution can be too optimistic and lead to poor out-of-sample performance [14, 8]. In other words, SAA can overfit the data. This is especially true in applications such as natural disaster management, where data is scarce or cannot be generated. On the other hand, robust optimization can lead to a very conservative and high cost solution.

The great majority of the literature has been focused on solving either a stochastic program or robust optimization model for the problem of pre-allocating and routing resources to affected areas after a disaster. Surveys of recent work on disaster relief operations can be found in [29] and [17], where the latter is focused on two-stage stochastic programming solutions. A case study on hurricane threats on the coastal states of the United States along the Gulf of Mexico and the Atlantic Ocean is presented and solved in [28], where scenarios are constructed and their probabilities estimated based on historical data. The same case study has been used in [30, 32, 34]. In [32], a two-stage robust model is solved; in [2] a two-stage evacuation planning model is solved, and in [12], uncertainties related to evacuation are considered in developing a robust model to determine supply distribution.

Natural disasters occur relatively rarely, and the scarcity of data makes it difficult to reliably estimate the underlying probability distribution of the uncertainty [30, 32], especially in developing countries where infrastructure to collect data is limited. As a result, approaches like SAA that attempt to estimate the probability distribution from historical data can lead to poor out-of-sample performance. It is only very recently that attention has been drawn to using a distributionally robust optimization (DRO) paradigm. In [34], the authors consider a two-stage DRO model under a box and polyhedral ambiguity set. In [30, 38], the authors consider a two-stage DRO model under a moment-based ambiguity set. Close in spirit to our work are [40] and [25], where the choice of the ambiguity set is the Wasserstein metric. Although not a natural disaster application, the former solves a continuous chance-constrained DRO model to determine emergency medical system locations; the model is reformulated as a MIP to handle the chance constraints. The latter considers a DRO model with a cluster-based ambiguity set using a Wasserstein metric, where the decision

is simplified to only picking facility locations; the model is unconstrained, with the only restriction being the set of possible facility locations, and demand locations are random and served by the closest available facility.

DRO permits us to be risk averse towards the empirical distribution obtained from historical data. In two-stage DRO (TSDRO), we seek to minimize the first-stage cost and the worst-case expected value of the second stage-cost with respect to probability distributions belonging to an ambiguity set. TSDRO problems are of the form

$$\min_{\boldsymbol{x} \in X} \quad \boldsymbol{c}^\top \boldsymbol{x} + \max_{\mathbb{P} \in \mathscr{P}} \mathbb{E}_{\mathbb{P}} \left[Q(\boldsymbol{x}, \boldsymbol{\xi}) \right], \tag{DRO}$$

where $\mathscr{P}$ is the ambiguity set.

1.1 Brief DRO Literature Review

There is a wide variety of ambiguity sets that have been studied in the literature. Ambiguity sets can be roughly split into two categories: moment-based ambiguity sets and balls in the space of probability distributions centered at some nominal distribution. In the former, the ambiguity sets include constraints on the moments of the distribution [11, 35, 39]. Given point estimates of the moments, the ambiguity set can include constraints such that moments match the estimates or are some distance away from the estimates to account for the uncertainty in the latter. For example, in [13], the authors propose an ambiguity set which constrains the mean and the second order moments to lie in an ellipsoid centered at the estimate of the mean and in the intersection of two positive semi-definite cones, respectively. Moment-based ambiguity sets have gained popularity for their computational benefits [14]. On the other hand, balls centered at a nominal distribution depend on a metric defined on probability distributions. Popular metrics include the ϕ-divergence [6, 26], the Kullback-Leibler divergence [10, 22], which is a special case of the ϕ-divergence, and the Wasserstein metric [1, 14, 16, 23, 36].

The Wasserstein ambiguity set has recently received significant attention due to its finite-sample guarantees and out-of-sample performance. It has been shown that (DRO) can be reformulated as a tractable convex program for various classes of loss functions and support sets, typically convex and continuous, respectively [14, 16]. Esfahani and Kuhn [14] discuss finite-sample guarantees of using DRO under the Wasserstein ambiguity set, and illustrate the benefits of DRO over SAA and robust optimization in out-of-sample performance. Xie [37] shows conditions under which tractable reformulations of TSDRO problems

with a convex second stage exist for zero-one uncertainties under the type-∞ Wasserstein set.

Limited work explores the case where integer variables are present in the second stage. In [41], a Benders decomposition approach is used to solve a two-stage distributionally robust unit-commitment problem with binary variables over a Wasserstein ball. Kim [20] extends the classical dual decomposition method to DRO models under the Wasserstein set to solve the Lagrangian dual. In [1], an extension of the integer L-shaped method is proposed to solve TSDRO models with binary first-stage decisions and mixed-binary second-stage decisions under the Wasserstein set. In two-stage robust optimization, the K-adaptability algorithm was developed to handle second-stage binary variables in [19] and second-stage mixed-integer variables in [31]. The K-adaptability algorithm was extended to two-stage distributionally robust optimization with moment-based ambiguity sets in [18].

The robust optimization literature is rich in tackling two-stage problems. A common approximation is to assume a parametrized policy for the second stage, where the latter is restricted to a specific structure [3]. Such methods tend to lead to a more tractable model, and are even optimal in certain applications [5]. Examples of such techniques are affine policies, where the second-stage variables are restricted to an affine policy with respect to the uncertainty [5, 7].

1.2 Wasserstein Ambiguity Set

The Wasserstein set is a ball in the space of probability distributions defined on the Wasserstein metric and centered at a nominal distribution, typically an empirical distribution obtained from a set of samples. It is desirable to have an ambiguity set that is rich enough to include the true distribution, but not too large so as to exclude pathological ones. A discussion of shortcomings of the ϕ-divergence metric that are not shared by the Wasserstein metric can be found in [16], where the former can exclude the true distribution while including pathological ones. Moreover, when using the Kullback-Leibler divergence, for example, the worst-case distribution can only have a support on scenarios with non-zero probability in the nominal distribution. In other words, if the nominal distribution is the empirical distribution obtained from historical data, the worst-case distribution can only have non-zero probability for observed scenarios, thus not protecting against unobserved scenarios. On the other hand, the Wasserstein set includes distributions that can assign non-zero probability to any scenario. This is especially advantageous in applications such as natural disaster management where we wish to protect ourselves against unobserved disaster scenarios. In this sense, relative to [30, 34], which also consider DRO in disaster relief operations, our work considers

a potentially richer ambiguity set that makes no assumptions on the underlying distribution. By using the Wasserstein ambiguity set, we circumvent the need to estimate moments of the distribution, which can be non-trivial with limited data. Finally, we note that our DRO model is further complicated by the presence of binary variables in the second stage.

1.3 Contribution

In this paper, we consider a two-stage distributionally robust model under a Wasserstein ambiguity set, where we allow binary variables to be present in the second stage. While this provides much greater modeling power, the presence of binary variables and the Wasserstein ambiguity set significantly complicates the problem, relative to the continuous case under moment-based sets. We develop a tailored column-and-constraint generation algorithm by leveraging the structure of a novel support set and the second-stage problem. We summarize our contributions as follows:

- We solve a distributionally robust disaster planning model by using a richer ambiguity set which does not require estimating moments or making any assumptions on the underlying distribution of the random parameters. Moreover, we achieve greater modeling power via binary variables in the second stage.

- By constructing a support set and defining an underlying distance metric of the support that is tailored to disaster management, we develop an efficient column-and-constraint generation algorithm, where we leverage the structure of our support set and second-stage value function. We show the conditions under which the second-stage value function is concave with respect to a subset of the uncertainty, leading to an efficient line search scheme to generate scenarios, and show how these results extend to the case of a fixed-charge network flow second-stage problem.

- Although we present our work in the context of hurricane disasters for ease of exposition, we discuss how our results and methods extend to different types of disasters, such as earthquakes, or disasters related to military operations or disease outbreaks.

- We perform extensive computational experiments, showing the computational efficiency of our method on synthetic instances with very large sets of scenarios, and illustrate our methods on the case study of hurricane threats on the Gulf of Mexico states, analyzing the solutions obtained from DRO and SAA.

While previous works have considered more general uncertainties, including cost, available capacities or edges, we limit ourselves to demand uncertainty due to the complexity introduced by binary variables in the second stage and the Wasserstein ambiguity set, as seen in other studies in emergency planning. Indeed, [25] solve an uncapacitated facility location model, where only the demand location is uncertain and the second-stage decision consists in picking the closest facility opened in the first stage to serve the observed demand location. A classical row-and-column generation algorithm is used to solve the problem. In [40]. the authors solve a continuous chance-constrained model. Besides [25], DRO for disaster management problems has been restricted to continuous second stages. In this paper, we consider a richer model with binary variables present in the second stage, and focus on algorithmic design and implementation. In emergency planning, binary variables permit us to, for example, model fixed-charge costs incurred in the second stage when opening a link between a facility and a disaster location. Such fixed-charge costs could be associated with clearing debris, for example.

We note that our methods can be extended to other types of uncertainty, as in classical column-and-constraint generation algorithms, but further work is required to leverage the structure of more complex uncertainties. For instance, the second-stage value function might be jointly concave with respect to multiple random variables.

The remainder of the paper is organized as follows. In Section 2, we formally define our two-stage model and support set. In Section 3, we study the structure of our model and support set, and present a column-and-constraint generation (CCG) algorithm and the algorithm used to generate new scenarios. In Section 4, we give details on the implementation of our method and how to tackle computational challenges, leading to an improved CCG algorithm. Finally, we present our computational experiments in Section 5 and give conclusions in Section 6.

1.4 Notation

Vectors are printed in bold to differentiate them from scalars. Given a finite support set of scenarios Ξ with an associated index set $\mathscr{S}$, we equivalently refer to both $\boldsymbol{\xi} \in \Xi$ and $s \in \mathscr{S}$ as scenarios. Similarly, for a set of samples of $\{\hat{\boldsymbol{\xi}}^n\}_{n\in\mathscr{N}} \subseteq \Xi$ indexed by $\mathscr{N} \subseteq \mathscr{S}$, we refer to $n \in \mathscr{N}$ and $\hat{\boldsymbol{\xi}}^n$ as samples. To differentiate between a random vector $\boldsymbol{\xi}$ and historical data, we denote samples by a hat as in $\hat{\boldsymbol{\xi}}^n$. For a support set Ξ, we define $\mathscr{M}(\Xi)$ to be the set of all probability distributions supported on Ξ and $\mathrm{d}(\cdot,\cdot)$ to be a valid underlying metric.

2 Model Formulation

2.1 Two-Stage Distributionally Robust Model

Given a network with nodes $\mathscr{V}$, let $I \subseteq \mathscr{V}$ be the set of possible facility locations, let $J \subseteq \mathscr{V}$ be the set of possible demand nodes, and let K be the set of commodities. In the first stage, binary variable o_i is 1 if facility $i \in I$ is opened at a cost of h_i, and continous variables z_{ik} represent the amount of commodity $k \in K$ pre-allocated to facility i at a unit cost of g_k. Each facility $i \in I$ has a known capacity of C_i, and a unit of commodity $k \in K$ requires a capacity of v_k.

In the second stage, let w_{ij} be the fixed-charge cost incurred if link (i, j) is used and let c_{kij} be the per-unit cost of transportation of commodity k on (i, j), for any facility $i \in I$ and demand node $j \in J$. In scenario $s \in \mathscr{S}$, let $d(\boldsymbol{\xi}^s)_{kj}$ be the demand of commodity k at node $j \in J$, where demand is determined by the random vector $\boldsymbol{\xi}^s \in \Xi$, and Ξ is the finite support set of scenarios. Note that we assume cost to be deterministic. We have binary variables σ_{ij} determining whether facility $i \in I$ is serving demand node $j \in J$ and non-negative variables t_{ij}^k which represent the amount of commodity k transported from facility i to demand node j. Let u be the unmet demand and U be the penalty per unit of unmet demand. Variables u ensure we have relatively complete recourse (the second stage problem is feasible for any first stage decision vector and any realization of the uncertainty). Moreover, we assume we know upper bounds M_{ij} on the amount that can be transported from facility i to demand node j. For example, M_{ij} could be the population at demand node $j \in J$.

Finally, we include a set $\mathscr{B}$ in the first stage that can incorporate additional constraints such as available budget, or the maximum number of facilities that can be opened; in the second stage, the set $\mathscr{C}$ can include valid inequalities. $\mathscr{B}$ and $\mathscr{C}$ can be chosen to be empty. The TSDRO model is defined as

$$
\begin{aligned}
\min_{\boldsymbol{o},\boldsymbol{z}} \quad & \sum_{i\in I}\left[h_i o_i + \sum_{k\in K} g_k z_{ik}\right] + \max_{p\in\mathscr{B}_W(\mathbb{P}_N,\theta)} \sum_{s\in\mathscr{S}} p_s Q\left(\boldsymbol{o},\boldsymbol{z},\boldsymbol{\xi}^s\right) \\
\text{s.t.} \quad & \sum_{k\in K} v_k z_{ik} \le C_i o_i, \ \forall i \in I, && \text{(2.1a)} \\
& z_{ik} \ge 0, && \forall i \in I,\ \forall k \in K, \\
& o_i \in \{0,1\}, && \forall i \in I, \\
& (\boldsymbol{o},\boldsymbol{z}) \in \mathscr{B}
\end{aligned}
$$

where the second stage cost $Q\left(\boldsymbol{o},\boldsymbol{z},\boldsymbol{\xi}^s\right)$ is

$$\begin{aligned}
\min_{\boldsymbol{\sigma},\boldsymbol{t},\boldsymbol{u}} \quad & \sum_{i\in I}\sum_{j\in J}\left[w_{ij}\sigma_{ij}+\sum_{k\in K}c_{kij}t_{ij}^k\right]+U\sum_{j\in J}\sum_{k\in K}u_j^k \\
\text{s.t.} \quad & \sum_{j\in J}t_{ij}^k \le z_{ik}, && \forall i\in I,\ \forall k\in K, && \text{(2.2a)}\\
& \sum_{i\in I}t_{ij}^k+u_j^k \ge d(\boldsymbol{\xi}^s)_{kj}, && \forall j\in J,\ \forall k\in K, && \text{(2.2b)}\\
& t_{ij}^k \le M_{ij}\sigma_{ij}, && \forall i\in I,\ \forall j\in J,\ \forall k\in K, && \text{(2.2c)}\\
& t_{ij}^k \ge 0, && \forall i\in I,\ \forall j\in J,\ \forall k\in K, && \text{(2.2d)}\\
& \sigma_{ij}\in\{0,1\}, && \forall i\in I,\ \forall j\in J, && \text{(2.2e)}\\
& u_j^k \ge 0, && \forall j\in J,\ \forall k\in K, && \text{(2.2f)}\\
& (\boldsymbol{\sigma},\boldsymbol{t},\boldsymbol{u})\in\mathscr{C}. && && \text{(2.2g)}
\end{aligned}$$

If $\mathscr{B}$ includes a budget constraint, then $\mathscr{B}=\{(\boldsymbol{o},\boldsymbol{z}):\sum_{i\in I}\left[h_io_i+\sum_{k\in K}g_kz_{ik}\right]\le b\}$, where b is a pre-defined budget in the first stage. We include valid inequalities $\sigma_{ij}\le o_i$ in $\mathscr{C}$ for all $i\in I$ and $j\in J$.

Constraint (2.1a) ensures we do not exceed the capacity of each facility i. Constraint (2.2a) ensures we can only transport what is available from facility i, constraint (2.2b) ensures demand satisfaction with additional variables u_j^k to keep track of unmet demand, and constraint (2.2c) ensures we only use open links (i,j). We note that the second stage problem is always feasible for all first stage decisions $(\boldsymbol{o},\boldsymbol{z})$ and random demand $\boldsymbol{d}(\boldsymbol{\xi})$. Before detailing the construction of the support set Ξ, we formally introduce the Wasserstein ambiguity set.

2.2 Wasserstein Ambiguity Set

We assume the support set Ξ is a finite but large set of scenarios indexed by $s\in\mathscr{S}$. Given N samples indexed by $\mathscr{N}\subseteq\mathscr{S}$, we define the empirical distribution as $\hat{\mathbb{P}}_N=\frac{1}{N}\sum_{n\in\mathscr{N}}\delta_{\hat{\boldsymbol{\xi}}^n}$, where $\delta_{\hat{\boldsymbol{\xi}}^n}$ is the Dirac distribution assigning unit mass to $\hat{\boldsymbol{\xi}}^n$, i.e. each sample is assigned equal probability. The Wasserstein ball of radius θ centered at $\mathbb{P}_N$ is defined as

$$\mathscr{B}_W(\mathbb{P}_N,\theta)=\{\mathbb{P}\in\mathscr{M}(\Xi):\mathrm{d}_W(\mathbb{P},\mathbb{P}_N)\le\theta\},$$

where $\mathrm{d}_W(\mathbb{P}, \mathbb{P}_N)$ is the Wasserstein distance between $\mathbb{P}$ and $\mathbb{P}_N$, and corresponds to the minimization problem

$$\begin{aligned}
\min_{\boldsymbol{\pi}} \quad & \sum_{s\in\mathscr{S}}\sum_{n\in\mathscr{N}} \mathrm{d}(\boldsymbol{\xi}^s, \hat{\boldsymbol{\xi}}^n)\pi_{sn} \\
\text{s.t.} \quad & \sum_{n\in\mathscr{N}} \pi_{sn} = p_s, \qquad s \in \mathscr{S}, \\
& \sum_{s\in\mathscr{S}} \pi_{sn} = \frac{1}{N}, \qquad n \in \mathscr{N}, \\
& \sum_{s\in\mathscr{S}}\sum_{n\in\mathscr{N}} \pi_{sn} = 1, \\
& \pi_{sn} \geq 0, \qquad\qquad s \in \mathscr{S},\ n \in \mathscr{N}.
\end{aligned} \tag{W}$$

Let $q_s^n = N\pi_{sn}$ be the conditional probability of $\boldsymbol{\xi}^s$ given that we have observed $\hat{\boldsymbol{\xi}}^n$ (the sampled scenario). It is convenient to write (W) as

$$\begin{aligned}
\min_{\boldsymbol{q}} \quad & \sum_{s\in\mathscr{S}}\sum_{n\in\mathscr{N}} \frac{1}{N} q_s^n \,\mathrm{d}(\boldsymbol{\xi}^s, \hat{\boldsymbol{\xi}}^n) \\
\text{s.t.} \quad & \sum_{n\in\mathscr{N}} \frac{1}{N} q_s^n = p_s, \forall s \in \mathscr{S}, \\
& \sum_{s\in\mathscr{S}} q_s^n = 1, \qquad \forall n \in \mathscr{N}, \\
& q_s^n \geq 0, \qquad\qquad s \in \mathscr{S},\ n \in \mathscr{N},
\end{aligned} \tag{W'}$$

where $\{\boldsymbol{\xi}^s\}_{s\in\mathscr{S}}$ is the set of all scenarios and $\{\hat{\boldsymbol{\xi}}^n\}_{n\in\mathscr{N}}$ is the set of sampled scenarios.

2.3 Support Set

Since the objective of DRO is to protect the decision-maker against potentially unobserved events, constructing a set of scenarios using only historical data is undesirable, especially when data is limited. On the other hand, constructing an overly complicated support set could lead to unrealistic disaster scenarios. To this end, we define a set of scenarios such that they are descriptive of the disaster, and lead to a manageable support set.

We define a random vector $\boldsymbol{\xi}$ using two sets of components: 1) One set of components represent the geographical characteristics of the disaster, determining which nodes are affected, and 2) a component determining the intensity of the disaster. More specifically, we define $\boldsymbol{\xi}$ to be comprised of four components: the landfall of the disaster, the radius of impact, the path of the disaster, and the fraction of the population

that is affected at the landfall node. Here, the first three components determine which nodes are affected by the disaster, and the last component is a measure of the intensity of the disaster, determining demand at the landfall; in every scenario, we assume there is exactly one landfall. Conditioned on the demand at the landfall node, we assume that the demands at the remaining affected nodes are deterministic, where the fraction of the population affected decreases the farther the node is from the landfall.

Let L be the set of landfalls, R be the set of radii of impact, A be the set of paths represented by angles, and let F be the set of possible fractions of the population affected at the landfall node. A scenario is constructed in a hierarchical fashion, where we first observe a two-dimensional vector $\boldsymbol{\xi}_{\boldsymbol{\ell}}$ representing the longitude and latitude of the landfall node, followed by a radius of impact $\xi_r \in R$ around the landfall. Although $\boldsymbol{\xi}_{\boldsymbol{\ell}}$ is a vector of the landfall nodes's coordinates, we shall mostly refer to $\boldsymbol{\xi}_{\boldsymbol{\ell}}$ as a node in J for simplicity. The angle $\xi_a \in A$ determines the portion of nodes within the radius ξ_r which are affected by the disaster. ξ_a can, for example, represent the slice of the circle that is affected. Finally, we observe the fraction of the population affected $\xi_f \in F$ at the landfall node. We then have

$$\Xi = \left\{ (\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f) : \boldsymbol{\xi}_{\boldsymbol{\ell}} \in L,\ \xi_r \in R,\ \xi_a \in A,\ \xi_f \in F \right\}.$$

Given a random vector $\boldsymbol{\xi} \in \Xi$, we determine the components of the demand vector $\boldsymbol{d}(\boldsymbol{\xi})$ as follows. Let $J(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a) = \{\boldsymbol{\xi}_{\boldsymbol{\ell}}, j_1, j_2, ...\} \subseteq J$ be the set of affected nodes as determined by the landfall $\boldsymbol{\xi}_{\boldsymbol{\ell}}$, the radius of impact ξ_r and angle ξ_a, and assume nodes are in increasing order of distance from the landfall. Note that the landfall $\boldsymbol{\xi}_{\boldsymbol{\ell}}$ is included in this set. We define functions $\Pi_j^k : L \times R \times A \times F \to [0,1]$ which map $\boldsymbol{\xi}$ to the fraction of the population needing commodity $k \in K$ at nodes $j \in J(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a)$, such that $\Pi^k_{\boldsymbol{\xi}_{\boldsymbol{\ell}}}(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f) \geq \Pi^k_{j_1}(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f) \geq \Pi^k_{j_2}(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f) \geq \dots$ for all k. These functions model how $\boldsymbol{\xi}$ translates to the fraction of population needing each commodity and how demand decreases as we move away from the landfall node. Multiplying the values of these functions by the population at each affected node provides the demand vector $\boldsymbol{d}(\boldsymbol{\xi})$. We allow the functions Π_j^k to be constant for all or any subset of the first three components. We give more specific details in Section 3.3.

We give an example of two scenarios with the same landfall and radius of impact but with different angles in Figure 1. This network was used in a case study of hurricane threats in the Gulf of Mexico states in [28] and is the case study on which we illustrate our method in Section 5.3.

In TSDRO under the Wasserstein ambiguity set, the model hedges against scenarios by transporting

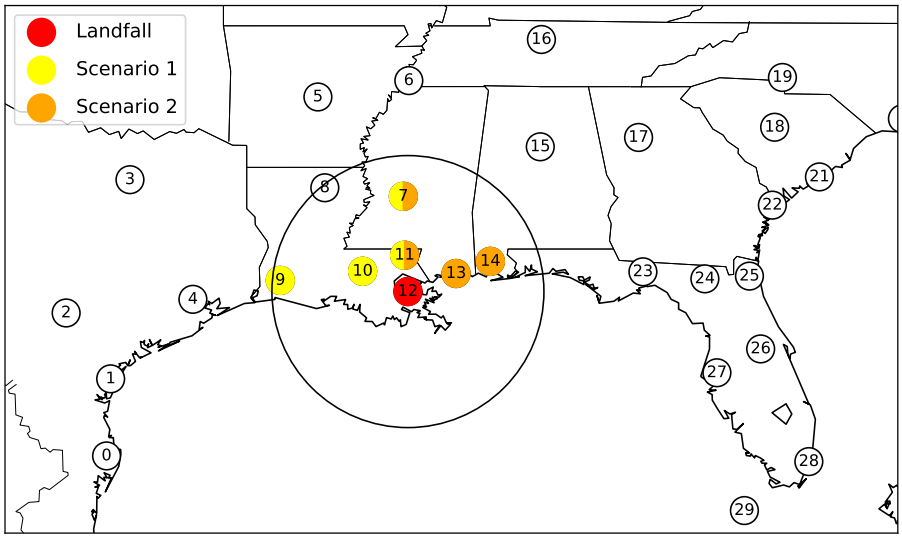


Figure 1: Example of two different angles ξ_a, where the landfall is New Orleans (node 12) and the radius is 320 km. Nodes filled with both colors are nodes affected in both scenarios.

weights from the scenarios in the sample set to other scenarios, possibly not in the sample set (i.e. unobserved scenarios) at a cost proportional to the distance $\mathrm{d}(\hat{\boldsymbol{\xi}}, \boldsymbol{\xi})$, subject to a budget θ, the radius of the Wasserstein ball. Our support set permits us to better measure dissimilarities between two scenarios by defining a metric tailored to the application in the Wasserstein ambiguity set, which leads to a more natural model, as well as algorithmic benefits which we show in Section 3. We define the distance between two scenarios as the sum of squared deviations between the components of $\boldsymbol{\xi}$:

$$\mathrm{d}(\boldsymbol{\xi}^{\mathbf{1}}, \boldsymbol{\xi}^{\mathbf{2}}) = ||\boldsymbol{\xi}^{\mathbf{1}}_{\boldsymbol{\ell}} - \boldsymbol{\xi}^{\mathbf{2}}_{\boldsymbol{\ell}}||^2 + (\xi^1_r - \xi^2_r)^2 + (\xi^1_a - \xi^2_a)^2 + (\xi^1_f - \xi^2_f)^2 \tag{2.3}$$

Recall $\boldsymbol{\xi}_{\boldsymbol{\ell}}$ is a two-dimensional vector representation of the coordinates, while the other components are scalars. Note that by using the metric defined in (2.3), the model considers disaster scenarios hitting landfalls that are close to each other or having similar characteristics (radius, path and intensity) to be similar scenarios. If we simply use the ℓ_2-norm between demand vectors, the relative distance between scenarios can be misleading, as certain underlying characteristics of the disasters such as location are lost. Finally, when computing distances between scenarios, it may be desirable to normalize $\boldsymbol{\xi}$ such that each component is between 0 and 1, to avoid having certain components dominate others.

3 Column and Constraint Generation Algorithm

In this section, we present our column and constraint generation algorithm (CCG). We first present a known extensive reformulation of (2.1), then describe how we leverage the structure of our support set and second stage value function.

3.1 Extensive Reformulation

To simplify notation, we concatenate all first stage variables under the vector $\boldsymbol{x}$, and all second stage variables under the vector $\boldsymbol{y}$. Let X be the feasible region of the first stage and let $Y(\boldsymbol{x}, \boldsymbol{\xi})$ be the feasible region of the second stage given $\boldsymbol{x}$ and random vector $\boldsymbol{\xi}$. More specifically, we work with (DRO), where $Q(\boldsymbol{x}, \boldsymbol{\xi}) = \min_{\boldsymbol{y}} \{\boldsymbol{q}^\top \boldsymbol{y} : \boldsymbol{y} \in Y(\boldsymbol{x}, \boldsymbol{\xi})\}$.

The following theorem presents a known extensive reformulation of (DRO).

Theorem 1. *Problem* (2.1) *is equivalent to*

$$
\begin{aligned}
\min \quad & \boldsymbol{c}^\top \boldsymbol{x} + \theta\lambda + \frac{1}{N}\sum_{n\in\mathcal{N}} \alpha_n \\
\text{s.t.} \quad & \alpha_n \geq \boldsymbol{q}^\top \boldsymbol{y}^s - \lambda\, \mathrm{d}(\boldsymbol{\xi}^s, \hat{\boldsymbol{\xi}}^n),\ s \in \mathcal{S}, n \in \mathcal{N}, \\
& \boldsymbol{y}^s \in Y(\boldsymbol{x}, \boldsymbol{\xi}^s), \qquad\qquad s \in \mathcal{S}, \\
& \boldsymbol{x} \in X, \\
& \lambda \geq 0
\end{aligned}
\tag{ER}
$$

Proof. See Appendix A.1 □

In the reformulated inner maximization problem of (DRO) (see Appendix A.1), the conditional probabilities q_s^n can be interpreted as the hedging strategy, where we are moving weight from the sampled scenario $\hat{\boldsymbol{\xi}}^n$ to scenario $\boldsymbol{\xi}^s$, at a cost of $\frac{1}{N}\mathrm{d}(\boldsymbol{\xi}^s, \hat{\boldsymbol{\xi}}^n)$, and the total weight moved from sample $\hat{\boldsymbol{\xi}}^n$ to all scenarios $\{\boldsymbol{\xi}^s\}_{s\in\mathcal{S}}$ has to equal one.

Due to the hierarchical nature of the scenarios, there is an obvious ordering of the scenarios with respect to the second-stage cost. For example, for the same landfall and direction, increasing the radius and/or intensity at the landfall leads to a greater second stage cost. We can reduce the number of scenarios to consider in the model and get an equivalent problem as follows. We define the following two sets for each

sample n:

$$\bar{\Xi}_n = \left\{ \boldsymbol{\xi} \in \Xi : \ (\xi_f < \hat{\xi}_f^n,\ \xi_r < \hat{\xi}_r^n) \ \text{ or } \ (\xi_f = \hat{\xi}_f^n,\ \xi_r < \hat{\xi}_r^n) \ \text{ or } \ (\xi_f < \hat{\xi}_f^n,\ \xi_r = \hat{\xi}_r^n) \right\},$$
$$\Xi_n = \Xi \setminus \bar{\Xi}_n.$$

The sets Ξ_n exclude scenarios where both intensity and radius are strictly smaller than the sample in question, and scenarios where one of the two is strictly smaller and the other is the same. For each Ξ_n, we define an associated index set $\mathscr{S}_n$ and define the set of all scenarios as $\mathscr{S}_{\mathscr{N}} = \bigcup_{n \in \mathscr{N}} \mathscr{S}_n$.

Theorem 2. (ER) *is equivalent to*

$$\begin{aligned}
\min \quad & \boldsymbol{c}^\top \boldsymbol{x} + \theta\lambda + \frac{1}{N} \sum_{n \in \mathscr{N}} \alpha_n & & & \\
s.t. \quad & \alpha_n \geq \boldsymbol{q}^\top \boldsymbol{y}^{\boldsymbol{s}} - \lambda\, \mathrm{d}(\boldsymbol{\xi}^s, \hat{\boldsymbol{\xi}}^n), & s \in \mathscr{S}_n,\ n \in \mathscr{N}, & \quad (c1), & \\
& \boldsymbol{y}^{\boldsymbol{s}} \in Y(\boldsymbol{x}, \boldsymbol{\xi}^s), & s \in \mathscr{S}_{\mathscr{N}}, & \quad (c2), & \\
& \boldsymbol{x} \in X, & & & \\
& \lambda \geq 0. & & &
\end{aligned} \tag{ER'}$$

Proof. See Appendix A.2 □

The sets Ξ_n exclude dominated scenarios which need not be included in the model. For a sample $\hat{\boldsymbol{\xi}}^n$, each scenario $\boldsymbol{\xi} \in \bar{\Xi}_n$ has a corresponding scenario $\boldsymbol{\xi'} \in \Xi_n$ which dominates $\boldsymbol{\xi}$, where $\mathrm{d}(\hat{\boldsymbol{\xi}}, \boldsymbol{\xi'}) < \mathrm{d}(\hat{\boldsymbol{\xi}}, \boldsymbol{\xi})$ and $Q(\boldsymbol{x}, \boldsymbol{\xi'}) \geq Q(\boldsymbol{x}, \boldsymbol{\xi})$. In other words, moving weight to $\boldsymbol{\xi'}$ would lead to a greater cost in the inner maximization of (2.1) while using less of the available budget θ in the Wasserstein ambiguity set. Note that the reduction in the number of scenarios included in the model directly depends on the sample set.

3.2 Column & Constraint Generation

Since the number of scenarios $|\mathscr{S}_{\mathscr{N}}|$ can be very large, solving (ER') is computationally infeasible. As in typical CCG algorithms, we start with a subset of scenarios $\hat{\mathscr{S}}_n \subseteq \mathscr{S}_n$ for each sample n, solve a restricted model, which we denote as (RER), and record optimal solutions $\hat{\boldsymbol{x}}, \hat{\boldsymbol{\alpha}}$ and $\hat{\lambda}$. For each sample $n \in \mathscr{N}$, we then search for a new scenario $s \in \mathscr{S}_n \setminus \hat{\mathscr{S}}_n$ such that $\hat{\alpha}_n < Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi}^s) - \hat{\lambda}\, \mathrm{d}(\boldsymbol{\xi}^s, \hat{\boldsymbol{\xi}}^n)$, where $Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi}^s)$ is obtained by solving the second stage problem given first stage decision $\hat{\boldsymbol{x}}$ and random vector $\boldsymbol{\xi}^s$. Note that we can

generate multiple scenarios per iteration. We repeat the process until we can no longer find such a scenario for any sample. We refer to the process of generating new scenarios as the separation problem. With a slight abuse of language, for some sample n, we say that a scenario s violates (c1) if constraint (c1) is violated at scenario s.

We summarize a vanilla CCG algorithm in Algorithm 1. The function `SEPARATION(`$\hat{\boldsymbol{\xi}}^n, \hat{\boldsymbol{x}}, \hat{\alpha}_n, \hat{\lambda}$`)` returns a set $V_n \in \mathscr{S}_n \setminus \hat{\mathscr{S}}_n$ of new scenarios which violate $\alpha_n \geq Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi}^s) - \lambda\, \mathrm{d}(\boldsymbol{\xi}^s, \hat{\boldsymbol{\xi}}^n)$ for a sample n and optimal solutions $\hat{\boldsymbol{x}}$, $\hat{\alpha}_n$ and $\hat{\lambda}$ of (RER).

Algorithm 1 CCG

```
Initialize 𝒮̂_n for each n ∈ 𝒩
while ∪_{n∈𝒩} V_n ≠ ∅ do
    Solve (RER) and record optimal solutions (x̂, α̂, λ̂).
    for each n ∈ 𝒩 do
        V_n ← SEPARATION(ξ̂^n, x̂, α̂_n, λ̂)
        𝒮̂_n ← 𝒮̂_n ∪ V_n
    end for
end while
```

3.3 Separation Problem

There are two main difficulties that arise in Algorithm 1. The first, more significant one, is in the separation step, where one might have to enumerate all scenarios in $\mathscr{S}_n \setminus \hat{\mathscr{S}}_n$, which is a very large set. Recall that for each scenario, we must solve a fixed-charge transportation problem (2.2). While today's commercial solvers can typically efficiently solve such models, the difficulty lies in repeatedly solving (2.2) a large number of times, and having to do it after each solve of (RER), the restricted model of (ER'). The second main difficulty is in solving (RER) after having generated many scenarios. After each iteration, the number of scenarios generated, if not controlled, can be large. Consider the case where each sample generates one new scenario. For each scenario s generated, we add the associated constraint (c1), a new set of variables y^s, and a set of second stage constraints (c2). Depending on the number of samples, this can lead to a large and difficult master problem.

Given optimal solutions $\hat{\boldsymbol{x}}$ and $\hat{\lambda}$ of (RER), the separation problem consists of solving the following

problem for each sample n:

$$\begin{aligned} \max_{\boldsymbol{\xi}} \quad & z_n(\boldsymbol{\xi}) \coloneqq Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi}) - \hat{\lambda}\, \mathrm{d}(\boldsymbol{\xi}, \hat{\boldsymbol{\xi}}^n) \\ \text{s.t.} \quad & \boldsymbol{\xi} \in \Xi. \end{aligned} \tag{SP_n}$$

Let $\boldsymbol{\xi}^{s*}$ be the optimal solution and let z_n^* be the optimal value of (SP_n). If $z_n^* > \hat{\alpha}_n$, then we generate a new scenario s^*; otherwise, we conclude none of the scenarios violate (c1) for sample n.

One simple greedy approach is to enumerate the scenarios, and stop at the first one that violates constraint (c1). We repeat this for each sample n. Such a separation method generally leads to quickly generating new scenarios, especially in the early stages of the algorithm, but typically leads to a greater number of outer iterations (i.e. master solves). Moreover, as the algorithm progresses, we enumerate more and more scenarios before finding one which violates (c1), and in the worst case, could end up enumerating all scenarios for some of the samples. Alternatively, we can solve (SP_n) by enumerating all scenarios and pick the scenario with the greatest objective $z_n(\boldsymbol{\xi})$.

We instead leverage the structure of our support set Ξ and the optimal value of the second stage $Q(\boldsymbol{x}, \boldsymbol{\xi})$ to solve (SP_n). Recall that $J(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a)$ is the set of affected nodes determined by the landfall $\boldsymbol{\xi}_{\boldsymbol{\ell}}$, radius of impact ξ_r and angle ξ_a, including the landfall $\boldsymbol{\xi}_{\boldsymbol{\ell}}$. We make the following two assumptions:

Assumption 1. *The functions $\Pi_j^k(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f) : L \times R \times A \times F \to [0,1]$ determining the fraction of the population needing commodity k at nodes $j \in J(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a)$ are concave in ξ_f for each landfall $\boldsymbol{\xi}_{\boldsymbol{\ell}} \in L$, radius of impact $\xi_r \in R$, and angle $\xi_a \in A$.*

Assumption 2. *A facility at node $i \in I$ has a limit k_i on the number of nodes it can serve, but is able to satisfy any amount of demand at nodes it is serving.*

Assumption 1 ensures we have well-behaved mappings Π_j^k. An example of a function satisfying Assumption 1 is the piece-wise linear function $\Pi_j^k(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f) = \min\{a\xi_f, \bar{\xi}_{fj}\}$ which is linear in ξ_f with upper bounds $\bar{\xi}_{fj}$ on each demand node j, where $a \leq 1$ is a pre-determined parameter. The upper bounds might depend on $\boldsymbol{\xi}_{\boldsymbol{\ell}}$, ξ_r, ξ_a and $k \in K$, but we drop this dependence in $\bar{\xi}_{fj}$ to simplify notation. Since the intensity of the disaster decreases as we move farther from $\boldsymbol{\xi}_{\boldsymbol{\ell}}$, upper bounds $\bar{\xi}_{fj}$ also decrease the farther j is from $\boldsymbol{\xi}_{\boldsymbol{\ell}}$.

Under Assumption 2, we no longer pre-allocate resources in the first stage and thus, variables z_{ik} are removed from the model. The upper bounds k_i can either be a pre-defined parameter, or a first stage decision

replacing z_{ik}. Variables k_i can be interpreted as the size of the facility built at node $i \in I$. Such an assumption is reasonable when it is not affordable or practical to pre-allocate resources which will sit unused in a facility for potentially years, and that resources will only be acquired as needed in the event of a disaster. In the case where k_i is a pre-determined parameter, (2.1) becomes:

$$\begin{aligned} \min_{\boldsymbol{o}} \quad & \sum_{i \in I} h_i o_i + \max_{p \in \mathscr{B}_W(\mathbb{P}_N, \theta)} \sum_{s \in \mathscr{S}} p_s Q(\boldsymbol{o}, \boldsymbol{\xi}^s) \\ \text{s.t.} \quad & o_i \in \{0,1\}, \forall i \in I, \\ & \boldsymbol{o} \in \mathscr{B} \end{aligned} \tag{3.1}$$

where the second stage cost $Q(\boldsymbol{o}, \boldsymbol{\xi}^s)$ is

$$\min_{\boldsymbol{\sigma}, \boldsymbol{t}, \boldsymbol{u}} \quad \sum_{i \in I} \sum_{j \in J} \left[w_{ij} \sigma_{ij} + \sum_{k \in K} c_{kij} t_{ij}^k \right] + U \sum_{j \in J} \sum_{k \in K} u_j^k$$

$$\text{s.t.} \quad \sum_{j \in J} \sigma_{ij} \leq k_i, \quad \forall i \in I, \tag{3.2a}$$

$$\sigma_{ij} \leq o_i \quad \forall i \in I, \forall j \in J, \tag{3.2b}$$

$$(2.2b) - (2.2g)$$

Constraint (3.2a) ensures each facility $i \in I$ only serves at most k_i demand nodes $j \in J$ and constraint (3.2b) ensures a link (i, j) can only be used if facility i is open. Note that if a demand node cannot be served, the unsatisfied demand is still penalized by U.

Assumptions 1 and 2 are key to exploiting the structure of the second stage optimal cost $Q(\boldsymbol{x}, \boldsymbol{\xi})$ and lead to the following results.

Theorem 3. *Under Assumptions 1 and 2, for any first stage decision vector $\boldsymbol{x}$, $Q(\boldsymbol{x}, \boldsymbol{\xi})$ is concave with respect to ξ_f for a fixed landfall $\boldsymbol{\xi}_{\ell}$, radius of impact ξ_r and angle ξ_a.*

Corollary 1. *Under Assumptions 1 and 2, for any first stage decision vector $\boldsymbol{x}$ and $\lambda \geq 0$, $z_n(\boldsymbol{\xi})$ is concave with respect to ξ_f for a fixed landfall $\boldsymbol{\xi}_{\ell}$, radius of impact ξ_r and angle ξ_a.*

Proof. See Appendix A.3 □

Theorem 3 and its corollary motivate performing a line search on ξ_f. Since F is a discrete set, we perform a Fibonacci search on F, which is the discrete version of a golden-section line search. More

specifically, we enumerate and fix $\boldsymbol{\xi}_{\ell}$, ξ_r and ξ_a, and solve $\max_{\xi_f \in F} f(\xi_f)$ using a Fibonacci search, where $f(\xi_f)$ is the objective of the separation problem $z_n(\xi)$ as a function of ξ_f with $\boldsymbol{\xi}_{\ell}$, ξ_r and ξ_a fixed. At each step of the Fibonacci search, we compute $f(\xi_f)$ by solving the second stage problem $Q(\boldsymbol{x}, \boldsymbol{\xi})$. This separation scheme, which we refer to as `SEPARATION-FIB`$\left(\hat{\boldsymbol{\xi}}^n, \hat{\boldsymbol{x}}, \hat{\alpha}_n, \hat{\lambda}\right)$, is described in Appendix C.2. The Fibonacci search `FIBONACCI`$\left(F, \boldsymbol{\xi}_{\ell}, \xi_r, \xi_a, \hat{\boldsymbol{\xi}}^n\right)$ takes as arguments the set of fractions F, the values of the fixed components of $\boldsymbol{\xi}$ and the sample $\hat{\boldsymbol{\xi}}^n$. Details on the Fibonacci search can be found in [15]. We detail our implementation of the Fibonacci search for self-containment in Appendix C.1.

Remark. *`FIBONACCI` terminates in $\mathcal{O}(\log|F|)$ steps [15]. Thus, to solve (SP_n), the second stage problem is solved $\mathcal{O}(|L| \times |R| \times |A| \times \log|F|)$ times, as opposed to $\mathcal{O}(|L| \times |R| \times |A| \times |F|)$ times if enumerating all scenarios.*

Since F is essentially the demand component of our random vector, we expect it to have the highest cardinality compared to L, R, and A. In fact, since two-stage DRO under the Wasserstein set aims to protect the decision-maker against unobserved scenarios, a finer discretization of F might be desired. By using `SEPARATION-FIB`, the separation step scales logarithmically in the size of F.

If we wish to solve the original model (2.1) when Assumption 2 does not hold, `SEPARATION-FIB` is not guaranteed to return an optimal solution to (SP_n). However, we observe in our experiments that $z_n(\boldsymbol{\xi})$ is still concave with respect to ξ_f for a fixed landfall, radius and angle in the vast majority of cases, and in the few cases where it is not concave, the solution is optimal or close to optimal. Thus, we can perform `SEPARATION-FIB` as an efficient heuristic. To ensure an exact method, we can perform a full enumeration to solve (SP_n) once no scenarios are generated by the heuristic; see Section 5 for more details.

3.4 Extensions

Our results can be extended to the case where the second-stage is a fixed-charge network flow problem; we provide details in Appendix B. Moreover, our model and methods can be extended to other types of disasters that can be described via "location" components (in our case, the landfall, radius of impact and path of the hurricane) and an intensity component, as long as it is practically reasonable to assume there is only demand in isolated regions of the network, which can be determined by the intensity of the disaster via an appropriate mapping Π as described in Section 2.3. Our support set is flexible in that it can be modified accordingly to account for different types of disasters such as earthquakes, wildfires, floods, biological or military disasters.

For example, in the case of earthquakes, we can drop the path component and only consider landfall, radius of impact and intensity. Moreover, commodities can also represent equipment or people to clear debris or repair damages to essential nodes, links, or infrastructure such as electricity or water [27].

In certain applications, we might want to consider multiple landfalls. For example, consider a military application where decision-makers wish to protect themselves against attacks that can occur in multiple locations. Alternatively, consider the problem of planning for a response to disease outbreaks which can occur at the same time in different locations of the network. To consider multiple landfalls, our results (with a slight modification to Theorem 2) remain valid if in any scenario involving more than one landfall, there can never be overlap in the affected nodes. In other words, given that each landfall, radius and angle result in a set of affected nodes, the intersection of these sets must be empty across all scenarios. Consider the case where there can be up to two landfalls. There would then be intensities ξ_{f_1} and ξ_{f_2} associated with landfalls $\boldsymbol{\xi}_{\ell_1}$ and $\boldsymbol{\xi}_{\ell_2}$, respectively. Then $z_n(\boldsymbol{\xi})$ is jointly concave with respect to ξ_{f_1} and ξ_{f_2}. `SEPARATION-FIB` would be modified to enumerate over $(\boldsymbol{\xi}_{\ell_1}, \xi_{r_1}, \xi_{a_1}, \boldsymbol{\xi}_{\ell_2}, \xi_{r_2}, \xi_{a_2}) \in L \times R \times A \times L \times R \times A$ instead. A more sophisticated joint search over ξ_{f_1} and ξ_{f_2} would be required, however.

4 Algorithm Implementation

In this section, we focus on the implementation and practical details of the algorithm to make it computationally efficient.

4.1 Solving RER

We have so far only discussed how to efficiently solve the separation problem, but have not tackled the restricted model (RER), which might become difficult to solve as the algorithm progresses and as we generate many scenarios, especially for larger numbers of samples. First, to get a tighter formulation, we include valid inequalities $\sigma_{ij} \leq o_i$ in $\mathscr{C}$ in the second stage model (2.2) as mentioned in Section 2 (note that this constraint is required if solving (3.1)). Moreover, we set upper bounds M_{ij} in constraint (2.2c) to be the highest demand that can possibly occur in any scenario at demand node j, i.e. $M_{ij} = \max_{\xi_f \in F} \xi_f P_j$, where P_j is the population at node j.

To control the size of (RER), we limit the number of generated scenarios in an iteration. The limit can be dynamically adjusted as the algorithm progresses. There are many possible variations, such as generating

multiple scenarios for one sample until we hit the limit, or setting a limit for each sample in addition to a limit across all samples. For simplicity, we ensure only one scenario is generated per sample, and only impose a limit on the total number of scenarios generated across all samples, which we keep constant throughout the algorithm. Finally, we observe in our experiments that the solver spends the vast majority of the time proving optimality, and seems to quickly find good, feasible integer solutions. To this end, we also dynamically adjust the MIP gap tolerance when solving (RER), where we start with a loose tolerance and tighten it as the algorithm progresses.

4.2 Separation Step Implementation

In the separation problem, when searching for a scenario that violates constraint (c1) for some sample n, we solve the second-stage problem to get the optimal value $Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi})$ for a first stage vector $\hat{\boldsymbol{x}}$ and candidate scenario $\boldsymbol{\xi}$. Once the second stage is solved at the candidate scenario $\boldsymbol{\xi}$ for some sample n, we do not need to re-compute $Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi})$ during the search for violated constraints at another sample n'. In other words, for a new sample n', we can solve (SP_n) by evaluating $Q^*(\hat{\boldsymbol{x}}, \boldsymbol{\xi}) - \hat{\lambda}\, \mathrm{d}(\boldsymbol{\xi}, \hat{\boldsymbol{\xi}}^{n'})$ for all $\boldsymbol{\xi} \in \Xi$, where $Q^*(\hat{\boldsymbol{x}}, \boldsymbol{\xi})$ is the optimal value of the second stage and is known for all $\boldsymbol{\xi} \in \Xi$, computed during the search of a violated constraint at the first sample n. This can be done very efficiently.

This leads to the question of how beneficial it is to use Fibonacci search in the separation step. For the first sample, using Fibonacci search does indeed lead to far fewer scenarios enumerated, where $Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi})$ is solved at a few select scenarios $\boldsymbol{\xi} \in \Xi$. However, by the end of the separation step, it is still possible that we compute $Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi})$ for all $\boldsymbol{\xi} \in \Xi$ in the worst case. Indeed, when solving (SP_n) for the second sample, the Fibonacci search might visit different scenarios from the ones visited for the first sample. For these scenarios, $Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi})$ must be solved. As the number of samples increases, it becomes more likely that this event occurs, especially for larger sets F. We observe in our experiments that although `SEPARATION-FIB` can never perform worse than full enumeration, the computational benefits can be further improved.

We empirically observe that for fixed $\boldsymbol{\xi}_\ell$, ξ_r and ξ_a, the maxima of $f(\xi_f)$ can be close to each other across different samples, if not the same in many instances. This motivates the following procedure: we perform `SEPARATION-FIB` for the first sample in the list, then simply consider scenarios for which the second-stage model has been solved for the remaining samples, where we return the scenario at which $z_n(\xi)$ is greatest. In other words, we are assuming that for the same fixed components $\boldsymbol{\xi}_\ell$, ξ_r and ξ_a, the maximum always occurs at the same ξ_f across the samples. In the next iteration, we perform `SEPARATION-FIB` for

the second sample, and so on. This heuristic to solving (SP_n) leads to significant computational benefits as we see in Section 5.2. Once no scenarios can be generated, we still have to perform a final check where we perform `SEPARATION-FIB` for each sample to ensure an exact method. This leads to our final column and constraint generation algorithm, which we call CCG-FIB.

We formalize and present CCG-FIB in Algorithm 2. In this context, the set of samples $\mathcal{N}$ is interpreted as an array which can be indexed. We refer to the scenarios at which the second-stage model has been solved at the current first stage decision $\hat{\boldsymbol{x}}$ as $\hat{\Xi}$ (i.e. during `SEPARATION-FIB`), and the process of enumerating them as `LAZY-ENUMERATION`$\left(\hat{\Xi}, \hat{\boldsymbol{\xi}}^n, \hat{\alpha}_n, \hat{\lambda}\right)$. In general, this function can return a set of scenarios $V_n \subseteq \hat{\Xi}$ violating (c1), but we limit it to returning at most one scenario whose objective value $z_n(\boldsymbol{\xi})$ is the greatest. Note that `LAZY-ENUMERATION` simply computes $z_n(\boldsymbol{\xi})$ for all $\boldsymbol{\xi} \in \hat{\Xi}$ without resolving $Q(\hat{\boldsymbol{x}}, \boldsymbol{\xi})$. We keep track of a boolean `final_check`, which is initialized to `False`. When `False`, we perform `SEPARATION-FIB` for one sample, and `LAZY-ENUMERATION` for the remaining samples. Thus, during this phase, we break out of the while-loop (line 9) at the end of the first iteration and jump to line 13, where we enumerate scenarios $\boldsymbol{\xi} \in \hat{\Xi}$, i.e. scenarios for which the second-stage problem has been solved at the current iteration. We then increment the sample index i in line 17, where the index loops back to 0 after reaching the end of the array. This ensures we rotate through the samples for which we fully solve $\max_{\boldsymbol{\xi}} z_n(\boldsymbol{\xi})$. Once no scenarios are generated, `final_check` becomes `True` (line 22). During this phase, we perform `SEPARATION-FIB` for each sample, and reset the sample index i to 0 in line 19. If no scenarios are generated, we reset `final_check` to `False` (line 24) and the algorithm exits the outer while-loop to terminate.

Finally, we note that we have omitted the changes to solving (RER) in Algorithm 2 for ease of exposition, specifically the limit on the number of scenarios generated and the adjustement of the MIP tolerance of (RER). By definition of `SEPARATION-FIB` and `LAZY-ENUMERATION`, we have that $|V_n| = 1$, i.e. we generate at most one scenario per sample, and given a limit on the total number of scenarios that can be generated, we either exit the inner while-loop (5-11) or the for-loop in lines 13-16 once the number of generated scenarios hits the limit. Moreover, given an array of MIP tolerances, we decrease the tolerance when solving (RER) once no scenarios can be generated during the phase where `final_check` is false. In other words, instead of setting `final_check` to `True` in line 22, we decrease the MIP tolerance and restart the process. Once we get to the lowest (tightest) desired tolerance, we proceed as normal where we enter the final check phase once no scenarios can be generated during the first phase.

Algorithm 2 CCG-FIB

Initialize $\hat{\mathscr{S}}_n$ for each $n \in \mathscr{N}$
$i \leftarrow 0$, `final_check` $\leftarrow$ *False*
while $\cup_{n\in\mathscr{N}} V_n \neq \emptyset$ **or** `final_check` **do**
  Solve (RER) and record optimal solutions $(\hat{x}, \hat{\alpha}, \hat{\lambda})$
  while $i \leq |\mathscr{N}| - 1$ **do**
    $n \leftarrow \mathscr{N}_i$
    $V_n \leftarrow$ `SEPARATION-FIB`$(\hat{\boldsymbol{\xi}}^n, \hat{\boldsymbol{x}}, \hat{\alpha}_n, \hat{\lambda})$
    $\hat{\mathscr{S}}_n \leftarrow \hat{\mathscr{S}}_n \cup V_n$
    if not `final_check` **then** `break`
    $i \leftarrow i + 1$
  end while
  if not `final_check` **then**
    for each $n' \in \mathscr{N}$, $n' \neq n$ **do**
      $V_{n'} \leftarrow$ `LAZY-ENUMERATION` $\left(\hat{\Xi}, \hat{\boldsymbol{\xi}}^{n'}, \hat{\alpha}_{n'}, \hat{\lambda}\right)$
      $\hat{\mathscr{S}}_{n'} \leftarrow \hat{\mathscr{S}}_{n'} \cup V_{n'}$
    end for
    $i \leftarrow i + 1 \pmod{|\mathscr{N}|}$
  else
    $i \leftarrow 0$
  end if
  if $\cup_{n\in\mathscr{N}} V_n = \emptyset$ & **not** `final_check` **then** ▷ Start final check
    `final_check` $\leftarrow$ `True`
  else if $\cup_{n\in\mathscr{N}} V_n = \emptyset$ & `final_check` **then** ▷ Algorithm terminates
    `final_check` $\leftarrow$ `False`
  end if
end while

5 Computational Experiments

We perform computational experiments to illustrate our method. We perform a series of experiments on randomly generated networks to test the computational benefits of our proposed method and perform experiments on a case study of hurricane threats on the coastal states of the United States along the Gulf of Mexico and the Atlantic Ocean presented in [28] and further studied in [30], analyzing the solutions obtained by the DRO model under the Wasserstein radius and by SAA. All experiments were performed on compute nodes running Intel Xeon Gold 6226 "Cascade Lake" at 2.7 Ghz. We limit our resources to 8 GB of RAM and 5 processors. We use GUROBI 9.1.0 as the solver for all models.

We first give a brief description of the data and other details that are common to the synthetic instances and the case study. We then discuss how we construct the synthetic instances and results, before tackling the case study.

5.1 Data

5.1.1 Commodity data

We consider only one commodity for simplicity, which can be interpreted as a bundle containing a medical kit, food and water. To calculate the per-unit cost of the bundle, we take the sum of the cost of each commodity adjusted to the needs of one person. More specifically, as in [28], we assume the average person needs three meals and two snacks per day, and one gallon of water per day. We further assume that a disaster lasts five days. Data related to water, food, and medical kits from [28] is given in Table D.1. Thus, 1,000 gallons of water costing \$647.7 to purchase and \$0.3 per unit of distance to transport translates to a cost of approximately \$3.24 to purchase and \$0.0015 to transport per person. Similarly, meal kits cost \$135.5 to purchase and \$0.001 per unit of distance to transport per person, and a medical kit costs \$35 to purchase and \$1.45e-4 to transport. We then have that a bundle costs about \$173.74 to purchase and \$0.0026 per unit of distance to transport.

5.1.2 Facility data

In both the synthetic instances and the case study, we assume we can open one type of facility of capacity 408,200 ft^3 at a cost of \$188,400 as in [30] (equivalent to the medium facility in [28]). Following a similar process as with the costs, a bundle occupies about 3.1 ft^3. Thus, we assume the facility has a capacity of about 131,837, and the bundle occupies a space of one unit.

5.1.3 Support Set

The construction of our support set is heavily influenced by the Hurricane Database (HURDAT) provided by the National Oceanic and Atmospheric Administration (NOAA) [24], which logs the paths and intensities of more than 150 years of hurricanes, as well as by the constructed scenarios in [28]. A visualization tool of the HURDAT data is available at https://coast.noaa.gov/hurricanes. Note that although the network of nodes is randomly generated in the synthetic instances, we assume the shores are to the east and south, representing the Atlantic Ocean and the Gulf of Mexico, respectively, so that the following applies to both the synthetic instances and the case study.

We assume there can only be one landfall node $\boldsymbol{\xi}_\ell$ per scenario, and that landfalls are close to shore. The landfall determines the remaining affected nodes depending on the radius of impact and angle (or path) of

the hurricane. We define the set of radii as $R = \{\xi_r = 100i, i = 0, 1, \ldots, 5\}$ in kilometers and the set of angles as $A = \{0, \frac{-\pi}{4}, \frac{-\pi}{2}, \frac{\pi}{4}, \frac{\pi}{2}\}$ in radians. We provide more details on the construction of F, the set of possible fractions of the population affected at the landfall node, as we present the instances.

We observe in the HURDAT data that hurricanes hitting the coastal states of the United States along the Gulf of Mexico and the Atlantic Ocean tend to take a curved path, generally starting from the south-east and curving towards the north-east and decreasing in intensity as the hurricane advances in-land. Thus, given a landfall and radius of impact, we assume only nodes within the northern half-circle are affected, where the half-circle is rotated by the angle ξ_a. For example, if $\xi_a = 0$, then the base of the half-circle is horizontal, and nodes that are north of the base and within the half-circle are affected. A positive angle rotates the base of the half-circle counter-clockwise and a negative angle rotates it clockwise. In the example given in Figure 1, Scenario 1 is obtained with $\xi_a = \frac{\pi}{4}$ and Scenario 2 with $\xi_a = \frac{-\pi}{4}$. Moreover, we choose functions $\Pi_j(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f) = \min\{\xi_f, \bar{\xi}_{fj}\}$ for all j, $\boldsymbol{\xi}_{\boldsymbol{\ell}}$, ξ_r and ξ_a. Upper bounds $\bar{\xi}_{fj}$ (and thus the functions Π_j) are independent of ξ_r and only depend on $\boldsymbol{\xi}_{\boldsymbol{\ell}}$ and ξ_a. Given a landfall and angle, we determine the upper bound $\bar{\xi}_{fj}$ as follows. We look at all the nodes that can be affected at the maximum radius. If node j is the i^{th} furthest affected node from the landfall, then $\bar{\xi}_{fj}$ is set to the i^{th} largest value in F, or to the smallest value in F if $i > |F|$. This is to also satisfy the fact that the intensity of the hurricane decreases as it travels in-land (i.e. the fraction of the population affected decreases as we move farther away from the landfall) and to satisfy Assumption 1.

Although we do not know the true distribution in practice, we construct a probability distribution associated with each scenario to simulate and test our methods. We use this distribution to get the sample set $\mathcal{N}$ and simulate the out-of-sample performance. It is unlikely that the components of our random vector are truly independent, but we assume independence for simplicity as in [30]. As such, the probability of a scenario $(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f)$ is the product of the probabilities of each component. For the radii, we assume a skew-normal distribution with the median of R, the standard deviation of R and -1 as parameter values for the location, scale and shape, respectively. This is a normal distribution that is slightly left skewed. For the angles, we assume $\mathbb{P}[\xi_a = 0] = \frac{4}{15}$, $\mathbb{P}\left[\xi_a = \frac{-\pi}{4}\right] = \frac{5}{15}$, $\mathbb{P}\left[\xi_a = \frac{-\pi}{2}\right] = \frac{3}{15}$, $\mathbb{P}\left[\xi_a = \frac{\pi}{4}\right] = \frac{2}{15}$, $\mathbb{P}\left[\xi_a = \frac{\pi}{2}\right] = \frac{1}{15}$. The idea is that it is more likely for a hurricane to be pathing east (negative angles have a higher probability) and more likely to be pathing north than at a very steep angle ($\mathbb{P}[\xi_a = 0] > \mathbb{P}\left[\xi_a = \frac{-\pi}{2}\right]$). We constructed these probabilities based on observations of the HURDAT data.

Finally, the probability distribution of ξ_f is determined based on the number of hurricanes by category

that have hit the coastal states between 1851 and 2004. Hurricanes are classified in categories from 1 to 5, with a higher category indicating a more major hurricane. These categories are based on the SAFFIR-SIMPSON scale, which is a rating determined by the hurricane's wind speeds. For example, Hurricane Katrina was a category 5 hurricane which hit the New Orleans area in 2005. According to [28] and [9], the number of hurricanes that have hit the coastal states between 1851 and 2004 at each category from 1 to 5 is 113, 74, 76, 18 and 3, respectively. Given a set F of possible values for ξ_f, we order F in increasing order, assign equal intervals of F to each category and assign the same weight to each value in the interval based on the number of historical hurricanes at each category. For example, if $F = \{0.1, 0.2, 0.3, \ldots, 1\}$, then $\xi_f \in \{0.1, \ldots, 0.2\}$ would be considered category 1 hurricanes and each value in the interval is assigned a weight of 113; $\xi_f \in [0.21, 0.4]$ would be category 2 hurricanes and each value is assigned a weight of 74, and so on. We then normalize so that the probabilities sum to one.

We give details on the probability distribution of $\boldsymbol{\xi}_{\ell}$, the landfall node, in the following sections, as it is different for the synthetic instances and the case study. Depending on the network, the set of nodes hit by a disaster can be the same for two different radii, leading to the same scenario; we exclude such redundant scenarios from our support. Moreover, as discussed in Section 2.3, we normalize the random vectors $\boldsymbol{\xi}$ in our constructed set so that each component is between 0 and 1 whenever we compute distances between scenarios.

5.2 Synthetic Instances

5.2.1 Instance Generation

To create the synthetic instances, we randomly generate a set of nodes on a grid by uniformly picking x and y coordinates between 0 and 20. To avoid generating nodes that are too close to each other, we set a minimum threshold such that any two nodes are at least a Euclidean distance of 2 away from each other. We assume the coast is on the right and bottom sides of the grid as in the case study, and assume nodes within a threshold distance of the coast can be a potential landfall node. An example of a randomly generated network can be found in Appendix D.4. We then generate random populations for each node, where we sample from a uniform distribution on the interval $[1e5, 2e6]$.

Recall that we use a cost of \$173.74 to purchase and \$0.0026 per unit of distance to transport a unit of commodity. Thus, we set the per-unit cost of transportation from facility i to demand node j to be

$c_{ij} = 0.0026\,\mathrm{d}_{ij}$, where d_{ij} is the ℓ_2-norm between node i and j in our grid, $i \neq j$. Since a node can be both a potential facility and demand node, we assume a facility serving its own node incurs the cheapest transportation cost out of all facilities; i.e. it always makes sense for facilities to serve its own node if there is demand. Note that we scale the distances up so that the order of magnitude of the distances is similar to that of the case study. We do this so that the balance between the first and second stage costs is reasonable and comparable to the case study. We set the fixed-charge cost to be a multiple of the cost of transportation, so that $w_{ij} = \phi c_{ij}$. In our experiments, we set $\phi = 5000$ and the penalty for unmet demand $U = 10$.

Finally, we construct a probability distribution for the landfall nodes that is inversely proportional to the sum of distances to the closest eastern and southern points, such that the closer a node is to the coast, the higher the likelihood of it being a landfall.

5.2.2 Experiment Setup

We perform four sets of experiments for different combinations of sample set sizes N and radii θ of the Wasserstein set, where we run tests for a sample size of 10 and 50, and a radius of 1e-2 and 1e-3 for the Wasserstein ball. For each set of experiments, we test how our methods perform as the discretization of F becomes finer, where we pick 15, 30, 50 and 75 equi-distant values between 0.001 and 0.3. In these sets of experiments, we assume Assumptions 1 and 2 hold, and are thus solving (3.1). We assume $\mathscr{B}$ to be empty. In the second stage, we fix the limit on the number of demand nodes that a facility can serve k_i to 3 for all facilities i. We keep the Gurobi parameters to their default values with the exception of the relative MIPGap and Threads parameter. We start with a loose tolerance of 0.12 for the relative MIP gap to solve (RER) and adjust it to 0.01 as detailed in Section 4. We set the limit on the number of scenarios generated per iteration to 5 when $N = 10$ and to 10 when $N = 50$. To reduce the computational resources used, we limit the number of threads that Gurobi can use to 1 when the tolerance is 0.12, and increase it to 5 when the tolerance is 0.01. Finally, we set a time limit of 8 hours for all experiments.

We randomly generate networks of 30 nodes, where we randomly pick a set I of 15 potential facility locations; every node in the network is a potential demand node. We compare our method to two classical CCG algorithms, where the separation step is a simple enumeration process. In one case we stop at the first scenario violating constraint (c1), and in the other we perform a full enumeration to pick the scenario $\boldsymbol{\xi}$ which maximizes $z_n(\boldsymbol{\xi})$ for sample $n \in \mathscr{N}$. We refer to the former as CCG-ENUM-0 and the latter as CCG-ENUM-1. Note that both algorithms are the same as CCG-FIB but only differ in the separation

function called in line 7 of CCG-FIB. Thus, for both CCG-ENUM-0 and CCG-ENUM-1, after performing an enumeration of scenarios for sample n where we are solving the second stage problem, we perform LAZY-ENUMERATION for samples $n' \neq n$. Moreover, we are still applying Theorem 2 but do not leverage the structure of the second stage value function.

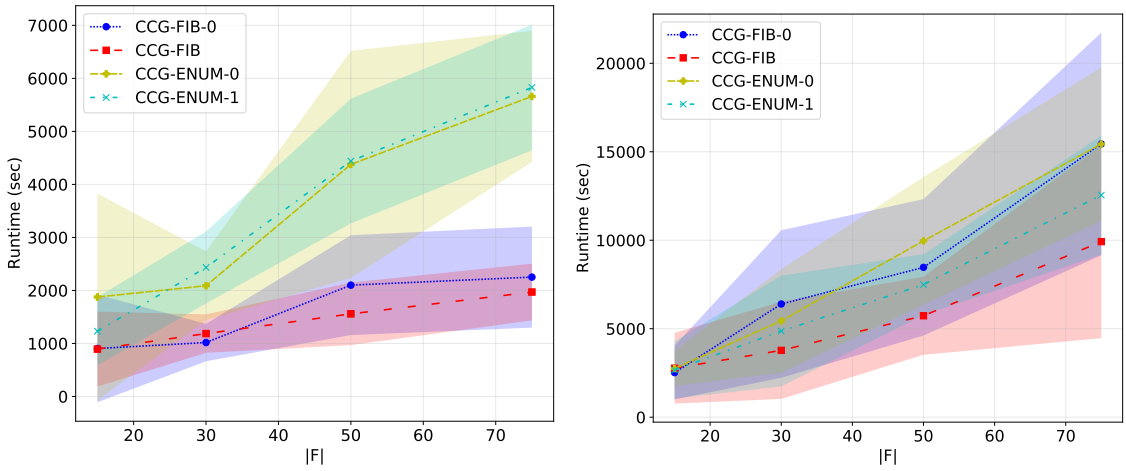
We also define CCG-FIB-0, which is similar to CCG-FIB but with a slight modification to the call to SEPARATION-FIB. Instead of performing Fibonacci search for each combination of components $(\boldsymbol{\xi}_\ell, \xi_r, \xi_a)$, we exit the outer-loop as soon as the Fibonacci search returns a scenario which violates (c1). Note that we can go a step further and terminate the separation step as soon as a scenario which violates (c1) is found during the Fibonacci search, but our experiments showed it is always better to finish the Fibonacci search before terminating the separation step, as it is relatively cheap to finish running Fibonacci search and it might lead to a scenario $\boldsymbol{\xi}$ with a greater objective value $z_n(\boldsymbol{\xi})$. The idea behind CCG-FIB-0 is to get the best of both CCG-ENUM-0 and CCG-FIB, where the separation problem is solved fast in the earlier stages of the algorithm, permitting to quickly generate scenarios, before naturally transitioning to CCG-FIB as the algorithm progresses, avoiding enumerating close to all scenarios in the later stages as would happen in CCG-ENUM-0.

The performance of our methods depends on the sample set and the order of the samples in the set. Thus, for each sample set size and discretization of F, we run 10 experiments where we generate a new random network and a new random set of samples. We provide the average number of scenarios across generated networks for each discretization of F after removing redundant scenarios in Appendix D.3.

For all experiments and computed metrics, we only consider runs where all methods terminated within the time limit. When using any of the four method, for $N = 50$ and $\theta = 0.001$, one run did not terminate for $|F| = 50$ and two runs did not terminate for $|F| = 75$. These were the same three runs for all methods. This is due to the solver stalling when solving (RER) in certain iterations. Such stalls can be circumvented by further adjusting the MIP tolerances or increasing the number of threads available to Gurobi. For $N = 50$ and $\theta = 0.01$, CCG-ENUM-0 did not terminate in one of the runs.

5.2.3 Results

In Figure 2 and Appendix D.5, we plot the average runtimes over the 10 runs as a function of the size of F for each sample set size N and radius of the Wasserstein ball θ. The upper and lower limits of the ribbon are one standard deviation from the average. In general, we observe that the runtimes are faster for the smaller



(a) $N = 10, \theta = 0.01$ (b) $N = 50, \theta = 0.01$

Figure 2: Average runtimes over 10 runs of CCG-FIB, CCG-FIB-0, CCG-ENUM-0, and CCG-ENUM-1 with Wasserstein radius $\theta = 0.01$ and sample set sizes N, with ribbons spanning two standard deviations.

radius. For $N = 10$, CCG-FIB significantly outperforms CCG-ENUM-0 and CCG-ENUM-1, with CCG-FIB-0 being a close second. We note, however, that CCG-FIB-0 has a greater standard deviation of runtimes than CCG-FIB, suggesting the latter is a more stable method. For $\theta = 0.001$, CCG-FIB is about 39% to 40% faster on average than both CCG-ENUM-0 and CCG-ENUM-1 when $|F| = 15$, and is about 69% faster on average when $|F| = 75$. Similarly, for $\theta = 0.01$, CCG-FIB is about 27% faster than CCG-ENUM-1 and 52% faster than CCG-ENUM-0 on average when $|F| = 15$, and is about 66% faster than CCG-ENUM-1 and 65% faster on average than CCG-ENUM-0 when $|F| = 75$. For $N = 50$, the gap between the methods is not as pronounced, but CCG-FIB still outperforms the other three. The two best performing methods are CCG-FIB and CCG-ENUM-1. For $|F| = 75$, on average, CCG-FIB is about 11% faster than CCG-ENUM-1 when $\theta = 0.001$, and about 21% faster when $\theta = 0.01$. In general, we observe that the benefit of using CCG-FIB increases as the size of $|F|$ increases, which is expected as the number of scenarios enumerated is logarithmic in the size of F for CCG-FIB but linear for CCG-ENUM-0 and CCG-ENUM-1.

As we increase the size of the sample set, solving (RER) becomes harder, and a larger portion of time is spent solving (RER) and smaller portion solving the separation step. This explains why the difference between CCG-FIB and CCG-ENUM-1 is not as large when $N = 50$, and why CCG-ENUM-0 and CCG-FIB-0 start to fall behind. For the latter, the separation step is solved fast (at least in the earlier stages), but (RER) is solved much more often, increasing runtime.

To better understand the behavior of the four methods, we record the number of second stage problems

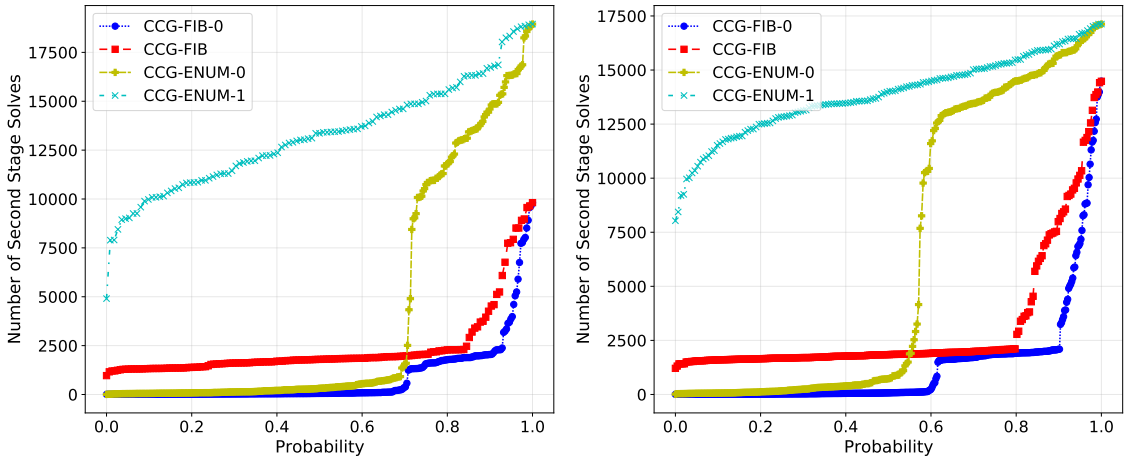
solved during the separation step and the time spent in the separation step, where we aggregate the values across all iterations and the 10 runs to get a distribution for both metrics. We fix $\theta = 0.01$, and plot the cumulative distribution (CDF) of the number of second stage solves for $|F| = 75$ in Figure 3 and for $|F| = 15$ in Appendix D.6. The CDF of the time spent in the separation step is in Appendix D.7.

When comparing CCG-FIB to CCG-ENUM-1, the number of second stage solves in the separation problem is significantly less for the former. We see that in over 80% of the iterations, the number of second stage problems solved is over 50% less in CCG-FIB than in CCG-ENUM-1 when $|F| = 15$. When $|F| = 75$, the gap is much more significant, with CCG-FIB consistently solving fewer than 2,500 second stage problems in over 80% of the iterations, compared to CCG-ENUM-1 ranging between 7,500 and over 15,000. The increase in the number of second stage solves during CCG-FIB occurs in the final phase (i.e. when `final_check` is set to `True`, and `SEPARATION-FIB` is applied to every sample). For $|F| = 75$, we see that the number of second stage solves in CCG-FIB gradually increases from about 2,500 to 10,000 for $N = 10$, and from about 2,500 to 14,000 for $N = 50$. As discussed in Section 4.2, while `SEPARATION-FIB` does lead to significantly fewer second stage solves for one sample, we might end up visiting new scenarios for which we have to compute the second stage problem if applied to all samples. Our experiments confirm our intuition that for larger sample sets, this is more likely to happen. Indeed, the maximum number of second stage solves in CCG-FIB jumps from about 10,000 to 14,000 when increasing the sample size from 10 to 50.

As expected, the CDF for both metrics are very similar. We note that the CDFs for CCG-FIB-0 are always upper-bounded by the CDFs for CCG-FIB, where they start very close to CCG-ENUM-0 and get close to CCG-FIB. This is in line with our idea behind CCG-FIB-0. However, as seen in Figure 2, the increase in the number of times (RER) is solved does not justify it. This analysis also permits us to see how long CCG-FIB spends in the final phase, i.e. the phase during which we perform `SEPARATION-FIB` for each sample (see Section 4). Figures 3 suggests that less than 20% of iterations are in the final phase when looking at all iterations across the 10 runs.

5.3 Case Study: Hurricane Threats in the Gulf of Mexico States

We now turn our focus to the case study of hurricanes threats on the coastal states of the United States first presented in [28]. A plot of the network and the potential landfall nodes can be found in Appendix D.2. We assume all nodes are potential facility locations and potential demand nodes. In this section, we analyze the



(a) $N = 10$, $|F| = 75$ (b) $N = 50$, $|F| = 75$

Figure 3: Cumulative distribution of number of second stage solves during the separation step where $\theta = 0.01$ and $|F| = 75$, for sample set sizes 10 and 50.

solutions obtained from DRO for various radii levels, and from SAA (when $\theta = 0$). We solve problem (2.1), in which case Assumption 2 no longer holds. We include valid inequalities $\sigma_{ij} \leq o_i$ in $\mathscr{C}$ for all i and j, and as in the synthetic instances, we set $\mathscr{B}$ to be empty. Recall that in this version of the model, we are opening facilities and pre-allocating resources in the first stage. We use CCG-FIB to solve all models, where we set the MIP tolerance to 0.12 and adjust it to 1e-3 as described in Section 4. As mentioned in section 3.3, once CCG-FIB terminates, we run one iteration of CCG-ENUM-1 to ensure an exact method. In almost all runs, CCG-ENUM-1 terminates in one iteration and does not generate additional scenarios, suggesting CCG-FIB has a strong computational performance as a heuristic even if Assumption 2 does not hold.

As in the synthetic instances, we use the data and support set described in Sections 5.1.1, 5.1.2 and 5.1.3, and fix $|F| = 50$ where we pick 50 equi-distant values between 0.001 and 0.3. We assume a similar cost structure for transportation costs, where the per-unit cost of transportation from facility i to demand node j is $c_{ij} = 0.0026\,\mathrm{d}_{ij}$, where d_{ij} is the haversine distance between nodes i and j, $i \neq j$. As in the synthetic instances, a node can be both a facility and demand node, and we define the transportation costs of a facility serving its own node to be lower than any other node. We set the fixed-charge cost to be $w_{ij} = \phi c_{ij}$ where $\phi = 5{,}000$. Moreover, we construct a probability distribution for the landfall nodes by using the frequency of hurricanes by region recorded in [28] and [9].

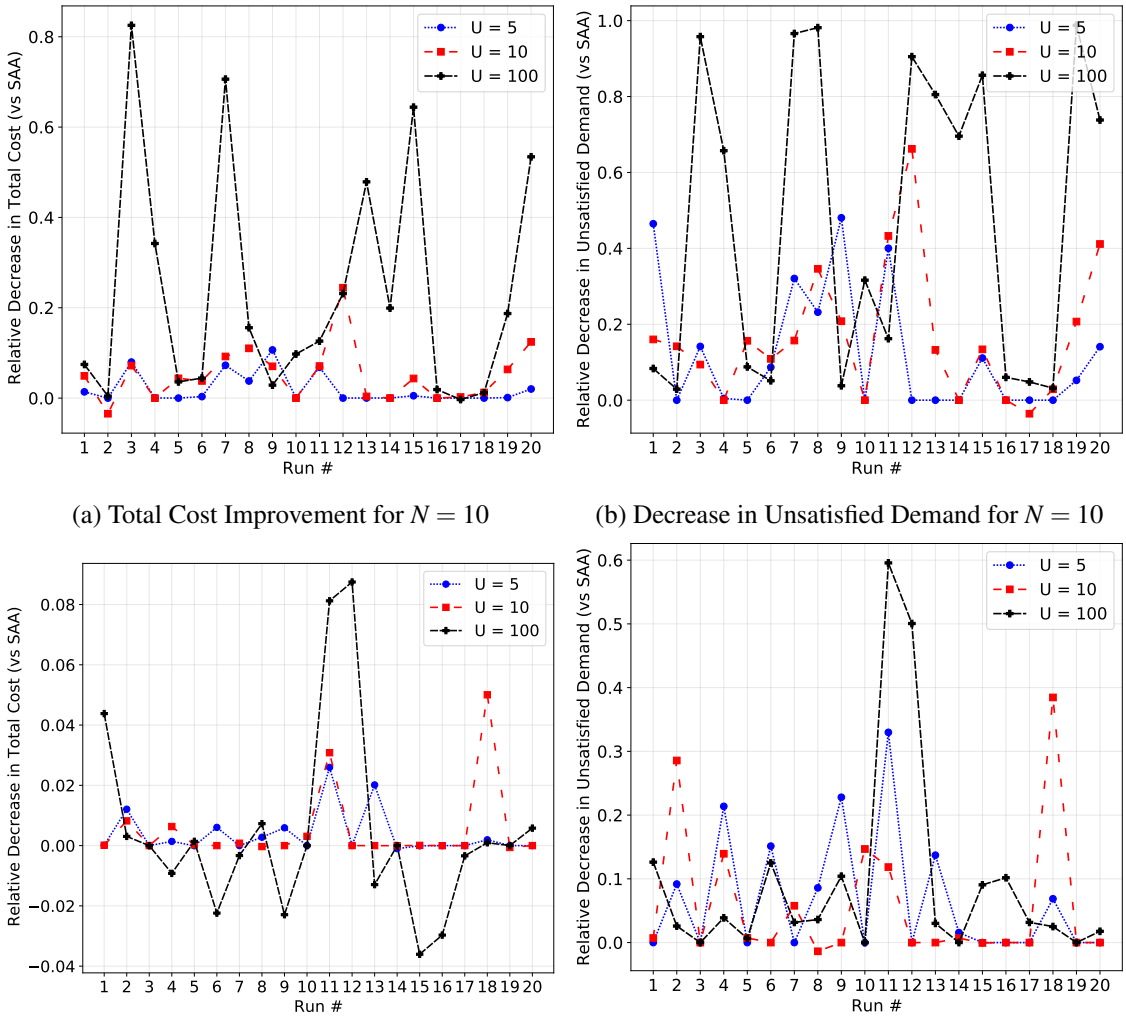
5.3.1 Results

We test the out-of-sample performance of our DRO model for various Wasserstein radii and penalty parameters U. As the radius of the Wasserstein ball increases, we are more risk averse, leading to higher costs in the first stage as we increase the number of facilities open and the amount of resources pre-allocated with the hope of satisfying more demand in more scenarios. As such, there is a natural trade-off between the first stage costs and the penalty incurred for unsatisfied demand U. We analyze the out-of-sample performance of DRO for $U \in \{5, 10, 100\}$. We run tests for $N \in \{10, 50\}$, and for radii $\theta \in \{$1e-5, 5e-5, 1e-4, 5e-4, 1e-3, 5e-3, 0.01, 0.05, 0.1$\}$.

Wasserstein Radius Selection Picking a good radius for the Wasserstein ambiguity set is not straightforward and is the subject of research in the literature. Although there exists some theoretical radii which give probabilistic guarantees on the out-of-sample performance (see [14]), such radii tend to be too large in practice and depend on constants which might be difficult to compute. In practice, the radius selection can be done in an ad-hoc manner through trial and error, or through k-fold cross-validation [14]. Given that we are dealing with very small sets of samples, cross-validation might be ill-suited. To this end, we choose a wide range of radii spanning a few orders of magnitudes for simplicity to illustrate our DRO model.

For each set of experiments, we generate N samples using the constructed probability distribution, solve the DRO model (2.1) with a Wasserstein radius of θ, and record optimal solutions $(\boldsymbol{o}^{\boldsymbol{\theta}}, \boldsymbol{z}^{\boldsymbol{\theta}})$. We then solve $Q(\boldsymbol{o}^{\boldsymbol{\theta}}, \boldsymbol{z}^{\boldsymbol{\theta}}, \boldsymbol{\xi})$ for all $\boldsymbol{\xi} \in \mathscr{S}$ and compute $\mathbb{E}\left[Q(\boldsymbol{o}^{\boldsymbol{\theta}}, \boldsymbol{z}^{\boldsymbol{\theta}}, \boldsymbol{\xi})\right] = \sum_{s \in S} p_s Q(\boldsymbol{o}^{\boldsymbol{\theta}}, \boldsymbol{z}^{\boldsymbol{\theta}}, \boldsymbol{\xi}^{\boldsymbol{s}})$, where $\{p_s\}_{s \in S}$ is the true, constructed, underlying distribution. We repeat this process 20 times for different sets of samples. We define the out-of-sample cost of run $m \in \{1, \ldots, 20\}$ for radius θ as $v_\theta^m = \sum_{i \in I} \left[h_i o_i^{\theta,m} + g z_i^{\theta,m}\right] + \mathbb{E}\left[Q(\boldsymbol{o}^{\theta,m}, \boldsymbol{z}^{\theta,m}, \boldsymbol{\xi})\right]$, where v_0^m corresponds to the out-of-sample cost of SAA. For each run m, we record the non-zero Wasserstein radius which leads to the lowest out-of-sample cost as $\theta^* \neq 0$. In Figures 4a and 4c, we plot the improvement of DRO over SAA in the total out-of-sample cost at θ^*, i.e. we plot $1 - \frac{v_{\theta^*}^m}{v_0^m}$ for all values of U. In Figures 4b and 4d, we plot the relative decrease in the average unsatisfied demand when using DRO over SAA. The average unmet demand for a fixed first-stage solution is defined as $\sum_{s \in \mathscr{S}} p_s \boldsymbol{u}^{s*}$, where $\boldsymbol{u}^{s*}$ is the unmet demand resulting from solving $Q(\boldsymbol{o}^{\theta,m}, \boldsymbol{z}^{\theta,m}, \boldsymbol{\xi}^{\boldsymbol{s}})$.

For $N = 10$, the relative improvement in total cost increases as U increases. For $U = 5$, the improvement over SAA is close to 0 for many runs, with a few runs sitting between about 1.4% and 8% and achieves up to 10% decrease in total cost. The average total cost decrease over the runs is about 2%. For $U = 10$,



(a) Total Cost Improvement for $N = 10$

(b) Decrease in Unsatisfied Demand for $N = 10$

(c) Total Cost Improvement for $N = 50$

(d) Decrease in Unsatisfied Demand for $N = 50$

Figure 4: Relative improvement of DRO over SAA in total out-of-sample cost and unsatisfied demand at θ^*.

DRO performs worse than SAA in run 2 with a 3.5% increase in total out-of-sample cost, but outperforms SAA otherwise, with decreases in out-of-sample cost of up to about 24%, with many runs achieving over 4% decrease. The average total cost decrease is about 5%. Finally, as we increase U to 100, the decrease in the out-of-sample cost becomes significant, achieving over 80% decrease in one of the runs and an average decrease of 23% over all runs. We observe that even when there is no or very little improvement in the out-of-sample cost, DRO consistently satisfies more demand than SAA for all U. For example, for $U = 10$, DRO satisfies up to 66% more demand, with an average of over 16%.

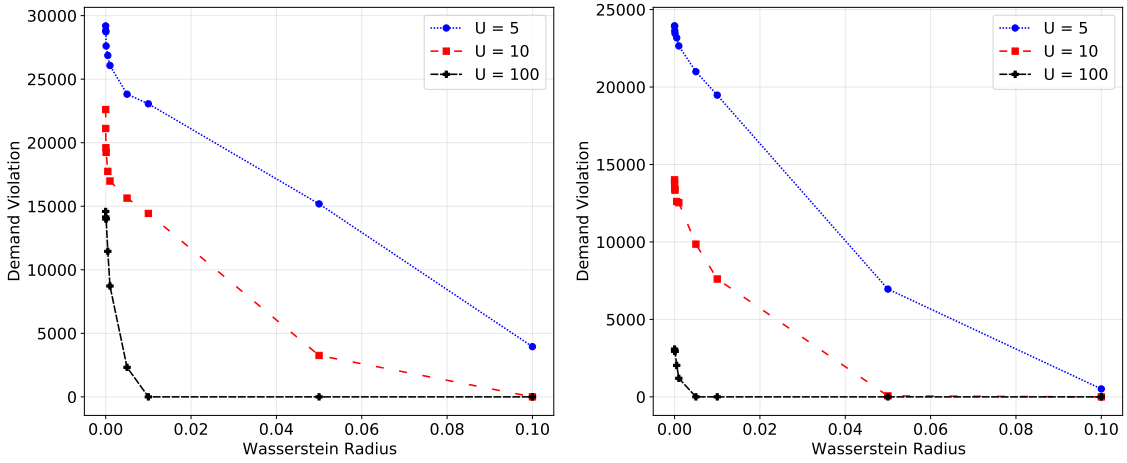
The results tell a different story for $N = 50$. DRO achieves a decrease in out-of-sample cost of about 3% and 5% in two runs when $U = 10$ and up to 2.5% when $U = 5$, but the change is close to 0% in most

runs. For $U = 100$, we observe that DRO achieves similar or lower costs than SAA, with up to 8% decrease, but achieves a higher cost in a few runs, up to 3.6% in total cost. Increasing the penalty parameter U has two potential outcomes: 1) If SAA has a high rate of unmet demand, then DRO has more potential to lead to a decrease in total out-of-sample cost, where a higher value of U means the decrease in the second stage costs outweights the increase in the first stage; 2) If the sample set contains one or a few scenarios with a high demand, a very high penalty forces SAA to open many facilities and pre-allocate a very large quantity of resources at each facility, leading to an already high cost solution and low unsatisfied demand on average across all scenarios. In the second case, there is less opportunity for DRO to decrease the second stage cost and make up for any increase in the first stage cost. We note that the second case is more likely to happen for larger N while the first case is more likely for smaller N as seen in the set of experiments for $N = 10$.

For $N = 50$ and $U = 100$, for run 6, the SAA solution opens 16 facilities and fills almost all of them to full capacity, and the average unsatisfied demand is about 91 units. At $\theta =$ 1e-5, DRO pre-allocates an additional 46,882 units across facilities with remaining capacity, and at $\theta =$ 5e-5, DRO opens a new facility and pre-allocates an additional 73,672 units compared to $\theta =$ 1e-5 (or about 120,555 more units compared to SAA). This leads to a significant increase in the first stage cost, with little benefits (on average) in the second stage. For $\theta =$ 1e-5 and $\theta =$ 5e-5, the average unmet demand is close to 80 and 64, respectively. We observe that for a high value of U, increasing the Wasserstein radius leads to more drastic changes in the solution. For runs 15 and 16, SAA opens five facilities, filling them to almost full capacity. Increasing the radius to $\theta =$ 1e-5 leads to an additional opened facility, and all six facilities at almost full capacity. This suggests that for high values of U, smaller increments in the Wasserstein radius could yield more useful solutions. Indeed, in Figure 5, we can see how the average unmet demand, averaged across the 20 runs, quickly reaches 0 for higher values of U as we increase θ. Regardless, it seems solutions are more unstable as evidenced by the runs where SAA opens over 10 facilities and allocating resources to almost full capacity everywhere. One can also circumvent such solutions by applying a budget on the number of facilities or first stage cost in the set B in the first stage (see model 2.1).

Although the improvement of DRO over SAA in total out-of-sample cost is not significant for $N = 50$, we see that DRO satisfies significantly more demand in the second stage across all runs and all values of U as seen in Figure 4d, similar to the sets of experiments for $N = 10$. DRO satisfies up to 30% more demand when $U = 5$, up to 38% when $U = 10$, and up to 60% when $U = 100$.

Finally, we report the first stage solutions obtained for one run when $N = 10$ and $N = 50$ in Tables 1 and



(a) $N = 10$ (b) $N = 50$

Figure 5: Average unsatisfied demand as a function of the radius of the Wasserstein ball θ for $N = 10$ and $N = 50$, averaged across all runs.

2, respectively, to illustrate how solutions might evolve as the radius of the Wasserstein ball increases. The values in the tables represent the number of resources allocated at the facilities. A non-zero value indicates an open facility, and we exclude facilities with 0 resources allocated across all Wasserstein radii θ. We can see how the model opens more facilities and pre-allocate more resources as the radius increases, as expected. For $N = 10$, the model initially only opens one facility in Biloxi at $\theta = 0$ and one in Mobile at $\theta =$ 1e-5, hedging against disasters in the Louisiana area, before opening a facility in Jacksonville when $\theta =$ 5e-5 and Savannah when $\theta =$ 5e-4. For $N = 50$, the model covers similar areas of the network, but pre-allocates more resources from $\theta = 0$ to $\theta =$1e-3, with an additional facility open for radii 0 and 1e-5. For radii 5e-3 and 0.01, less resources are pre-allocated when $N = 50$, however. To more easily visualize a few first stage solutions, we plot open facilities from Table 1 for $\theta \in \{0,\ 5\text{e-}5,\ 5\text{e-}3\}$ in Figure 6.

We note that in practice, the best radius is of course unknown. However, it is generally the case in our experiments that smaller changes in the radius and in the solution of SAA lead to better out-of-sample cost. As we increase the radius, once there is a drastic jump in the first stage cost caused by a large amount of additional resources pre-allocated or additional facilities opened, the out-of-sample cost tends to get worse (depending on U), as the first stage costs incurred outweigh the decrease in the second stage costs. The radius of the Wasserstein ball provides a convenient lever of risk-aversion to decision-makers to control the trade-off between first stage costs and potential second stage costs.

Table 1: First stage solutions of run 20, $N = 10$ and $U = 10$ for various Wasserstein radii θ.

Facility \ θ	0	1e-05	5e-05	1e-4	5e-4	1e-3	5e-3	0.01
4	0	0	0	0	0	0	131837	131837
10	0	0	0	0	0	0	0	131837
12	0	0	91926	91926	91926	91926	131837	0
13	91926	0	0	0	0	0	0	0
14	0	131837	0	0	0	0	0	0
22	0	0	0	0	63543	0	0	0
25	0	0	63543	63543	0	63543	125521	125499

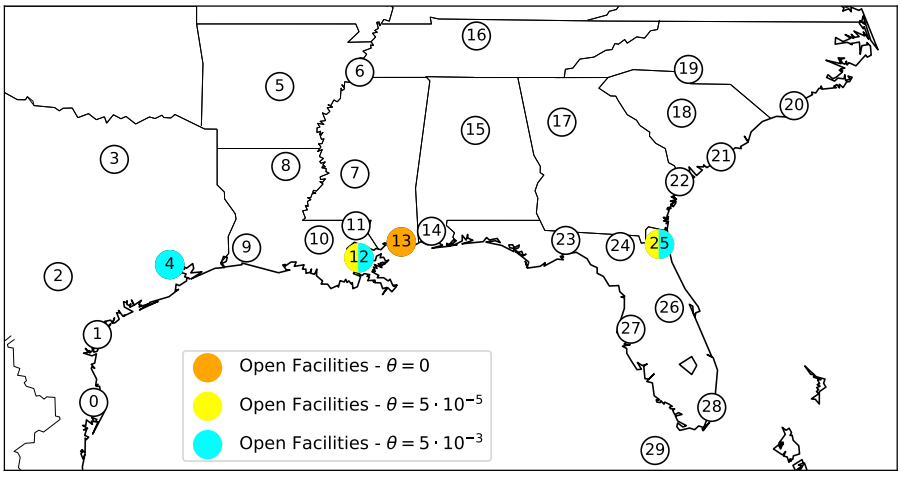


Figure 6: Open facilities on run 20, $N = 10$, and $U = 10$ for $\theta \in \{0,\ 5\text{e-}5,\ 5\text{e-}3\}$

Table 2: First stage solutions of run 10, $N = 50$ and $U = 10$ for various Wasserstein radii θ.

Facility \ θ	0	1e-05	5e-05	1e-4	5e-4	1e-3	5e-3	0.01
10	0	0	0	0	0	0	100942	125923
11	69438	89299	89299	0	71327	71327	0	0
13	0	0	0	90491	0	0	0	0
23	0	0	0	0	127448	127448	47420	0
24	0	0	0	0	0	0	94128	78615
25	91345	92536	92536	91345	0	0	0	88031

6 Conclusions

The lack of data in natural disaster management applications can lead to optimistic solutions and poor out-of-sample performance when using classical stochastic programming methods such as SAA. In this paper, we tackle a two-stage DRO model under the Wasserstein ambiguity set, where the first stage is a facility location problem, where we decide facility locations and the amount of resources to pre-allocate, and where the second stage is a fixed-charge transportation problem. We develop an efficient column and constraint generation algorithm, where we leverage the structure of our support set and second stage value function to efficiently solve the separation problem. Specifically, we show conditions under which the second stage value function is concave with respect to the intensity component of our random vector and show how our results extend to the case where the second stage is a fixed-charge network flow problem. We also provide guidance on numerical improvements which can significantly impact computational efficiency. We conduct extensive computational experiments on synthetic instances, showing strong computational performance compared to (mostly) classical column and constraint generation algorithms, and analyze the solutions obtained from our DRO model on a case study of hurricane threats on the coastal states of the United States. We show that our DRO model achieves lower out-of-sample costs than SAA in many instances, and consistently satisfies significantly more demand after a disaster occurs, even when the out-of-sample costs are comparable. Given the performance of DRO under the Wasserstein set for hurricane disasters, it would be worth exploring its benefits for other types of disasters. Finally, although we can have joint concavity with respect to intensities associated with multiple landfalls, it is unclear how to efficiently search over multiple intensities.

Acknowledgment

The authors' work was partially supported by the Office of Naval Research (N00014-18-1-2075). The authors thank the review team for their valuable and timely feedback, which helped improve the manuscript.

References

[1] Manish Bansal, Kuo-Ling Huang, and Sanjay Mehrotra. "Decomposition algorithms for two-stage distributionally robust mixed binary programs". In: *SIAM Journal on Optimization* 28.3 (2018), pp. 2360–2383.

[2] Vedat Bayram and Hande Yaman. "Shelter location and evacuation route assignment under uncertainty: A benders decomposition approach". In: *Transportation science* 52.2 (2018), pp. 416–436.

[3] Aharon Ben-Tal, Omar El Housni, and Vineet Goyal. "A tractable approach for designing piecewise affine policies in two-stage adjustable robust optimization". In: *Mathematical Programming* 182.1 (2020), pp. 57–102.

[4] Aharon Ben-Tal and Arkadi Nemirovski. "Robust optimization–methodology and applications". In: *Mathematical Programming* 92.3 (2002), pp. 453–480.

[5] Aharon Ben-Tal et al. "Adjustable robust solutions of uncertain linear programs". In: *Mathematical programming* 99.2 (2004), pp. 351–376.

[6] Aharon Ben-Tal et al. "Robust solutions of optimization problems affected by uncertain probabilities". In: *Management Science* 59.2 (2013), pp. 341–357.

[7] Dimitris Bertsimas and Vineet Goyal. "On the power and limitations of affine policies in two-stage adaptive optimization". In: *Mathematical programming* 134.2 (2012), pp. 491–531.

[8] Dimitris Bertsimas, Vishal Gupta, and Nathan Kallus. "Robust sample average approximation". In: *Mathematical Programming* 171.1 (2018), pp. 217–282.

[9] Eric S Blake et al. "The deadliest, costliest, and most intense United States tropical cyclones from 1851 to 2004 (and other frequently requested hurricane facts)". In: *NOAA Technical Memorandum NWS TPC-4* (2007).

[10] Giuseppe C Calafiore. "Ambiguous risk measures and optimal robust portfolios". In: *SIAM Journal on Optimization* 18.3 (2007), pp. 853–877.

[11] Zhi Chen, Melvyn Sim, and Huan Xu. "Distributionally robust optimization with infinitely constrained ambiguity sets". In: *Operations Research* 67.5 (2019), pp. 1328–1344.

[12] Jyotirmoy Dalal and Halit Üster. "Robust emergency relief supply planning for foreseen disasters under evacuation-side uncertainty". In: *Transportation science* 55.3 (2021), pp. 791–813.

[13] Erick Delage and Yinyu Ye. "Distributionally robust optimization under moment uncertainty with application to data-driven problems". In: *Operations research* 58.3 (2010), pp. 595–612.

[14] Peyman Mohajerin Esfahani and Daniel Kuhn. "Data-driven distributionally robust optimization using the Wasserstein metric: Performance guarantees and tractable reformulations". In: *Mathematical Programming* 171.1-2 (2018), pp. 115–166.

[15] David E Ferguson. "Fibonaccian searching". In: *Communications of the ACM* 3.12 (1960), p. 648.

[16] Rui Gao and Anton J Kleywegt. "Distributionally robust stochastic optimization with Wasserstein distance". In: *arXiv preprint arXiv:1604.02199* (2016).

[17] Emilia Grass and Kathrin Fischer. "Two-stage stochastic programming in disaster management: A literature survey". In: *Surveys in Operations Research and Management Science* 21.2 (2016), pp. 85–100.

[18] Grani A Hanasusanto, Daniel Kuhn, and Wolfram Wiesemann. "K-adaptability in two-stage distributionally robust binary programming". In: *Operations Research Letters* 44.1 (2016), pp. 6–11.

[19] Grani A Hanasusanto, Daniel Kuhn, and Wolfram Wiesemann. "K-adaptability in two-stage robust binary programming". In: *Operations Research* 63.4 (2015), pp. 877–891.

[20] Kibaek Kim. "Dual Decomposition of Two-Stage Distributionally Robust Mixed-Integer Programming under the Wasserstein Ambiguity Set". In: *Preprint manuscript* (2020).

[21] Anton J Kleywegt, Alexander Shapiro, and Tito Homem-de-Mello. "The sample average approximation method for stochastic discrete optimization". In: *SIAM Journal on Optimization* 12.2 (2002), pp. 479–502.

[22] Burak Kocuk. "Conic reformulations for Kullback-Leibler divergence constrained distributionally robust optimization and applications". In: *arXiv preprint arXiv:2007.05966* (2020).

[23] Daniel Kuhn et al. "Wasserstein distributionally robust optimization: Theory and applications in machine learning". In: *Operations Research & Management Science in the Age of Analytics*. INFORMS, 2019, pp. 130–166.

[24] Christopher W. Landsea and James L. Franklin. "Atlantic Hurricane Database Uncertainty and Presentation of a New Database Format". In: *Monthly Weather Review* 141.10 (2013), pp. 3576–3592. DOI: 10.1175/mwr-d-12-00254.1.

[25] Yuchen Li, Daniel Zhuoyu Long, and Jianghua Zhang. "Data-driven distributionally robust emergency facility deployment under demand location uncertainty". In: *Available at SSRN 4093823* (2022).

[26] David Love and Güzin Bayraksan. "Phi-divergence constrained ambiguous stochastic programs for data-driven optimization". In: *Technical report, Department of Integrated Systems Engineering, The Ohio State University, Columbus, Ohio* (2015).

[27] Engineering National Academies of Sciences, Medicine, et al. "Strengthening post-hurricane supply chain resilience: Observations from Hurricanes Harvey, Irma, and Maria". In: (2020).

[28] Carmen G Rawls and Mark A Turnquist. "Pre-positioning of emergency supplies for disaster response". In: *Transportation research part B: Methodological* 44.4 (2010), pp. 521–534.

[29] Monir Sabbaghtorkan, Rajan Batta, and Qing He. "Prepositioning of assets and supplies in disaster operations management: Review and research gap identification". In: *European Journal of Operational Research* 284.1 (2020), pp. 1–19.

[30] Karmel S Shehadeh and Emily L Tucker. "A Distributionally Robust Optimization Approach for Location and Inventory Prepositioning of Disaster Relief Supplies". In: *arXiv preprint arXiv:2012.05387* (2020).

[31] Anirudh Subramanyam, Chrysanthos E Gounaris, and Wolfram Wiesemann. "K-adaptability in two-stage mixed-integer robust optimization". In: *Mathematical Programming Computation* 12.2 (2020), pp. 193–224.

[32] German A Velasquez, Maria E Mayorga, and Osman Y Özaltın. "Prepositioning disaster relief supplies using robust optimization". In: *IISE Transactions* 52.10 (2020), pp. 1122–1140.

[33] Bram Verweij et al. "The sample average approximation method applied to stochastic routing problems: a computational study". In: *Computational Optimization and Applications* 24.2-3 (2003), pp. 289–333.

[34] Weiqiao Wang et al. "Two-stage distributionally robust programming based on worst-case mean-CVaR criterion and application to disaster relief management". In: *Transportation Research Part E: Logistics and Transportation Review* 149 (2021), p. 102332.

[35] Wolfram Wiesemann, Daniel Kuhn, and Melvyn Sim. "Distributionally robust convex optimization". In: *Operations Research* 62.6 (2014), pp. 1358–1376.

[36] David Wozabal. "A framework for optimization under ambiguity". In: *Annals of Operations Research* 193.1 (2012), pp. 21–47.

[37] Weijun Xie. "Tractable Reformulations of Distributionally Robust Two-stage Stochastic Programs with ∞-Wasserstein Distance". In: *arXiv preprint arXiv:1908.08454* (2019).

[38] Yongjian Yang et al. "Distributionally robust multi-period location-allocation with multiple resources and capacity levels in humanitarian logistics". In: *European Journal of Operational Research* 305.3 (2023), pp. 1042–1062.

[39] Xian Yu and Siqian Shen. "Multistage distributionally robust mixed-integer programming with decision-dependent moment-based ambiguity sets". In: *Mathematical Programming* (2020), pp. 1–40.

[40] Yuefei Yuan, Qiankun Song, and Bo Zhou. “A Wasserstein distributionally robust chance constrained programming approach for emergency medical system planning problem”. In: *International Journal of Systems Science* 53.10 (2022), pp. 2136–2148.

[41] Chaoyue Zhao. *Data-driven risk-averse stochastic program and renewable energy integration*. PhD Thesis, University of Florida, 2014.

A Proofs

A.1 Proof of Theorem 1

Proof. Using (W'), the inner maximization problem of (DRO) can be written as

$$\begin{array}{ll} \max\limits_{\boldsymbol{p}} & \sum\limits_{s\in\mathscr{S}} p_s Q(\boldsymbol{x},\boldsymbol{\xi}^s) \\ \text{s.t.} & \min\limits_{\boldsymbol{q}\geq 0}\left\{ \sum\limits_{s\in\mathscr{S}}\sum\limits_{n\in\mathscr{N}} \frac{1}{N}\mathrm{d}(\boldsymbol{\xi}^s,\hat{\boldsymbol{\xi}}^n) q_s^n : \begin{array}{l} \sum_{s\in\mathscr{S}} q_s^n = 1,\ \forall n\in\mathscr{N} \\ \sum_{n\in\mathscr{N}} \frac{1}{N} q_s^n = p_s,\ \forall s\in\mathscr{S} \end{array} \right\} \leq \theta \end{array}$$

which is equivalent to

$$\begin{array}{lll} \max\limits_{\boldsymbol{q}} & \sum\limits_{s\in\mathscr{S}}\sum\limits_{n\in\mathscr{N}} \frac{1}{N} q_s^n Q(\boldsymbol{x},\boldsymbol{\xi}^s) & \\ \text{s.t.} & \sum\limits_{s\in\mathscr{S}}\sum\limits_{n\in\mathscr{N}} \frac{1}{N}\mathrm{d}(\boldsymbol{\xi}^s,\hat{\boldsymbol{\xi}}^n) q_s^n \leq \theta, & \\ & \sum\limits_{s\in\mathscr{S}} q_s^n = 1 & \forall n\in\mathscr{N}, \\ & q_s^n \geq 0, & \forall s\in\mathscr{S},\ \forall n\in\mathscr{N}. \end{array} \tag{A.1}$$

Its dual is

$$\begin{array}{ll} \min\limits_{\lambda,\alpha} & \theta\lambda + \frac{1}{N}\sum\limits_{n\in\mathscr{N}} \alpha_n \\ \text{s.t.} & \alpha_n \geq Q(\boldsymbol{x},\boldsymbol{\xi}^s) - \lambda\,\mathrm{d}(\boldsymbol{\xi}^s,\hat{\boldsymbol{\xi}}^n),\ \forall s\in\mathscr{S},\ \forall n\in\mathscr{N}, \\ & \lambda \geq 0. \end{array}$$

We then have

$$\begin{array}{ll} \min\limits_{\boldsymbol{x},\lambda} & \boldsymbol{c}^\top\boldsymbol{x} + \theta\lambda + \frac{1}{N}\sum\limits_{n=1}^{N} \alpha_n \\ \text{s.t.} & \alpha_n \geq Q(\boldsymbol{x},\boldsymbol{\xi}^s) - \lambda\,\mathrm{d}(\boldsymbol{\xi}^s,\hat{\boldsymbol{\xi}}^n),\ s\in\mathscr{S}, n\in\mathscr{N}, \\ & \boldsymbol{x}\in X, \\ & \lambda \geq 0 \end{array}$$

which is equivalent to (ER). □

A.2 Proof of Theorem 2

Proof. Recall from the proof of Theorem 1, we can write the inner maximization as

$$\begin{aligned} \max_{\boldsymbol{q}} \quad & \sum_{s\in\mathscr{S}}\sum_{n\in\mathscr{N}} \frac{1}{N} q_s^n Q(\boldsymbol{x},\boldsymbol{\xi}^s) \\ \text{s.t.} \quad & \sum_{s\in\mathscr{S}}\sum_{n\in\mathscr{N}} \frac{1}{N} \mathrm{d}(\boldsymbol{\xi}^s,\hat{\boldsymbol{\xi}}^n) q_s^n \le \theta, \\ & \sum_{s\in\mathscr{S}} q_s^n = 1 \qquad\qquad \forall n\in\mathscr{N}, \\ & q_s^n \ge 0, \qquad\qquad \forall s\in\mathscr{S},\ \forall n\in\mathscr{N}. \end{aligned} \tag{A.2}$$

where q_s^n is the conditional probability of scenario $\boldsymbol{\xi}^s$ given we have observed $\hat{\boldsymbol{\xi}}^n$, the sampled scenario.

In (A.2), we are moving weight q_s^n from a sample $\hat{\boldsymbol{\xi}}^n$ to a scenario $\boldsymbol{\xi}^s$ for all s and n, with the objective of maximizing the expected cost of the second stage. Consider a sample $\hat{\boldsymbol{\xi}}^n$ and a scenario $\boldsymbol{\xi}\in\bar{\Xi}_n$. We show that there exists a scenario $\boldsymbol{\xi'}$ which dominates $\boldsymbol{\xi}$, in the sense that $\mathrm{d}(\hat{\boldsymbol{\xi}},\boldsymbol{\xi'}) < \mathrm{d}(\hat{\boldsymbol{\xi}},\boldsymbol{\xi})$ and $Q(\boldsymbol{x},\boldsymbol{\xi'}) \ge Q(\boldsymbol{x},\boldsymbol{\xi})$. Let $\boldsymbol{\xi'} = (\boldsymbol{\xi}_{\boldsymbol{\ell}}, \hat{\xi}_r^n, \xi_a, \hat{\xi}_f^n)$. Then we have $Q(\boldsymbol{x},\boldsymbol{\xi'}) \ge Q(\boldsymbol{x},\boldsymbol{\xi})$ for all $\boldsymbol{x}\in X$, since $\xi_f \le \hat{\xi}_f^n$ and $\xi_r \le \hat{\xi}_r^n$, where at least one of the two inequalities holds strictly, and $\boldsymbol{\xi}_{\boldsymbol{\ell}} = \boldsymbol{\xi'}_{\boldsymbol{\ell}}$ and $\xi_a = \xi'_a$. Moreover, we have $\mathrm{d}(\hat{\boldsymbol{\xi}}^n,\boldsymbol{\xi'}) < \mathrm{d}(\hat{\boldsymbol{\xi}}^n,\boldsymbol{\xi})$ since:

$$\begin{aligned} \mathrm{d}(\hat{\boldsymbol{\xi}}^n,\boldsymbol{\xi'}) &= ||\hat{\boldsymbol{\xi}}^{\boldsymbol{n}}_{\boldsymbol{\ell}} - \boldsymbol{\xi}_{\boldsymbol{\ell}}||_2^2 + (\hat{\xi}_a^n - \xi_a)^2 \\ \mathrm{d}(\hat{\boldsymbol{\xi}}^n,\boldsymbol{\xi}) &= ||\hat{\boldsymbol{\xi}}^{\boldsymbol{n}}_{\boldsymbol{\ell}} - \boldsymbol{\xi}_{\boldsymbol{\ell}}||_2^2 + (\hat{\xi}_r^n - \xi_r)^2 + (\hat{\xi}_a^n - \xi_a)^2 + (\hat{\xi}_f^n - \xi_f)^2 \end{aligned}$$

where one of $(\hat{\xi}_r^n - \xi_r)^2$ or $(\hat{\xi}_f^n - \xi_f)^2$ must be non-zero because $\boldsymbol{\xi}\in\bar{\Xi}_n$. □

A.3 Proof of Theorem 3 and Corollary 1

Proof. First note that each demand node is served by at most one facility. Given a set of open facilities, let $a = \{i_1, i_2, \dots\}$ be a set of assignments from facilities to nodes with positive demand, where i_j is the facility assigned to node j, and let A be the set of all possible assignments between facilities and demand nodes satisfying the cardinality constraints (3.2a). Let $\hat{J}$ be the set of demand nodes in $J(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a)$ that are not served by a facility in a set of assignments a. Let P_j be the population at node j. Define $f(\xi_f)$ to be the objective of the separation problem $z_n(\xi)$ as a function of ξ_f, where $\boldsymbol{\xi}_{\boldsymbol{\ell}}$, ξ_r and ξ_a are fixed.

First note that as we increase ξ_f for a fixed set of assignments $a\in A$, the fixed-charge cost component remains constant at $w_{i_j j}$, and the transportation cost is equal to $c_{ki_jj}\Pi_j^k(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f)P_j$ for each served node j and commodity $k\in K$. For nodes that are not served, the cost is $U\Pi_j^k(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f)P_j$. The objective value of the second stage is then

$$\sum_{j\in J(\boldsymbol{\xi}_{\boldsymbol{\ell}},\xi_r,\xi_a)\setminus\hat{J}} \left[w_{i_j j} + \sum_{k\in K} c_{ki_jj}\Pi_j^k(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f)P_j \right] + U\sum_{j\in\hat{J}} \Pi_j^k(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f)P_j. \tag{A.3}$$

Note that (A.3) is concave with respect to ξ_f by Assumption 1. We then have

$$\begin{aligned} f(\xi_f) &= Q(\boldsymbol{x},\boldsymbol{\xi}) - \lambda\,\mathrm{d}(\boldsymbol{\xi},\hat{\boldsymbol{\xi}}^n) \\ &= \min_{a\in A} \left\{ \sum_{j\in J(\boldsymbol{\xi}_{\boldsymbol{\ell}},\xi_r,\xi_a)\setminus\hat{J}} \left[w_{i_j j} + \sum_{k\in K} c_{ki_jj}\Pi_j^k(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f)P_j \right] + U\sum_{j\in\hat{J}} \Pi_j^k(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f)P_j \right\} - \lambda\,\mathrm{d}(\boldsymbol{\xi},\hat{\boldsymbol{\xi}}^n), \end{aligned}$$

where the first term is the minimum of concave functions and is therefore concave with respect to ξ_f, and $-\lambda\, \mathrm{d}(\boldsymbol{\xi}, \hat{\boldsymbol{\xi}}^n)$ is concave since $\lambda \geq 0$. We conclude that $f(\xi_f)$ is concave. □

B Second-Stage Fixed-Charge Network Flow

We show that our results extend to the case where the second stage problem is an uncapacitated fixed-charge network flow problem. Given a graph $G = (\mathscr{V}, \mathscr{A})$, we define all nodes incident to outgoing and incoming arcs as $\delta^+(e_1) = \{e_2 : (e_1, e_2) \in \mathscr{A}\}$ and $\delta^-(e_2) = \{e_1 : (e_1, e_2) \in \mathscr{A}\}$, respectively. In line with Assumption 2, we assume each facility sends an uncapacitated vehicle to serve demand nodes, and can serve up to k_i demand nodes. Variable σ_{ij} is 1 if facility i is serving demand node j. We introduce a new variable $\boldsymbol{\beta}$ where $\beta_{e_1e_2}$ is 1 if arc (e_1, e_2) is used. Variable $t^{ik}_{e_1e_2}$ represents the amount of commodity k transported through arc (e_1, e_2) by facility i's vehicle. Finally, let M be a sufficiently large number. We write the second stage problem as:

$$
\begin{aligned}
\min_{\boldsymbol{\sigma}, \boldsymbol{\beta}, \boldsymbol{t}, \boldsymbol{u}} \quad & \sum_{(e_1,e_2)\in\mathscr{A}} \left[w_{e_1e_2}\beta_{e_1e_2} + \sum_{i\in I}\sum_{k\in K} c_{ke_1e_2} t^{ik}_{e_1e_2} \right] + U \sum_{j\in J}\sum_{k\in K} u^k_j && && \text{(B.1a)}\\
\text{s.t.} \quad & \sum_{j\in J} \sigma_{ij} \leq k_i, && \forall i \in I, && \text{(B.1b)}\\
& \sigma_{ij} \geq \frac{1}{M} \sum_{e_1\in\delta^-(j)} t^{ik}_{e_1 j}, && \forall i \in I,\ \forall j \in J, && \text{(B.1c)}\\
& \sum_{e_2\in\delta^+(i)} t^{ik}_{ie_2} - \sum_{e_1\in\delta^-(i)} t^{ik}_{e_1 i} = \sum_{j\in J} d(\boldsymbol{\xi}^s)_{kj}\sigma_{ij} && \forall i \in I,\ \forall k \in K, && \text{(B.1d)}\\
& \sum_{e_2\in\delta^+(j)} t^{ik}_{je_2} - \sum_{e_1\in\delta^-(j)} t^{ik}_{e_1 j} = -d(\boldsymbol{\xi}^s)_{kj}\sigma_{ij} && \forall i \in I,\ \forall j \in J,\ \forall k \in K, && \text{(B.1e)}\\
& u^k_j \geq d(\boldsymbol{\xi}^s)_{kj} - M\sum_{i\in I}\sigma_{ij} && \forall j \in J,\ \forall k \in K, && \text{(B.1f)}\\
& t^{ik}_{e_1e_2} \leq M\beta_{e_1e_2}, && \forall (e_1,e_2) \in \mathscr{A},\ \forall i \in I,\ \forall k \in K, && \text{(B.1g)}\\
& \sigma_{ij} \leq o_i && \forall i \in I,\ \forall j \in J, && \text{(B.1h)}\\
& t^k_{ij} \geq 0, && \forall i \in I,\ \forall j \in J,\ \forall k \in K, &&\\
& \sigma_{ij} \in \{0,1\}, && \forall i \in I,\ \forall j \in J, &&\\
& u^k_j \geq 0, && \forall j \in J,\ \forall k \in K, &&\\
& (\boldsymbol{\sigma}, \boldsymbol{t}, \boldsymbol{u}) \in C. &&
\end{aligned}
$$

Constraints (B.1b) ensure a facility i serves at most k_i demand nodes j as in (3.1); constraints (B.1c) set σ_{ij} to 1 if facility i is serving node j; constraints (B.1d) are network flow constraints at facility nodes, ensuring $d(\boldsymbol{\xi}^s)_{kj}$ units of commodity k leaves the facility for each demand node j it is serving; constraints (B.1e) ensure facility i's vehicle delivers $d(\boldsymbol{\xi}^s)_{kj}$ units of commodity k to node j if i is serving j; constraint (B.1f) ensures $u^k_j = d(\boldsymbol{\xi}^s)_{kj}$ if demand node $j \in J$ is not served by any facility $i \in I$, i.e. $\sum_{i\in I}\sigma_{ij} = 0$; constraints (B.1g) ensure $\beta_{e_1e_2} = 1$ if $t^{ik}_{e_1e_2} > 0$ for any commodity k and facility i; finally constraints (B.1h) ensure facility i can serve demand node j only if facility i is open.

We have the following similar result.

Theorem 4. *Under Assumption 1, if the second stage problem corresponds to (B.1), then for fixed first stage decision vector x and $\lambda \geq 0$, $Q(x, \boldsymbol{\xi})$ and $z_n(\boldsymbol{\xi})$ are concave with respect to ξ_f for a fixed landfall $\boldsymbol{\xi}_\ell$, radius of impact ξ_r, and angle ξ_a.*

Proof. The proof is very similar to the proof of Theorem 3 and its corollary. Let $\hat{I}$ be the set of open facilities. For an open facility $i \in \hat{I}$, let $J_i \subseteq J(\boldsymbol{\xi}_\ell, \xi_r, \xi_a)$ be the set of demand nodes it serves, and let $\rho_i = \{\rho_{ij}\}_{j \in J_i}$ be a set of paths from facility i to nodes $j \in J_i$, where ρ_{ij} is a set of edges from facility i to demand node j. Thus, $\{\rho_i\}_{i \in \hat{I}}$ corresponds to a solution to the second stage problem, representing paths from each open facility $i \in \hat{I}$ to the demand nodes they each serve. Let ρ be the set of all possible $\{\rho_i\}_{i \in \hat{I}}$ satisfying constraint (B.1b). Recall that $\hat{J}$ is the set of demand nodes in $J(\boldsymbol{\xi}_\ell, \xi_r, \xi_a)$ that are not served by any facility, and $f(\xi_f)$ corresponds to $z_n(\boldsymbol{\xi})$ as a function of ξ_f, where $\boldsymbol{\xi}_\ell$, ξ_r and ξ_a are fixed.

For a fixed set of paths for each facility $i \in \hat{I}$, i.e. for a fixed $\{\rho_i\}_{i \in \hat{I}}$, the second stage cost is:

$$\sum_{i \in \hat{I}} \sum_{j \in J_i} \sum_{e \in \rho_{ij}} \left[w_{e_1 e_2} + \sum_{k \in K} c_{k e_1 e_2} \Pi_j^k(\boldsymbol{\xi}_\ell, \xi_r, \xi_a, \xi_f) P_j \right] + U \sum_{j \in \hat{J}} \Pi_j^k(\boldsymbol{\xi}_\ell, \xi_r, \xi_a, \xi_f) P_j, \tag{B.2}$$

which is concave with respect to ξ_f. We then have

$$\begin{aligned} f(\xi_f) &= Q(\boldsymbol{x}, \boldsymbol{\xi}) - \lambda \, \mathrm{d}(\boldsymbol{\xi}, \hat{\boldsymbol{\xi}}^n) \\ &= \min_{\{\rho_i\}_{i \in \hat{I}} \in \rho} \left\{ \sum_{i \in \hat{I}} \sum_{j \in J_i} \sum_{e \in \rho_{ij}} \left[w_{e_1 e_2} + \sum_{k \in K} c_{k e_1 e_2} \Pi_j^k(\boldsymbol{\xi}_\ell, \xi_r, \xi_a, \xi_f) P_j \right] + U \sum_{j \in \hat{J}} \Pi_j^k(\boldsymbol{\xi}_\ell, \xi_r, \xi_a, \xi_f) P_j \right\} \\ &\qquad - \lambda \, \mathrm{d}(\boldsymbol{\xi}, \hat{\boldsymbol{\xi}}^n), \end{aligned}$$

and the result follows. □

C Pseudo-Code of Algorithms

C.1 Fibonacci Search

In Algorithm 3, we define Ω to be the array of scenarios in increasing order of $\xi_f \in F$ and an array Θ of Fibonacci numbers up to the q^{th} Fibonacci number, such that $q \geq |\Omega| - 1$. Our sequence of Fibonacci numbers starts with 0, 1 and 1. Let $m = |\Theta|$ be the size of this array. We index the i^{th} element of Ω and Θ by Ω_i and Θ_i, and start the indexing at 0. In lines 4-8, we handle the corner cases where there are only one or two scenarios in Ω, in which case it is a simple check. Throughout the search, we keep track of three variables: ℓ, i_a and u, where $\ell \leq i_a \leq u$. These are indices of Ω which represent intervals where the maximum might be. The three variables are Fibonacci numbers and the idea is to maintain a constant ratio of the interval sizes, such that $\frac{u - i_a}{i_a - \ell}$ is close to the golden-ratio. In order to search the intervals, we define a fourth index i_b between i_a and u. We then compare $z_n(\Omega_{i_a})$ to $z_n(\Omega_{i_b})$. If $z_n(\Omega_{i_a}) > z_n(\Omega_{i_b})$, then we know the maximum is somewhere between ℓ and i_b, otherwise, it is between i_a and u. We update the indices accordingly in lines 13-21. In order to maintain the same interval ratios as the algorithm progresses, i_b is also chosen to be a Fibonacci number where $i_b = \ell + u - i_a$. We repeat this process until $i_a = i_b$. Note that this will find the maximum if it is not at one of the initial endpoints (i.e. 0 or Θ_{m-1}). If either $\ell = 0$ or $u = \Theta_{m-1}$ after exiting the while-loop, we must do a manual check against i_a. This is done in lines 23-29.

C.2 Separation Scheme

In Algorithm 4, we enumerate components $(\boldsymbol{\xi}_\ell, \xi_r, \xi_a)$ in $L \times R \times A$. In the inner loop, we apply Theorem 2 and only consider components ξ_f such that $\boldsymbol{\xi} \in \Xi_n$, and exclude scenarios which are already included in (RER) to get a subset $\hat{F} \subseteq F$. If $\hat{F}$ is not empty, we call the Fibonacci search function in line 10 to get the scenario ξ^0 which maximizes $f(\xi_f)$. We keep track of the greatest value of z and the maximizing scenario

Algorithm 3 FIBONACCI$\left(F, \boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \hat{\xi}^n\right)$

Input: set F in increasing order, $\boldsymbol{\xi}_{\boldsymbol{\ell}}$, ξ_r, ξ_a and sample $\hat{\xi}^n$

$\Omega \leftarrow [(\boldsymbol{\xi}_{\boldsymbol{\ell}}, \xi_r, \xi_a, \xi_f)]_{\xi_f \in F}$

Θ: array of Fibonacci numbers of size m in increasing order

// Corner cases

if $|\Omega| = 1$ **then**

  return Ω_0

else if $|\Omega| = 2$ **then**

  return $\boldsymbol{\xi}^* := \arg\max_{\boldsymbol{\xi}} \{z_n(\boldsymbol{\xi}) : \boldsymbol{\xi} \in \{\boldsymbol{\Omega_0}, \boldsymbol{\Omega_1}\}\}$

end if

// Initialize indices ℓ, u, i_a and i_b

$(\ell, i_a, u) \leftarrow (0, \Theta_{m-3}, \Theta_{m-1})$

$i_b \leftarrow \ell + u - i_a$

while $i_a \neq i_b$ **do**

  if $i_b > |\Omega| - 1$ **or** $z_n(\boldsymbol{\Omega}_{i_a}) > z_n(\boldsymbol{\Omega}_{i_b})$ **then**

    $u \leftarrow i_b$

    $i_b \leftarrow i_a$

    $i_a \leftarrow \ell + u - i_b$

  else

    $\ell \leftarrow i_a$

    $i_a \leftarrow i_b$

    $i_b \leftarrow \ell + u - i_a$

  end if

end while

if $\ell = 0$ **then**

  $\boldsymbol{\xi}^* \leftarrow \arg\max_{\boldsymbol{\xi}} \{z_n(\boldsymbol{\xi}) : \boldsymbol{\xi} \in \{\boldsymbol{\Omega}_{i_a}, \boldsymbol{\Omega}_{\boldsymbol{\ell}}\}\}$

else if $u = \Theta_{m-1}$ **then**

  $\boldsymbol{\xi}^* \leftarrow \arg\max_{\boldsymbol{\xi}} \{z_n(\boldsymbol{\xi}) : \boldsymbol{\xi} \in \{\boldsymbol{\Omega}_{i_a}, \boldsymbol{\Omega}_{\boldsymbol{u}}\}\}$

else

  $\boldsymbol{\xi}^* \leftarrow \boldsymbol{\Omega}_{i_a}$

end if

return $\boldsymbol{\xi}^*$

by z^* and $\boldsymbol{\xi}^*$, respectively. These are updated in lines 12-15. SEPARATION-FIB either returns the index associated with $\boldsymbol{\xi}^*$ if that scenario violates (c1), or returns an empty set otherwise.

Algorithm 4 SEPARATION-FIB$\left(\hat{\boldsymbol{\xi}}^n, \hat{\boldsymbol{x}}, \hat{\alpha}_n, \hat{\boldsymbol{\lambda}}\right)$

Input: sample $\hat{\xi}^n$ and optimal solutions of (RER) $\hat{x}$, $\hat{\alpha}_n$ and $\hat{\boldsymbol{\lambda}}$

$z^* \leftarrow -\infty$
for each $(\boldsymbol{\xi}_\ell, \xi_r, \xi_a) \in L \times R \times A$ **do**
  $\hat{F} \leftarrow \emptyset$
  for each $\xi_f \in F$ **do**
    $\boldsymbol{\xi} \leftarrow (\boldsymbol{\xi}_\ell, \xi_r, \xi_r, \xi_f)$
    if $\xi \in \Xi_n$ & $\xi \notin \hat{S}_n$ **then**
      $\hat{F} \leftarrow \hat{F} \cup \{\xi_f\}$
    end if
  end for
  if $\hat{F} \neq \emptyset$ **then** $\boldsymbol{\xi^0} \leftarrow$ FIBONACCI$\left(\hat{F}, \boldsymbol{\xi}_\ell, \xi_r, \xi_a, \hat{\xi}^n\right)$ ▷ Solve $\max_{\xi_f \in \hat{F}} f(\xi_f)$
  // Update maximizing ξ^* and optimal value z^*
  if $z_n(\boldsymbol{\xi^0}) > z*$ **then**
    $\boldsymbol{\xi}^* \leftarrow \boldsymbol{\xi^0}$
    $z^* \leftarrow z_n(\boldsymbol{\xi^0})$
  end if
end for
Let s^* be the index associated with $\boldsymbol{\xi}^*$
if $\alpha_n < z^*$ **then**
  return $\{s^*\}$
else
  return $\emptyset$
end if

D Computational Experiments

D.1 Case Study Data

Table 3: Per unit purchase cost, volume, and transportation cost per unit distance of water, food and medical kits.

Commodity	Unit Purchase Cost ($)	Volume (ft^3)	Transportation Cost ($)
Water (1000 galons)	647.7	144.6	0.3
Food (1000 kits)	5420	83.33	0.04
Medicine Kit (serves 4)	140	1.16	5.8e-4

D.2 Case Study Network

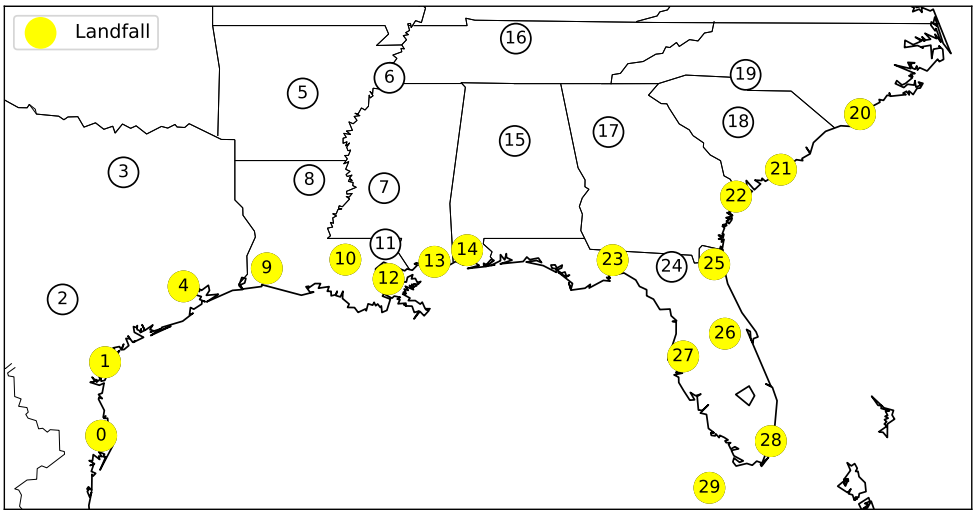


Figure 7: Map of the network with potential landfall nodes colored in yellow.

D.3 Synthetic Instances: Number of Scenarios

Table 4: Average number of scenarios across randomly generated networks for fixed size of the set F.

Size of F	Avg. Number of Scenarios
15	3070.0
30	6468.0
50	10020.0
75	14715.0

D.4 Synthetic Instances: Network Example

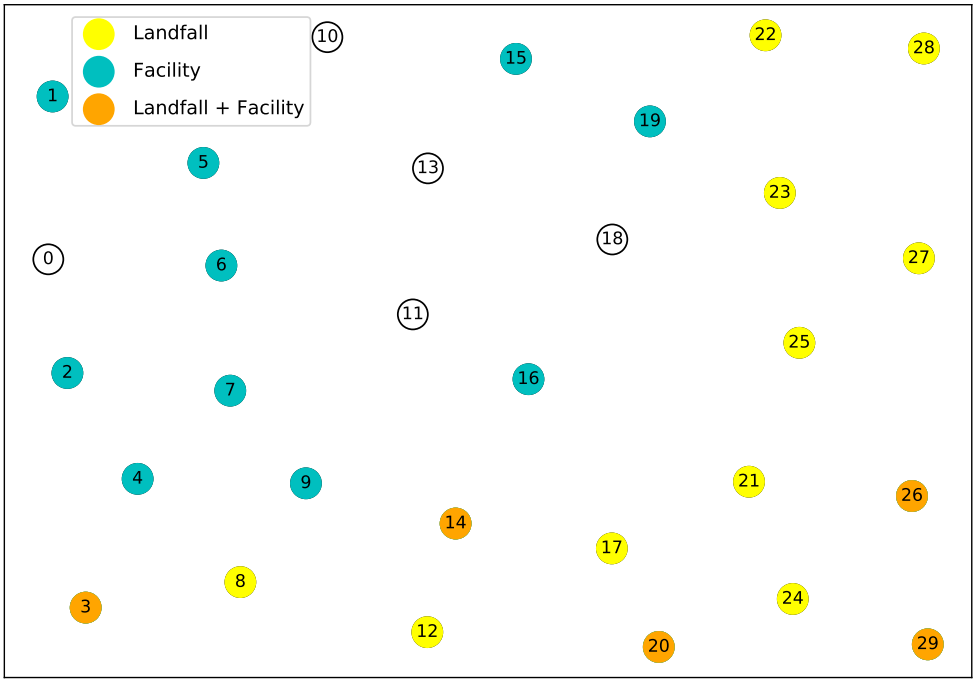
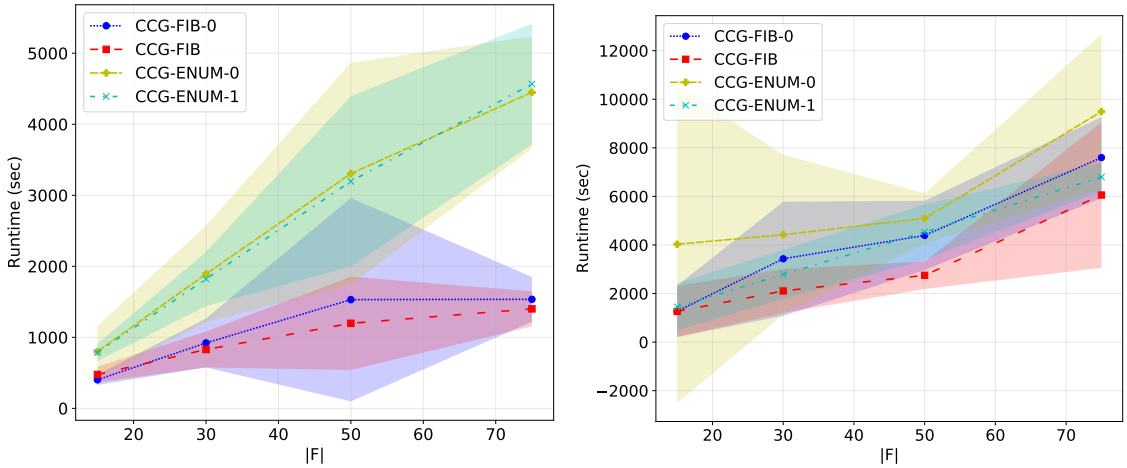


Figure 8: Landfall nodes and facilities of a randomly generated network of 30 nodes, where yellow nodes are potential landfall nodes, blue nodes are potential facility nodes, and orange nodes are both

D.5 Synthetic Instances: Runtimes

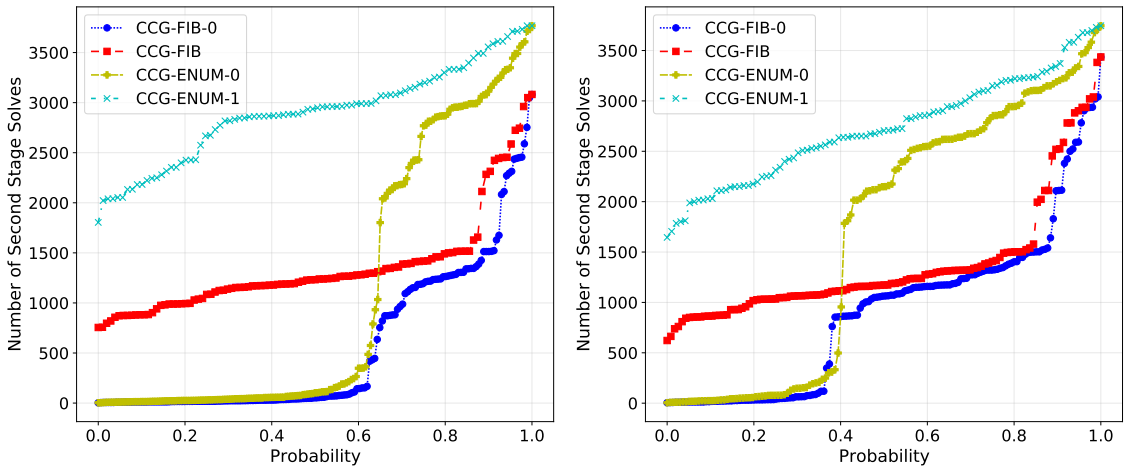


(a) $N = 10, \theta = 0.001$

(b) $N = 50, \theta = 0.001$

Figure 9: Average runtimes over 10 runs of CCG-FIB, CCG-FIB-0, CCG-ENUM-0, and CCG-ENUM-1 with Wasserstein radius $\theta = 0.001$ and sample set sizes N, with ribbons spanning two standard deviations.

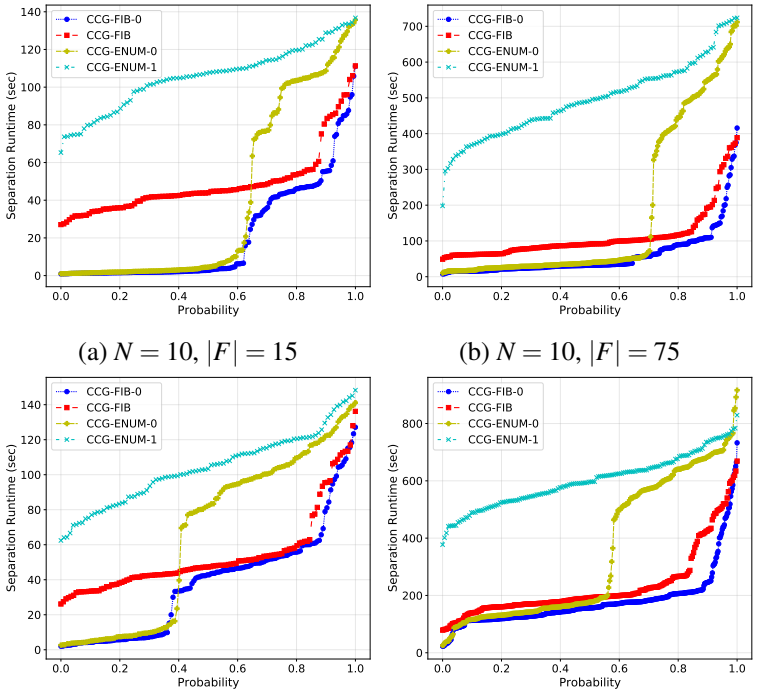
D.6 Synthetic Instances: Number of Second Stage Solves



(a) $N = 10$, $|F| = 15$ (b) $N = 50$, $|F| = 15$

Figure 10: Cumulative distribution of number of second stage solves during the separation step where $\theta = 0.01$ and $|F| = 15$, for sample set sizes 10 and 50.

D.7 Synthetic Instances: Separation Step Runtimes



(a) $N = 10$, $|F| = 15$ (b) $N = 10$, $|F| = 75$

(c) $N = 50$, $|F| = 15$ (d) $N = 50$, $|F| = 75$

Figure 11: Cumulative distribution of runtimes in the separation step of each method where $\theta = 0.01$, for sample set sizes 10 and 50, and for two discretizations of F, where $|F| = 15$ and $|F| = 75$.