

Learning to optimize with hidden constraints

Aaron Babier¹, Timothy C. Y. Chan¹, Adam Diamant², Rafid Mahmood¹

¹Mechanical & Industrial Engineering, University of Toronto, Toronto, Canada, {ababier, tcychan, rmahmood}@mie.utoronto.ca

²Schulich School of Business, York University, Toronto, Canada, adiamant@schulich.yorku.ca

The topic of learning to solve optimization problems has received interest from both the operations research and machine learning communities where each considers contrasting approaches: conditional stochastic optimization frameworks solved using provably optimal structured models versus deep learning models that leverage large data sets to yield empirically effective decision estimators. In this work, we combine the best of both worlds to solve the problem of learning to generate decisions to instances of continuous optimization problems where the feasible set varies with contextual features. We propose a novel framework for training a generative model to estimate optimal decisions by combining interior point methods and adversarial learning which we further embed within an active learning algorithm. Decisions generated by our model satisfy in-sample and out-of-sample optimality guarantees. Finally, we investigate case studies in portfolio optimization and personalized treatment design, demonstrating that our approach yields significant advantages over predict-then-optimize and supervised deep learning techniques, respectively.

1. Introduction

Advances in machine learning have led to growing interest in solving contextual constrained optimization problems, i.e., models whose objective or constraints depend on contextual features that vary from instance to instance. The goal is to automate complex decision-making processes in dynamic environments that require both speed and superior solution quality. However, the exact nature of the dependency between contexts and high-quality solutions is generally unknown a priori and thus, must be estimated using historical data. Furthermore, many applications may require solving problems with non-convex contextual constraints. Consider the following examples.

Portfolio optimization. Automated online brokers are increasingly used to recommend investment portfolios. Users have individual risk tolerances and restrictions on which investments to hold, resulting in the formulation of non-convex contextual constraints (Bansal and Pachamanova 2019, Kim et al. 2019, Li 2020). Further, new investors may not be able to articulate their preferences which means that brokers must infer these hidden constraints from the users’ information (Swezey and Charron 2018). The high-stakes nature of investing necessitates the creation of portfolios that have a certifiable

quality (Ban et al. 2018), but the ideal automated broker should also be able to quickly generate candidate portfolios.

Personalized cancer treatment. Radiation therapy is a widely used cancer treatment modality where each patient’s dose should meet a set of pass-fail non-convex criteria (Delaney et al. 2005). In general, not all criteria can be satisfied simultaneously. Thus, oncologists use their clinical expertise to identify personalized (i.e., context-dependent) subsets of the criteria to use as clinical constraints for a given patient. The emerging paradigm of automated treatment planning employs machine learning models to generate personalized, clinically acceptable doses for new patients using a library of historical treatments (Babier et al. 2019, Kearney et al. 2018). Ideally, patient-specific plans should be generated quickly to ensure that treatment is not delayed and should be of highest clinical quality so as to guarantee that patients receive the prescribed dose for their tumor.

Techniques from both operations research and machine learning can be used to solve contextual optimization problems, but there is generally a trade-off between quickly producing solutions and guaranteeing that they are of certifiable quality. In operations research, the traditional paradigm is the ‘*predict-then-optimize*’ approach which maps the context to an input parameter in an optimization model (see Chan et al. 2012, Angalakudati et al. 2014, Ferreira et al. 2015 for examples and Mišić and Perakis 2020 for a modern survey). These optimization problems, especially if non-convex, may be challenging to solve quickly. However, it is possible to generate decisions that are provably optimal, especially if the prediction model that transforms the context is appropriately trained (Elmachtoub and Grigas 2017, Bertsimas and Kallus 2019). In contrast, machine learning approaches, especially deep neural networks, directly predict optimal solutions to contextual optimization problems (see Bengio et al. 2020 and Mazyavkina et al. 2020 for recent surveys). These frameworks rely on large data sets of optimal decisions and do not provide theoretical optimality guarantees (and sometimes cannot be guaranteed to even produce feasible solutions). However, empirical evidence has shown that good predictions can be obtained in a fraction of the time it takes to solve a mathematical program (Larsen et al. 2018, Kool et al. 2019).

In this paper, we develop a methodology for learning to solve constrained optimization problems with hidden, potentially non-convex contextual constraints that combines the best of operations research (optimality guarantees) and machine learning (speed). We develop a prediction model that quickly produces decisions for constrained optimization

problems when given an input context. Trained using principled techniques derived from adversarial learning and convex optimization, our model yields optimality guarantees for generated decisions.

1.1. Main results and contributions

We first develop a theoretical foundation for our approach, which is an interior point method (IPM) for solving optimization problems where only partial information regarding the feasible set is known. We extend existing theory to consider a barrier function over a relaxation of the feasible set and derive a new ϵ -optimality guarantee. We then consider the problem of learning to quickly produce high-quality decisions with hidden, context-dependent, feasible sets when given only the context information as input. Our approach employs two machine learning models that are adversarially trained using data on past decisions and contexts. The first model is a binary classifier that takes a context and a decision as input and predicts whether a decision is feasible for that context. The second is a generative model that directly predicts solutions to contextual optimization problems given a context vector; once this model is trained, solutions can be generated quickly. The generator is trained using a similar process as in IPMs, i.e., the barrier parameter is annealed towards zero from a sufficiently large value with the trained classifier acting as the barrier function. Since learning to predict decisions typically requires large data sets for training, we embed our prediction models in an iterative active learning framework where an oracle of feasibility is repeatedly called to label new data and augment the existing sets. In applications where additional structure on the hidden constraints are provided, we can create computational, rule-based oracles that generate data to improve our models.

Like all prediction models, we evaluate the performance of our algorithm on out-of-sample problem instances. In particular, we first develop a theoretical generalization bound on the out-of-sample performance for any generative model that estimates optimal decisions, which complements our contribution to IPM theory. We then provide empirical evidence of our algorithm’s performance through comprehensive numerical results on the two example applications described earlier. We demonstrate that our approach has advantages over both conventional operations research and machine learning-based techniques.

Figure 1 summarizes the flow of this paper. Our specific contributions are as follows:

1. We introduce the concept of a δ -barrier; a barrier function for a relaxation of a hidden feasible set. We prove a new (δ, ϵ) -optimality guarantee for optimization problems

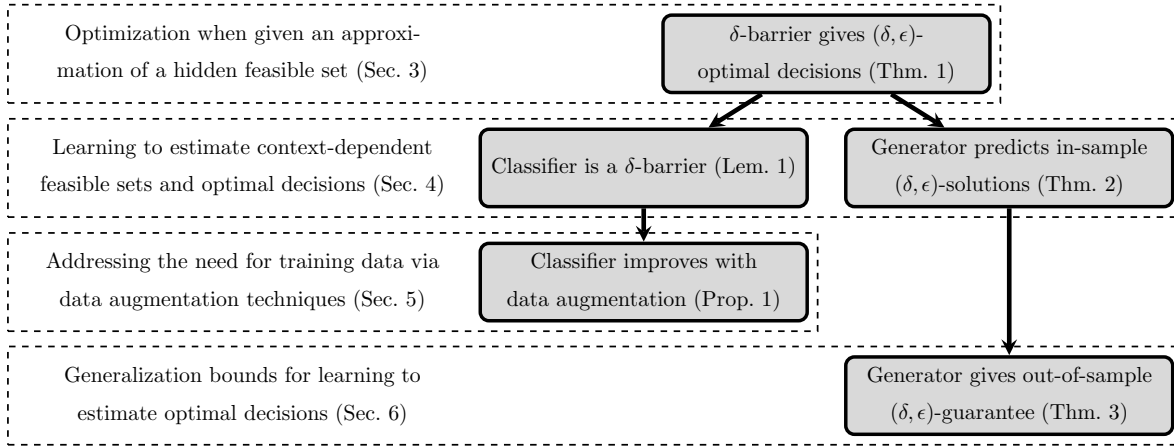


Figure 1 A roadmap of the main topics and contributions of this paper. The arrows denote conceptual dependencies between results.

where the feasible set can only be partially characterized through this relaxation and generalize key properties of IPM algorithms to this setting.

2. We introduce two learning problems for estimating a hidden feasible set and predicting a (δ, ϵ) -optimal solution when given a context vector. Training these models by combining IPM theory (operations research) and adversarial learning (machine learning) ensures the optimal values for in-sample instances to be bounded by our predicted objective function values.
3. We present an oracle-guided active learning algorithm—Interior Point Methods with Adversarial Networks (IPMAN)—that progressively predicts tighter (δ, ϵ) -optimal solutions by iteratively growing the training data set.
4. We prove a generalization bound on the out-of-sample (δ, ϵ) -optimality gap for any model that predicts solutions to a constrained optimization problem which includes IPMAN. Thus, we can characterize the ability of our model to predict optimal solutions for both in-sample and out-of-sample instances.
5. We validate our framework through two case study applications and compare against conventional operations research and deep learning techniques, respectively.
 - In a personalized portfolio optimization application where the conventional baseline would be to predict (investor preferences) then optimize (the portfolio), IPMAN directly predicts near-optimal decisions in an order-of-magnitude less time than it would take a commercial optimization solver. Furthermore, the predict-then-optimize baseline is extremely sensitive to parameter estimation error whereas our approach is shown to be robust.

- In a personalized radiation therapy application where the state-of-the-art uses supervised deep learning models to estimate dose decisions from CT images, IPMAN learns to generate decisions that better satisfy clinical constraints. In addition, IPMAN learns to generate decisions that satisfy one set of clinical metrics using data corresponding to another. This implies that our tool can be applied to new domains that differ in use-cases and/or policy.

1.2. Relationship to prior work

Our use of two predictive models, one to evaluate and one to generate decisions, is common in reinforcement learning (e.g., actor-critic methods, Konda and Tsitsiklis 2000) and deep learning (e.g., generative adversarial networks, Goodfellow et al. 2014). Our training function is derived using IPMs (Nesterov and Nemirovskii 1994) and our learning guarantees extend previous Rademacher complexity results for data-driven optimization (Bertsimas and Kallus 2019). Our oracle-guided data augmentation procedure is common in active learning (Settles 2009). Finally, our designing of a model to generate decisions is related to estimation of distribution algorithms (EDAs), commonly used in black-box optimization (Pelikan et al. 2002). We detail the most relevant work below.

Interior Point Methods. Interior point methods are among the most popular techniques for solving constrained optimization problems (Nesterov and Nemirovskii 1994). IPMs have been primarily applied to linear and quadratic optimization problems where they quickly converge to optimal solutions (Gondzio 2012). Recent results on barriers for arbitrary convex sets have renewed interest in IPMs for general convex optimization (Bubeck and Eldan 2019, Badenbroek and de Klerk 2018). IPMs also exist for non-convex problems (Vanderbei and Shanno 1999, Benson et al. 2004, Hinder and Ye 2018). These prior papers all assume access to explicit constraints or a barrier that is well-defined for the entire feasible set. Here, we construct a barrier function (i.e., our classifier) that approximates a relaxation of the hidden feasible set and develop an IPM theory for this setting.

Contextual optimization. The most common approach to contextual optimization is the ‘predict-then-optimize’ paradigm, i.e., to construct a parametric optimization model of the decision-making problem and use machine learning to predict parameters from context-dependent inputs (Angalakudati et al. 2014, Ferreira et al. 2015, Elmachoub and Grigas

2017). A non-parametric alternative is to directly estimate the effect of the context in terms of a conditional stochastic optimization model. For example, a stochastic objective that is conditioned on the context vector may be characterized by embedding a machine learning model to estimate probability weights on the objective (Kao et al. 2009, Hannah et al. 2010, Ban and Rudin 2018, Bertsimas and Kallus 2019, Bertsimas and McCord 2018).

Our work is most similar to Ban and Rudin (2018) who use Empirical Risk Minimization (ERM) to construct a predictor for the optimal solution to a Newsvendor problem. The Newsvendor problem is only constrained by the non-negativity of the variables, and thus, we generalize their result to an arbitrary set of restrictions by incorporating constraint satisfaction via a binary classifier. Bertsimas and Kallus (2019) remark on the challenges of constraint satisfaction when using ERM, and instead, propose a weighted learning framework that estimates the weights (i.e., conditional probability terms) in a sample-average optimization problem. They also prove several generalization results that arise from ERM theory. We extend their generalization bounds to out-of-sample ϵ -optimality guarantees, and in particular, to problems where the feasible set is not fully specified.

Deep learning for constrained optimization. Recent advances in deep learning have prompted a growing interest in using neural networks to solve constrained optimization problems (Bengio et al. 2020). These task-specific models are trained via customized learning algorithms. Different approaches include supervised learning with a data set of contexts and optimal solutions (Vinyals et al. 2015, Larsen et al. 2018), reinforcement learning (Bello et al. 2017, Kool et al. 2019), and task-based learning (Donti et al. 2017).

Predicting solutions is faster than using an optimization approach since a prediction model outputs solutions via a single function call. However, a trained model may not guarantee that predicted solutions can satisfy every constraint. There are, however, several approaches that address this issue. For instance, training a supervised learning technique on a large data set may help make predictions more accurate (Larsen et al. 2018). Recently, a class of neural network layers can be used to directly learn the implicit function of the KKT equations of a convex problem when the structure of the constraints are known (Amos and Kolter 2017, Agrawal et al. 2019). Finally, the loss function may be customized to encourage constraint satisfaction (Donti et al. 2017). We train a prediction model using a loss function motivated by IPM theory. This encourages the model to produce feasible solutions and allows us to prove optimality guarantees on the generated solutions. In

addition, in contrast to previous work, we relax the assumption that the training data consists solely of optimal solutions to analyze problem instances with different contexts.

Active learning. Many operational applications are characterized by data that is scarce or expensive to obtain (Gupta and Rusmevichientong 2020). In these settings, active learning and data augmentation are common approaches that can be used to train prediction models when there are limited amounts of labelled data (Settles 2009). In active learning, a prediction model is iteratively trained by first using the labelled data set and then calling an oracle to select and label a new subset of data from a prior pool of unlabelled points (Brinker 2003, Joshi et al. 2009). The goal is to enhance training by creating new labels for points that were originally part of the unlabelled data set. The oracle in active learning may be a heuristic model (e.g., Chen et al. 2020) or a human in-the-loop (e.g., Castrejon et al. 2017) and is typically expected to label thousands of unlabelled data points between iterations (Sener and Savarese 2018). While active learning is increasingly popular for classification, we are the first to apply it for prescriptive analytics by treating the problem of estimating whether a decision is feasible as a classification task.

2. Problem definition

In this paper, vectors are denoted in bold and sets in calligraphic. The interior, boundary, and closure of a set are $\text{int}(\mathcal{X})$, $\text{bd}(\mathcal{X})$, and $\text{cl}(\mathcal{X})$, respectively. The exclusion of $\mathcal{X}_1$ from $\mathcal{X}_2 \supseteq \mathcal{X}_1$ is denoted $\mathcal{X}_2 \setminus \mathcal{X}_1$. We denote probability distributions with $\mathbb{P}$ and their support with $\text{supp}(\mathbb{P})$. Finally, $\|\cdot\|$ denotes the l_2 -norm.

Let $\mathbf{x} \in \mathbb{R}^n$ denote a decision vector and $\mathbf{u} \in \mathcal{U}$ be a context vector. Consider the problem

$$\mathbf{OP}(\mathbf{u}): \min_{\mathbf{x}} \left\{ \mathbf{c}^\top \mathbf{x} \mid \mathbf{x} \in \mathcal{X}(\mathbf{u}), \mathbf{x} \in \mathcal{P} \right\},$$

where $\mathbf{c} \in \mathbb{R}^n$ is a cost vector (assumed without loss of generality that $\|\mathbf{c}\| = 1$), $\mathcal{P} := \{\mathbf{x} \mid \mathbf{a}_m^\top \mathbf{x} \leq b_m, m = 1, \dots, M\}$ is a known set of linear constraints, and $\mathcal{X}(\mathbf{u})$ is an *unknown* and potentially non-convex set of instance-specific constraints that must be learned. We assume in this paper that the context space $\mathcal{U}$ and feasible sets $\mathcal{X}(\mathbf{u})$ are well-behaved.

ASSUMPTION 1 (Compactness and feasibility).

1. For all $\mathbf{u} \in \mathcal{U}$, the feasible set $\mathcal{X}(\mathbf{u})$ is compact and has a non-empty interior. Further, the joint set $\{(\mathbf{x}, \mathbf{u}) \mid \mathbf{u} \in \mathcal{U}, \mathbf{x} \in \mathcal{X}(\mathbf{u})\}$ is compact and has a non-empty interior.

2. The known constraints $\mathcal{P} = \{\mathbf{x} \mid \mathbf{a}_m^\top \mathbf{x} \leq b_m, m = 1, \dots, M\}$ define a compact polyhedron whose interior bounds the hidden feasible set, i.e., $\mathcal{X}(\mathbf{u}) \subset \text{int}(\mathcal{P})$ for all $\mathbf{u} \in \mathcal{U}$.

The first statement ensures compactness and feasibility. The second states that the known constraints are relaxations of the hidden set. This is trivially satisfied when $\mathcal{P}$ and $\mathcal{X}(\mathbf{u})$ are compact since one can always re-define a larger $\mathcal{P}$ with no change to $\mathbf{OP}(\mathbf{u})$.

Let $\mathbf{x}^*(\mathbf{u})$ denote an optimal solution to $\mathbf{OP}(\mathbf{u})$. In Section 3, we first consider optimizing over a single instance of $\mathbf{OP}(\mathbf{u})$. We introduce a barrier function for a relaxation of the feasible set and show that an optimal solution to the corresponding unconstrained barrier problem satisfies an optimality bound that depends on the size of the relaxation.

In Section 4 and onward, we address our main problem of estimating a relaxation of the feasible set and a corresponding optimal solution for any $\mathbf{u}$. We use the theory developed in Section 3 to construct a generative model $F: \mathcal{U} \rightarrow \mathbb{R}^n$, i.e., a function that takes as input a context vector and outputs a decision. Decisions produced by this model satisfy an error bound

$$\left| \mathbf{c}^\top F(\mathbf{u}) - \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) \right| < \epsilon \quad (1)$$

for some ϵ . We assume access to a data set of $N_{\mathbf{x}}$ feasible and $\bar{N}_{\mathbf{x}}$ infeasible decisions as well as $N_{\mathbf{u}}$ context-dependent inputs. Due to the practical challenge of optimally solving $\mathbf{OP}(\mathbf{u})$, we do not assume that the given decisions are optimal. Specifically, we have:

- A data set of context vectors $\hat{\mathcal{U}} := \{\hat{\mathbf{u}}_i\}_{i=1}^{N_{\mathbf{u}}}$.
- A training data set of feasible decisions $\mathcal{D} := \{(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i)\}_{i=1}^{N_{\mathbf{x}}}$, where $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$ and $\hat{\mathbf{x}}_i \in \mathcal{X}(\hat{\mathbf{u}}_i)$ for all $i \in \{1, \dots, N_{\mathbf{x}}\}$. This data consists of decisions that were implemented in the past by domain experts or heuristics and may include multiple decisions for each $\hat{\mathbf{u}}_i$.
- A training data set of infeasible decisions $\bar{\mathcal{D}} := \{(\bar{\mathbf{x}}_i, \hat{\mathbf{u}}_i)\}_{i=1}^{\bar{N}_{\mathbf{x}}}$, where $\bar{\mathbf{x}}_i \in \mathbb{R}^n \setminus \mathcal{X}(\hat{\mathbf{u}}_i)$. This set may consist of proposed decisions that were not implemented or data generated by random sampling (e.g., Chan et al. 2020a).

Some applications may not possess high-quality data with both feasible and infeasible decisions. Thus in Section 5, we demonstrate how to train the models using an active learning framework to augment the data set with newly labelled points. In particular, we assume access to an oracle of feasibility $\Psi(\mathbf{x}, \mathbf{u})$, where $\Psi(\mathbf{x}, \mathbf{u}) = 1$ if $\mathbf{x} \in \mathcal{X}(\mathbf{u})$ and 0 otherwise. While oracles have previously included human annotators in many machine

learning problems (Castrejon et al. 2017), we instead construct data-driven, rule-based oracles defined over the training set for which we can make a large number of queries. Note that such oracles are not available for new out-of-sample instances.

3. Optimization with a hidden feasible set

In this section, we extend IPM theory to the case where the barrier function can only partially characterize the feasible region. In particular, consider the problem $\mathbf{OP}(\mathbf{u})$ for a single context vector $\mathbf{u}$. Although we do not know the hidden feasible set $\mathcal{X}(\mathbf{u})$, we do know the constraints associated with the relaxation $\mathcal{P}$. Thus, we first define the notion of a barrier function for partially specified feasible sets and then propose a barrier optimization problem that produces a solution $\mathbf{OP}(\mathbf{u})$ that satisfies an optimality guarantee.

If $\mathbf{OP}(\mathbf{u})$ had no hidden constraints then one could construct a canonical log-barrier, i.e., $\log B(\mathbf{x}) = \sum_{m=1}^M \log(b_m - \mathbf{a}_m^\top \mathbf{x})$, to solve $\mathbf{OP}(\mathbf{u})$ within an IPM (see Nesterov and Nemirovskii 1994). In our case, $\mathcal{X}(\mathbf{u}) \subset \mathcal{P}$ and thus, a canonical barrier using $\mathcal{P}$ may incorrectly return non-zero values for $\mathbf{x} \in \mathcal{P} \setminus \mathcal{X}(\mathbf{u})$. This canonical barrier can be viewed as a barrier function over a relaxation or superset of $\mathcal{X}(\mathbf{u})$. As a result, we define a new class of functions, δ -barriers, that are strictly positive for all $\mathbf{x} \in \mathcal{X}(\mathbf{u})$ (as in a barrier), and are zero for all $\mathbf{x}$ that are sufficiently far from $\mathcal{X}(\mathbf{u})$.

DEFINITION 1. For some $\delta > 0$, let $\mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) = \{\mathbf{x} + \boldsymbol{\epsilon} \mid \mathbf{x} \in \mathcal{X}(\mathbf{u}), \|\boldsymbol{\epsilon}\| < \delta\}$ be a neighborhood of $\mathcal{X}(\mathbf{u})$. A δ -barrier $B_\delta: \mathbb{R}^n \times \mathcal{U} \rightarrow [0, 1)$ is a continuous function that satisfies

$$\mathcal{X}(\mathbf{u}) \subset \{\mathbf{x} \mid B_\delta(\mathbf{x}, \mathbf{u}) > 0\} \subseteq \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})).$$

Note that any function $B(\mathbf{x})$ supported over a superset of $\mathcal{X}(\mathbf{u})$ is a δ -barrier for δ at least equal to the Hausdorff distance $d_H(\cdot, \cdot)$ between $\mathcal{X}(\mathbf{u})$ and the support of $B(\mathbf{x})$, i.e.,

$$\delta \geq d_H(\mathcal{X}(\mathbf{u}), \{\mathbf{x} \mid B(\mathbf{x}) > 0\}) = \min_{\xi \geq 0} \left\{ \xi \mid \{\mathbf{x} \mid B(\mathbf{x}) > 0\} \subseteq \mathcal{N}_\xi(\mathcal{X}(\mathbf{u})) \right\}. \quad (2)$$

Given the set of known constraints $\mathcal{P}$ and Assumption 1, a δ -barrier for $\mathcal{X}(\mathbf{u})$ can always be constructed simply by re-scaling the canonical barrier for $\mathcal{P}$. In particular, let $C^\mathcal{P} > \max_{m \in \{1, \dots, M\}, \mathbf{x} \in \mathcal{P}} \{b_m - \mathbf{a}_m^\top \mathbf{x}\}$ be a normalization factor and consider the function

$$B^\mathcal{P}(\mathbf{x}) := \prod_{m=1}^M \left[\frac{b_m - \mathbf{a}_m^\top \mathbf{x}}{C^\mathcal{P}} \right]^+. \quad (3)$$

This function is a δ -barrier for $\mathcal{X}(\mathbf{u})$ where $\delta = d_H(\mathcal{X}(\mathbf{u}), \mathcal{P})$.

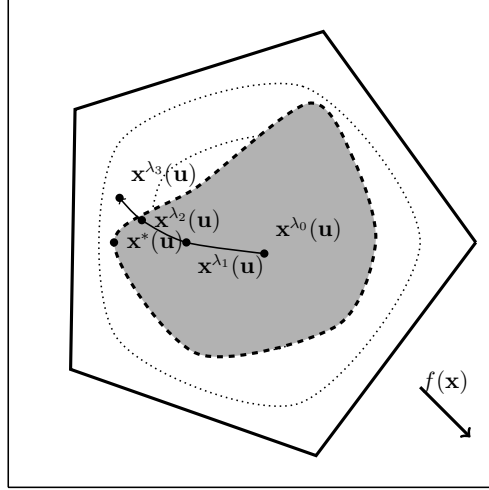


Figure 2 The bold shape is $\mathcal{P}$ and the filled region is $\mathcal{X}(\mathbf{u})$. The dotted lines represent level sets for $\log B^{\mathcal{P}}(\mathbf{x})$. An optimal solution to $\mathbf{OP}(\mathbf{u})$ is $\mathbf{x}^*(\mathbf{u})$.

Suppose we have a δ -barrier $B_\delta(\mathbf{x}, \mathbf{u})$ for some δ . Let $\lambda > 0$ be a constant corresponding to a Lagrangian dual variable and consider the unconstrained barrier optimization problem

$$\mathbf{BP}(\mathbf{u}, B_\delta, \lambda) : \min_{\mathbf{x}} \left\{ \mathbf{c}^\top \mathbf{x} - \lambda \log B_\delta(\mathbf{x}, \mathbf{u}) \right\}. \quad (4)$$

We now show that the optimal value of $\mathbf{OP}(\mathbf{u})$ is bounded by the optimal value of $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$. We note that the classical IPM assumptions of coerciveness and self-concordance are unnecessary; the former is replaced with continuity and a compact feasible set while the latter is only required for proofs of polynomial-time optimality guarantees.

THEOREM 1. *For any $\lambda > 0$, $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$ has an optimal solution $\mathbf{x}^\lambda(\mathbf{u})$. Furthermore, this solution is (δ, ϵ) -optimal for $\mathbf{OP}(\mathbf{u})$:*

$$\mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u}) - \epsilon < \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) < \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u}) + \delta, \quad (5)$$

where $\epsilon = C\lambda$ with C being a positive constant.

The (δ, ϵ) -optimality inequalities in Theorem 1 generalize the classical ϵ -optimality bound of IPMs (Nesterov and Nemirovskii 1994) to problems where the feasible set can only be partially characterized. That is, when $\delta = 0$, (5) reduces to the classical ϵ -optimality bound $\mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u}) - \epsilon < \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) < \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})$ which applies when the feasible set can be fully characterized. Further, similar to classical IPMs, the (δ, ϵ) -optimality of solutions to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$ can be controlled by tuning λ . Because $\epsilon = C\lambda$ for a fixed C , as λ goes to 0, so does ϵ . However, in contrast to classical theory on IPMs, δ is a property of the barrier function

and does not change with λ . Intuitively, we may expect that when λ is large (i.e., ϵ is also large), $\mathbf{x}^\lambda(\mathbf{u})$ may also have a large objective function value and be sub-optimal for $\mathbf{OP}(\mathbf{u})$. When λ is small (as is ϵ), $\mathbf{x}^\lambda(\mathbf{u})$ may have a small objective function value and be infeasible for $\mathbf{OP}(\mathbf{u})$. Although counter-examples may exist in the general case, we show in the Companion EC.2 that under a regularity assumption on the δ -barrier, we can guarantee $\mathbf{x}^\lambda(\mathbf{u}) \in \mathcal{X}(\mathbf{u})$ is feasible for a sufficiently large λ and that as λ decreases, $\mathbf{x}^\lambda(\mathbf{u})$ eventually becomes infeasible. Figure 2 shows a sample sequence of decreasing λ values and the corresponding solutions $\mathbf{x}^\lambda(\mathbf{u})$. Thus, an interior point algorithm using δ -barriers can be developed in much the same way as IPMs with canonical barrier functions.

4. Learning to optimize with hidden constraints

For a constrained optimization problem with a hidden feasible set, Section 3 demonstrates that we can solve a barrier optimization problem and obtain a (δ, ϵ) -optimal solution. However, the theory rests on two strong assumptions: (i) that we have access to a δ -barrier; and (ii) that we optimize $\mathbf{OP}(\mathbf{u})$ over a single context vector $\mathbf{u}$. In this section, we relax both assumptions. In particular, we assume that a δ -barrier must be constructed from data and learn to predict (δ, ϵ) -optimal solutions to $\mathbf{OP}(\mathbf{u})$ for any context vector $\mathbf{u}$ as input. To this end, we first construct a function that serves as a δ -barrier for any $\mathbf{u}$ by training a classifier using a data set of feasible ($\mathcal{D}$) and infeasible decisions ($\bar{\mathcal{D}}$). This classifier takes as input a decision $\hat{\mathbf{x}}$ and a context vector $\mathbf{u}$ and outputs 1 when it predicts a decision to be feasible and 0 otherwise. Then, to predict solutions to $\mathbf{OP}(\mathbf{u})$, we adversarially train a generative model, or generator, using the trained classifier as a δ -barrier.

Let $\mathcal{B} := \{B : \mathbb{R}^n \times \mathcal{U} \rightarrow [0, 1]\}$ denote the model class of classifiers and let $\mathcal{F} := \{F : \mathcal{U} \rightarrow \mathbb{R}^n\}$ denote the model class of generators. We assume both models are continuous in their parameters and in the decision vector $\mathbf{x}$. The two learning tasks are defined below:

1. **Learning a δ -barrier from classification:** Train $B \in \mathcal{B}$ to label $(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i) \in \mathcal{D}$ as feasible and $(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i) \in \bar{\mathcal{D}}$ as infeasible via a *Feasibility Classification Problem*:

$$\mathbf{FCP}(\mathcal{D}, \bar{\mathcal{D}}) : \max_{B \in \mathcal{B}} \left\{ \frac{1}{N_{\mathbf{x}}} \sum_{i=1}^{N_{\mathbf{x}}} \log B(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i) + \frac{1}{\bar{N}_{\mathbf{x}}} \sum_{\bar{i}=1}^{\bar{N}_{\mathbf{x}}} \log (1 - B(\bar{\mathbf{x}}_{\bar{i}}, \bar{\mathbf{u}}_{\bar{i}})) \right\}. \quad (6)$$

Let $B^*(\mathbf{x}, \mathbf{u})$ be a classifier trained by minimizing $\mathbf{FCP}(\mathcal{D}, \bar{\mathcal{D}})$.

2. **Learning to generate solutions to the barrier problem:** For $j \in \{1, \dots, J\}$ steps, let λ_j denote a decreasing dual parameter (i.e., $\lambda_{j+1} < \lambda_j$). We train the generator over context vectors $\hat{\mathcal{U}}$ via the *Generative Barrier Problem*:

$$\mathbf{GBP}(\hat{\mathcal{U}}, B^*, \lambda_j) : \min_{F \in \mathcal{F}} \left\{ \frac{1}{N_{\mathbf{u}}} \sum_{i=1}^{N_{\mathbf{u}}} \mathbf{c}^\top F(\hat{\mathbf{u}}_i) - \lambda_j \log B^*(F(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) - \lambda_j \log B^{\mathcal{P}}(F(\hat{\mathbf{u}}_i)) \right\}, \quad (7)$$

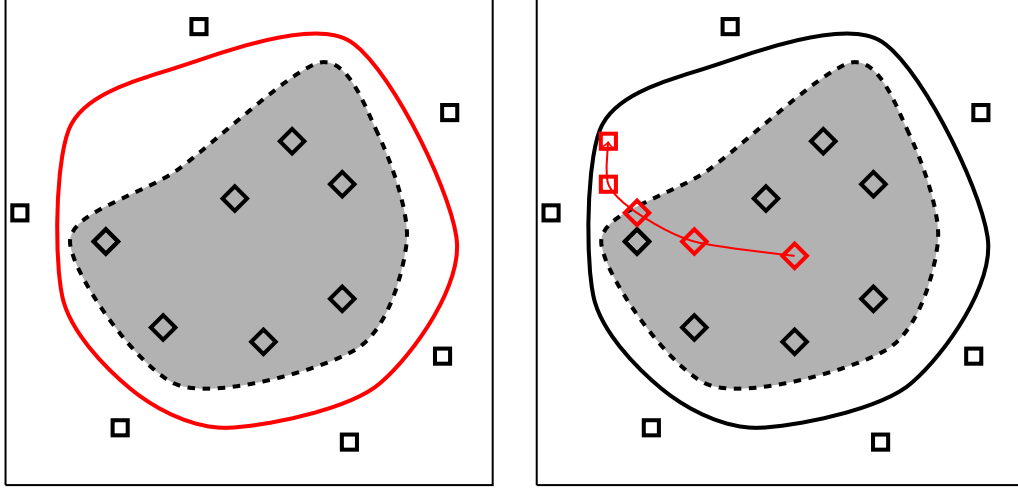
where $B^*(\mathbf{x}, \mathbf{u})$ is a trained classifier and $B^{\mathcal{P}}(\mathbf{x})$ is the canonical barrier from (3). For each j , let $F^{(j)}(\mathbf{u})$ be a generator trained by minimizing $\mathbf{GBP}(\hat{\mathcal{U}}, B^*, \lambda_j)$.

In Figure 3, we visualize the training procedure for a single context vector $\hat{\mathbf{u}}_i$. In the left-hand figure, a classification model uses points in $\mathcal{D}$ and $\bar{\mathcal{D}}$ to train a barrier function to distinguish between feasible and infeasible decisions. In the right-hand figure, the barrier function is used to train a sequence of generative models with decreasing λ_j . That is, we solve $\mathbf{GBP}(\hat{\mathcal{U}}, B^*, \lambda_j)$ for J steps to obtain a set of trained generators $\{F^{(1)}(\mathbf{u}), \dots, F^{(J)}(\mathbf{u})\}$; there is one generator for each value of λ_j . As in classical IPM theory, the sequence of generators produce solutions that are sub-optimal but feasible for large values of λ_j and infeasible for small values of λ_j (see EC.2 for conditions that guarantee strict convergence). After training has completed, we select a particular generator $F^{(j^*)}(\mathbf{u})$ by cross-validating the feasibility of decisions outputted by the model with a decision-maker or oracle of feasibility (see Section 5) for a hold-out set of context vectors. That is, we select the model $F^{(j^*)}(\mathbf{u})$ that most often predicts feasible decisions while still ensuring a suitably low objective function value. Thus, λ_j effectively acts as a tunable regularization parameter.

Given enough training data, the classifier will learn to become a sufficiently strong δ -barrier. Furthermore, the Generative Barrier Problem has a bounded support which ensures that the final generator trained by minimizing the empirical risk outputs in-sample predictions that satisfy a (δ, ϵ) -optimality bound with respect to $\mathbf{OP}(\mathbf{u})$. Thus, after these two learning tasks, we can use the trained generator to take a context vector $\mathbf{u}$ and predict a (δ, ϵ) -optimal decision for $\mathbf{OP}(\mathbf{u})$.

4.1. Learning a δ -barrier using classification

A classifier learns from decision data to output larger values when $\mathbf{x} \in \mathcal{X}(\mathbf{u})$ and smaller values when $\mathbf{x} \in \mathbb{R}^n \setminus \mathcal{X}(\mathbf{u})$. If the classifier can output positive values for all $\mathbf{x} \in \mathcal{X}(\mathbf{u})$, and zero for all $\mathbf{x}$ sufficiently far from $\mathcal{X}(\mathbf{u})$, then it is effectively a δ -barrier. By minimizing a Binary Cross Entropy (BCE) loss function, the Feasibility Classification Problem



(a) After B^* is trained, the support of the classifier (in red) approximates a δ -barrier. (b) The points in the line represent the set of predictions $F^{(j)}(\hat{\mathbf{u}}_i)$ for j .

Figure 3 The two learning problems for a single $\hat{\mathbf{u}}_i$ and corresponding $\text{OP}(\hat{\mathbf{u}}_i)$. $\diamond$ and $\square$ represent points in $\mathcal{D}$ and $\bar{\mathcal{D}}$, respectively. The filled region is $\mathcal{X}(\hat{\mathbf{u}}_i)$. The solid line shows the support of $B(\mathbf{x}, \hat{\mathbf{u}}_i)$.

in (6) attempts to train a classifier with this behavior. While nearly any learning algorithm and model class can be trained to approximate a δ -barrier, minimizing BCE is easy-to-implement with deep learning models because it is differentiable and can be further customized to different tasks (Goodfellow et al. 2016).

We now provide a *population level* argument for choosing to minimize BCE. Let $\mathbb{P}_{(\mathbf{x}, \mathbf{u})}$ be a distribution of feasible pairs, i.e., $(\mathbf{x}, \mathbf{u})$ where $\mathbf{x} \in \mathcal{X}(\mathbf{u})$, and let $\bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})}$ be a distribution of infeasible pairs, i.e., $(\mathbf{x}, \mathbf{u})$ where $\mathbf{x} \notin \mathcal{X}(\mathbf{u})$. These pairs can correspond to empirical data distributions or any other arbitrary probabilities. Then, consider the stochastic optimization variant of the Feasibility Classification Problem:

$$\mathbf{S}\text{-FCP}(\mathbb{P}_{(\mathbf{x}, \mathbf{u})}, \bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})}) : \max_{B \in \mathcal{B}} \left\{ \mathbb{E}_{\mathbf{x}, \mathbf{u} \sim \mathbb{P}_{(\mathbf{x}, \mathbf{u})}} [\log B(\mathbf{x}, \mathbf{u})] + \mathbb{E}_{\mathbf{x}, \mathbf{u} \sim \bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})}} [\log (1 - B(\mathbf{x}, \mathbf{u}))] \right\}.$$

LEMMA 1. Assume that $\mathbb{P}_{(\mathbf{x}, \mathbf{u})}$ and $\bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})}$ have closed and disjoint supports and that $\mathcal{B}$ includes all continuous functions mapping $\mathbb{R}^n \times \mathcal{U} \rightarrow [0, 1]$.

1. There exists a continuous function $B^*(\mathbf{x}, \mathbf{u})$ of $\mathbf{S}\text{-FCP}(\mathbb{P}_{(\mathbf{x}, \mathbf{u})}, \bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})})$ where $B^*(\mathbf{x}, \mathbf{u}) = 1$ for all $(\mathbf{x}, \mathbf{u}) \in \text{supp}(\mathbb{P}_{(\mathbf{x}, \mathbf{u})})$ and $B^*(\mathbf{x}, \mathbf{u}) = 0$ for all $(\mathbf{x}, \mathbf{u}) \in \text{supp}(\bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})})$ that achieves an optimal value of 0 for $\mathbf{S}\text{-FCP}(\mathbb{P}_{(\mathbf{x}, \mathbf{u})}, \bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})})$.
2. If $\text{supp}(\mathbb{P}_{(\mathbf{x}, \mathbf{u})}) = \{(\mathbf{x}, \mathbf{u}) \mid \mathbf{x} \in \mathcal{X}(\mathbf{u}), \mathbf{u} \in \mathcal{U}\}$, then for any $\mathbf{u} \in \mathcal{U}$, the product $B^*(\mathbf{x}, \mathbf{u})B^{\mathcal{P}}(\mathbf{x})$ is a δ -barrier with $\delta = d_H(\mathcal{X}(\mathbf{u}), \mathcal{P})$.

The first statement Lemma 1 is a variation of a result by Arjovsky and Bottou (2017, Theorem 2.1), which demonstrates that an optimal continuous function binary classifier trained by minimizing BCE will equal to 1 and 0 over the distributions for each respective class. The second statement indicates that a product of classifiers is a δ -barrier when trained with a probability distribution over feasible decisions. Although the lemma is stated in terms of population-level probability distributions, the assumption of closed and disjoint supports is satisfied with empirical data distributions and thus, it holds without loss of generality. Further, Lemma 1 assumes that the model class $\mathcal{B}$ contains all continuous mappings of $\mathbb{R}^n \times \mathcal{U} \rightarrow [0, 1]$. This assumption is a sufficient rather than necessary condition. Since large neural networks can approximate any continuous function classifier (see Hornik 1991), we find that in practice, designing a neural network model of appropriate size and architecture produces a classifier of sufficient quality for our prediction tasks.

4.2. Learning to generate solutions to the barrier problem

We train a generator $F^{(j)}(\mathbf{u})$ to solve $\mathbf{GBP}(\hat{\mathcal{U}}, B^*, \lambda_j)$ (an empirical risk minimization of the barrier problem presented in Section 3) for a decreasing sequence of dual regularizers $\lambda_j > 0$. In this subsection, we show that training a generator via $\mathbf{GBP}(\hat{\mathcal{U}}, B^*, \lambda_j)$ ensures that for all in-sample context vectors $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$, the error between the solution generated by $F^{(j)}(\hat{\mathbf{u}}_i)$ and the optimal value of $\mathbf{OP}(\hat{\mathbf{u}}_i)$ is bounded. In particular, the generative model outputs decisions satisfying the optimality guarantee given by (1).

Recall that $\mathbf{GBP}(\hat{\mathcal{U}}, B^*, \lambda_j)$ has two barrier terms consisting of the classifier and the known constraints. Let $B^{*,\mathcal{P}}(\mathbf{x}, \mathbf{u}) := B^*(\mathbf{x}, \mathbf{u})B^{\mathcal{P}}(\mathbf{x})$ be the product of the two barriers as described in Lemma 1. We simplify (7) to formulate the single-barrier objective function

$$\min_{F \in \mathcal{F}} \left\{ \frac{1}{N_{\mathbf{u}}} \sum_{i=1}^{N_{\mathbf{u}}} \mathbf{c}^T F(\hat{\mathbf{u}}_i) - \lambda_j \log B^{*,\mathcal{P}}(F(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) \right\}.$$

From Lemma 1, given sufficient training data, the product barrier will be a δ -barrier. Furthermore, since $\mathcal{P}$ is compact under Assumption 1, $B^{*,\mathcal{P}}(\mathbf{x}, \mathbf{u})$ will always have bounded support. This property ensures that all solutions satisfy an in-sample optimality bound.

THEOREM 2. *For $\lambda_j > 0$, let $F^{(j)}(\mathbf{x}, \mathbf{u})$ and $B^{*,\mathcal{P}}(\mathbf{x}, \mathbf{u}) = B^*(\mathbf{x}, \mathbf{u})B^{\mathcal{P}}(\mathbf{x})$ be the trained generator and product barrier, respectively. For any $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$, let $\mathbf{x}^{\lambda_j}(\hat{\mathbf{u}}_i)$ be an optimal solution to $\mathbf{BP}(\hat{\mathbf{u}}_i, B^{*,\mathcal{P}}, \lambda_j)$. Then, there exists $\delta, \epsilon > 0$ such that*

$$|\mathbf{c}^T F^{(j)}(\hat{\mathbf{u}}_i) - \mathbf{c}^T \mathbf{x}^*(\hat{\mathbf{u}}_i)| < |\mathbf{c}^T F^{(j)}(\hat{\mathbf{u}}_i) - \mathbf{c}^T \mathbf{x}^{\lambda_j}(\hat{\mathbf{u}}_i)| + \max(\delta, \epsilon). \quad (8)$$

Theorem 2 ensures that for any predicted decision from an in-sample context vector $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$, the optimality gap is bounded above by the sum of two terms: (i) the empirical error between the optimal solution to $\mathbf{BP}(\hat{\mathbf{u}}_i, B^{*,\mathcal{P}}, \lambda_j)$ and $\mathbf{OP}(\hat{\mathbf{u}}_i)$ as given in Theorem 1; and (ii) a (δ, ϵ) -optimality bound. The value of this bound, and hence the quality of the trained generator $F^{(j)}(\mathbf{u})$, depends on the quality of the classifier. Recall that λ_j is effectively a regularization parameter for training $F^{(j)}(\mathbf{u})$. If the classifier is sufficiently trained so that it is a δ -barrier for $\hat{\mathbf{u}}_i$, then selecting an appropriate λ_j ensures that the trained generator produces a solution that is arbitrarily close to the optimal value for $\mathbf{OP}(\hat{\mathbf{u}}_i)$.

In the proof of Theorem 2, we demonstrate that ϵ can be calculated using known quantities. However, δ is a property of the classifier and the hidden feasible set $\mathcal{X}(\hat{\mathbf{u}}_i)$. Since we may not be able to characterize $\mathcal{X}(\hat{\mathbf{u}}_i)$ exactly, we do not know the exact value of δ . The next result demonstrates that we can bound this value as a function of the data.

COROLLARY 1. *For any $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$, δ in (8) is bounded by $\delta \leq d_H(\{\hat{\mathbf{x}} \mid (\hat{\mathbf{x}}, \hat{\mathbf{u}}_i) \in \mathcal{D}\}, \mathcal{P})$.*

Corollary 1 reveals the key challenge of learning to optimize over hidden feasible sets. While we may generate decisions that satisfy a (δ, ϵ) -optimality guarantee, the quality of the δ -barrier is highly dependent on the available training data. Indeed, in order to appropriately train a classifier to learn to become a δ -barrier, we require a significant amount of data that includes both feasible $\mathcal{D}$ and infeasible $\bar{\mathcal{D}}$ decisions. This setup is typical of many deep learning applications. For example, the FaceNet architecture (Schroff et al. 2015) used more than 400,000 samples to train a neural network to verify and recognize faces. Fortunately, $\mathcal{D}$ only has to contain feasible decisions (as opposed to optimal solutions) which may be easier to obtain. Nevertheless, since many operational applications may not have data sets of this magnitude, in the next section, we propose an active learning approach to increase the training data for our predictive models.

5. Improving learning to optimize with active learning

Active learning is an increasingly popular approach to classification tasks with insufficient labelled data but easily accessible unlabelled data. The general active learning framework involves multiple iterations where a predictive model is first trained with the available labelled data and then an oracle is used to label a new subset of the unlabelled data which is then incorporated into the labelled set. One of the main challenges in active learning is to accurately label a sufficient number of points. For example, classification tasks may

use an approximate oracle as a heuristic or crowd-source humans via a Mechanical Turk service. In our problem, the oracle must correctly identify whether a decision is feasible given the context vector. Thus, in this section, we first introduce an approach to construct feasibility oracles for in-sample contexts; such oracles can be designed by assuming certain structure on the constraints. We then introduce our active learning algorithm and prove that training the models from Section 4 in such an iterative framework improves learning.

5.1. Constructing data-driven oracles

We require an oracle that is a black-box labelling function $\Psi(\mathbf{x}, \mathbf{u})$, which yields $\Psi(\mathbf{x}, \mathbf{u}) = 1$ if $\mathbf{x} \in \mathcal{X}(\mathbf{u})$ and 0 otherwise. So far in this work, we have not assumed any specific structure on the constraints of $\mathcal{X}(\mathbf{u})$ but in a practical application, we may know the nature of the constraints, which we can then use to construct a data-driven labelling function. For example, the hidden set may consist of a linear constraint where we know the left-hand-side but do not know the right-hand-side value. Below, we present a simple example for creating such a data-driven oracle; we use similar constructions in our numerical experiments.

EXAMPLE 1 (DATA-DRIVEN ORACLE). *Suppose that the hidden feasible set consists of a single linear constraint $\mathcal{X}(\mathbf{u}) = \{\mathbf{x} \mid \mathbf{a}^\top \mathbf{x} \leq b(\mathbf{u})\}$ where the vector $\mathbf{a} \in \mathbb{R}^n$ is known and is independent of the context. The right-hand-side can take one of two potential values that are dependent on the context vector, i.e., $b(\mathbf{u}) \in \{b_1, b_2\}$ where $b_1 < b_2$. Since every $(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i) \in \mathcal{D}$ is a feasible pair, we can estimate the right-hand-side for any context vector in the training set by evaluating $\mathbf{a}^\top \hat{\mathbf{x}}_i$ and designing the corresponding oracle:*

$$\Psi(\mathbf{x}, \hat{\mathbf{u}}_i) = \begin{cases} 1 & \text{if } \mathbf{a}^\top \mathbf{x} \leq b_1 \text{ and } \max_{(\hat{\mathbf{x}}, \hat{\mathbf{u}}_i) \in \mathcal{D}} \mathbf{a}^\top \hat{\mathbf{x}} \leq b_1 \\ 1 & \text{if } \mathbf{a}^\top \mathbf{x} \leq b_2 \text{ and } b_1 \leq \max_{(\hat{\mathbf{x}}, \hat{\mathbf{u}}_i) \in \mathcal{D}} \mathbf{a}^\top \hat{\mathbf{x}} \leq b_2, \\ 0 & \text{otherwise} \end{cases} \quad \forall \hat{\mathbf{u}}_i \in \hat{\mathcal{U}}.$$

Data-driven oracles, such as the one presented in Example 1, offer significant advantages compared to a human-in-the-loop oracle. A computational oracle can be called a large number of times during training and does not need to be defined for all possible context vectors. One disadvantage of the oracle above is that it is an approximation; here, a tighter subset of $\mathcal{X}$ (e.g., if $\mathcal{D}$ contains feasible decisions that inadvertently satisfy constraints more conservatively than necessary). While we develop theory assuming a perfect oracle of feasibility, we demonstrate in our numerical experiments that our approach is robust to erroneous or noisy oracles that may misclassify unlabelled points.

Algorithm 1 Interior Point Methods with Adversarial Networks (IPMAN)

Input: Number of IPMAN iterations K and IPM steps J ; Dual regularizers $\{\lambda_j\}_{j=1}^J$; Initial data sets $\mathcal{D}^{(1)}$, $\bar{\mathcal{D}}^{(1)}$, and $\hat{\mathcal{U}}$.

Output: Final generative models $F^{(j,K)}$ for $j \in \{1, \dots, J\}$

- 1: **for** $k = 1$ **to** K **do**
- 2: Train classifier $B^{(k)}$ via $\mathbf{FCP}(\mathcal{D}^{(k)}, \bar{\mathcal{D}}^{(k)})$.
- 3: **for** $j = 1$ **to** J **do**
- 4: Train generator $F^{(j,k)}$ via $\mathbf{GBP}(\hat{\mathcal{U}}, B^{(k)}, \lambda_j)$.
- 5: Update $\mathcal{D}^{(k+1)} = \mathcal{D}^{(k)} \cup \mathcal{Q}$, where $\mathcal{Q} := \left\{ (F^{(j,k)}(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) \mid \Psi(F^{(j,k)}(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) = 1, \hat{\mathbf{u}}_i \in \hat{\mathcal{U}} \right\}$.
- 6: Update $\bar{\mathcal{D}}^{(k+1)} = \bar{\mathcal{D}}^{(k)} \cup \bar{\mathcal{Q}}$, where $\bar{\mathcal{Q}} := \left\{ (F^{(j,k)}(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) \mid \Psi(F^{(j,k)}(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) = 0, \hat{\mathbf{u}}_i \in \hat{\mathcal{U}} \right\}$.
- 7: **end for**
- 8: **end for**
- 9: **return** $F^{(j,K)}$ for $j \in \{1, \dots, J\}$

5.2. Interior Point Methods with Adversarial Networks

We now introduce an oracle-guided algorithm, Interior Point Methods with Adversarial Networks (IPMAN), for learning to solve constrained optimization problems with hidden, and potentially non-convex, contextual constraints. In each iteration of the algorithm, we train a classifier and a generator in succession as described in Section 4. Then, for each context $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$, we have the trained generator predict decisions. We use the oracle to label each predicted decision as feasible or infeasible and then augment the existing data sets of feasible and infeasible decisions for the next iteration. Let $k \in \{1, \dots, K\}$ index an iteration of training and superscript (k) index the model and data sets after the conclusion of the k -th iteration. Algorithm 1 summarizes the approach.

In each iteration, the classifier $B^{(k+1)}(\mathbf{x}, \mathbf{u})$ is trained with data sets $\mathcal{D}^{(k+1)}$ and $\bar{\mathcal{D}}^{(k+1)}$, respectively, which are larger than the data sets $\mathcal{D}^{(k)}$ and $\bar{\mathcal{D}}^{(k)}$ due to the labelling procedure. Due to this data augmentation step, the optimal solution set of the Feasibility Classification Problem can be shown to contract after each iteration, i.e., $B^{(k+1)}(\mathbf{x}, \mathbf{u})$ represents a tighter approximation to $\mathcal{X}(\mathbf{u})$ than $B^{(k)}(\mathbf{x}, \mathbf{u})$, which we now show formally.

PROPOSITION 1. *Assume that $\mathcal{B}$ contains all continuous functions mapping $\mathcal{U} \times \mathbb{R}^n \rightarrow [0, 1]$. For any k , let $\mathcal{B}^{(k)}$ be the optimal solution set of $\mathbf{FCP}(\mathcal{D}^{(k)}, \bar{\mathcal{D}}^{(k)})$. Then, $\mathcal{B}^{(k+1)} \subset \mathcal{B}^{(k)}$.*

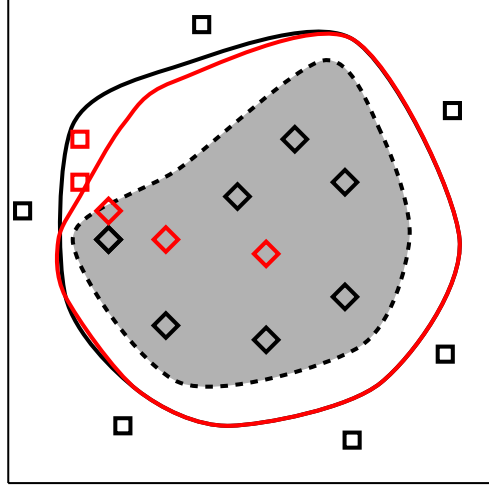


Figure 4 The trained classifier after an iteration of data augmentation. $\diamond$ and $\square$ represent points in $\mathcal{D}^{(k+1)}$ and $\bar{\mathcal{D}}^{(k+1)}$, respectively, with the colored (red) points denoting the augmented points. The filled region is $\mathcal{X}(\mathbf{u})$. The gray line shows the support of $B^{(k)}(\mathbf{x}, \mathbf{u})$ and the red line shows the support of $B^{(k+1)}(\mathbf{x}, \mathbf{u})$.

Proposition 1 demonstrates that by iteratively increasing the amount of available data using the oracle, we can contract the optimal solution set. Because the sets are augmented with correctly labelled data (assuming a perfect oracle), data augmentation removes classifiers that are a looser approximation to the feasible set. Intuitively, it allows the classifier to correct regions it had previously mislabelled as feasible and reinforce regions it had correctly labelled so that it does not incorrectly mislabel them in future iterations. Figure 4 shows an example of this behavior; the red classifier is a tighter approximation of $\mathcal{X}(\mathbf{u})$ than the gray classifier due to the additional points.

Recall from Theorem 2 that a trained generator satisfies a (δ, ϵ) -optimality bound where δ is a property of the classifier. Corollary 1 demonstrates that while δ cannot be directly calculated, we can compute its upper bound which is dependent on the data set of feasible decisions $\mathcal{D}$. In IPMAN, by increasing the size of the data set after each iteration, we ensure that this upper bound is non-increasing and the resulting optimality gap decreases.

6. Generalization of (δ, ϵ) -optimality to unseen instances

The IPMAN framework iteratively trains a generative model to predict solutions to a barrier optimization problem when given a context vector. In this section, we analyze the potential of our approach, or any machine learning model used to generate solutions to contextual constrained optimization problems, to predict (δ, ϵ) -optimal solutions when given an out-of-sample context vector $\mathbf{u}$. To this end, assume there exists a probability

distribution over context vectors $\mathbb{P}_{\mathbf{u}}$ and that the data set of context vectors $\hat{\mathcal{U}}$ is sampled i.i.d. according to $\mathbb{P}_{\mathbf{u}}$. We then use Rademacher complexity theory to obtain a probabilistic bound on the empirical error from an out-of-sample input (Bartlett and Mendelson 2002).

DEFINITION 2 (BERTSIMAS AND KALLUS (2019)). *Let $\mathcal{F} \subset \{F(\mathbf{u}) : \mathcal{U} \rightarrow \mathbb{R}^n\}$ be a function class and $\hat{\mathcal{U}} \sim \mathbb{P}_{\mathbf{u}}$ be an i.i.d. data set. The empirical multivariate Rademacher complexity of $\mathcal{F}$ is*

$$\hat{\mathfrak{R}}_{N_{\mathbf{u}}}(\mathcal{F}, \hat{\mathcal{U}}) = \mathbb{E}_{\boldsymbol{\sigma} \sim p_{\boldsymbol{\sigma}}} \left[\frac{2}{N_{\mathbf{u}}} \sup_{F \in \mathcal{F}} \sum_{i=1}^{N_{\mathbf{u}}} \boldsymbol{\sigma}_i^{\top} F(\hat{\mathbf{u}}_i) \mid \hat{\mathcal{U}} = \{\hat{\mathbf{u}}_i\}_{i=1}^{N_{\mathbf{u}}} \right],$$

where $\boldsymbol{\sigma}_i \sim p_{\boldsymbol{\sigma}}$ is an n -dimensional vector of i.i.d. Rademacher variables. The multivariate Rademacher complexity of $\mathcal{F}$ is $\mathfrak{R}_{N_{\mathbf{u}}}(\mathcal{F}) = \mathbb{E}_{\hat{\mathcal{U}} \sim \mathbb{P}_{\mathbf{u}}} [\hat{\mathfrak{R}}_{N_{\mathbf{u}}}(\mathcal{F}, \hat{\mathcal{U}})]$.

In statistical learning theory, Rademacher complexities are used to generate risk bounds that are dependent on the class of learning model that is used ($\mathcal{F}$) and the training data (Bartlett and Mendelson 2002). Bertsimas and Kallus (2019) use the theory to develop generalization bounds for predicting decisions to problems with a conditional stochastic optimization objective; we extend their work by providing a probabilistic bound on (δ, ϵ) -optimality when the feasible set is not fully specified. Typically, models that can learn complex relationships will require a more complex model class. The IPMAN algorithm is agnostic to the class of model used. However, if a specific generalization bound is required, a model class $\mathcal{F}$ with a tightly bounded Rademacher complexity should be selected.

REMARK 1. *Although the literature mostly focuses on the single-variate Rademacher complexity, Bertsimas and Kallus (2019) and Maurer (2016) prove bounds for linear multivariate classes (e.g., $\mathcal{F}_R = \{\mathbf{W}\mathbf{u} \mid \|\mathbf{W}\| \leq R\}$). In general, if $F(\mathbf{u}) = (F_1(\mathbf{u}), \dots, F_n(\mathbf{u}))$, then $\mathcal{F} \subset \times_{\ell=1}^n \mathcal{F}_{\ell}$, where $\mathcal{F}_{\ell} = \{F(\mathbf{u})^{\top} \mathbf{e}_{\ell} \mid F \in \mathcal{F}\}$ and $\mathbf{e}_{\ell}$ is the ℓ -th identity vector. Then, $\hat{\mathfrak{R}}_{N_{\mathbf{u}}}(\mathcal{F}, \hat{\mathcal{U}}) \leq \sum_{\ell=1}^n \hat{\mathfrak{R}}_{N_{\mathbf{u}}}(\mathcal{F}_{\ell}, \hat{\mathcal{U}})$ decomposes to a sum of single-variate complexities. We refer to Bartlett and Mendelson (2002) for bounds on linear and tree models and Bartlett and Mendelson (2002), Neyshabur et al. (2015) and Foster et al. (2018) for neural networks.*

Instead of using the product barrier composed of the classifier and the barrier for the known constraints, we assume that the generator is trained using a δ -barrier $B_{\delta}(\mathbf{x}, \mathbf{u})$ for some $\delta > 0$, i.e., by minimizing $\mathbf{GBP}(\hat{\mathcal{U}}, B_{\delta}, \lambda)$. Note that since we are using a fixed data

set $\hat{\mathcal{U}}$, there is no guarantee that $F(\mathbf{u})$ will be feasible or even satisfy $B_\delta(F(\mathbf{u}), \mathbf{u}) > 0$ for an arbitrary $\mathbf{u}$. Because $\mathcal{P}$ is available, however, we can always project any generated solution to the polyhedron to ensure that the generator predicts decisions that are bounded.

ASSUMPTION 2. Consider a δ -barrier $B_\delta(\mathbf{x}, \mathbf{u})$ and generator $F(\mathbf{u})$ trained via $\mathbf{GBP}(\hat{\mathcal{U}}, B_\delta, \lambda)$. We use projections $F^*(\mathbf{u}) = \arg \min_{\mathbf{x}} \{\|\mathbf{x} - F(\mathbf{u})\| \mid \mathbf{x} \in \mathcal{P}\}$ at test time.

Our generalization bound follows from Bertsimas and Kallus (2019). While they derive an empirical risk bound on an unconstrained stochastic optimization problem, we focus on (δ, ϵ) -optimality for a constrained continuous optimization problem (see EC.3 for proof).

THEOREM 3. Let F^* satisfy Assumption 2, K and L_∞ be sufficiently large positive constants, and fix $\beta \in (0, 1)$. Then, for any $\gamma > 0$, the following inequality holds

$$\begin{aligned} & \mathbb{P}_{\mathbf{u}} \left\{ \mathbf{c}^\top F^*(\mathbf{u}) - \epsilon - \gamma < \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) < \mathbf{c}^\top F^*(\mathbf{u}) + \delta + \gamma \right\} \\ & \geq 1 - \frac{\frac{1}{N_{\mathbf{u}}} \sum_{i=1}^{N_{\mathbf{u}}} |\mathbf{c}^\top F^*(\hat{\mathbf{u}}_i) - \mathbf{c}^\top \mathbf{x}^\lambda(\hat{\mathbf{u}}_i)| + K \sqrt{\frac{\log(1/\beta)}{2N_{\mathbf{u}}}} + \sqrt{2n} L_\infty \mathfrak{R}_{N_{\mathbf{u}}}(\mathcal{F})}{\gamma}, \end{aligned}$$

with probability at least $1 - \beta$ with respect to the sampling of $\hat{\mathcal{U}} \sim \mathbb{P}_{\mathbf{u}}$.

Theorem 3 bounds the (δ, ϵ) -optimality of a random context vector. It specifies, given a γ and F^* , the probability that the model will predict a $(\delta + \gamma/L, \epsilon + \gamma)$ -optimal solution for an out-of-sample context vector $\mathbf{u}_{N_{\mathbf{u}}+1} \sim \mathbb{P}_{\mathbf{u}}$. The first term $(\sum_{i=1}^{N_{\mathbf{u}}} |\mathbf{c}^\top F^*(\hat{\mathbf{u}}_i) - \mathbf{c}^\top \mathbf{x}^\lambda(\hat{\mathbf{u}}_i)|)/N_{\mathbf{u}}$, is the empirical error associated with solving $\mathbf{GBP}(\hat{\mathcal{U}}, B_\delta, \lambda)$ versus $\mathbf{BP}(\hat{\mathbf{u}}_i)$ for all $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$. It effectively measures how well the model performs on in-sample data. The second term is dependent on the constants K and $1/\beta$ and scales with $O(1/N_{\mathbf{u}})$. Thus, as the size of the data set increases, the smaller this term becomes. The third term is dependent the Rademacher complexity of $\mathcal{F}$. The greater the representative capacity of the learning model the larger its Rademacher complexity. The summation of these terms is a prediction bound on the optimality of unseen instances and holds with probability at least $1 - \beta$. Thus, in order to obtain a tight and useful bound, we must balance the trade-off between a model class with high complexity versus obtaining a model with low empirical error.

7. Case studies

In this section, we perform two numerical case studies. In the first set of experiments, we analyze an application that typically uses optimization to produce optimal decisions. Here, we consider an online automated investment service whose users require investment strategies yielding personalized risk tolerances and investment quotas (Swezey and Charron 2018, Li 2020). Many first-time users, who represent the target market of these services (e.g., WealthSimple), may not be able to accurately identify their ideal investment parameters. A conventional solution may be to estimate the parameters from user profiles or questionnaires (e.g., age, income level, occupation) and then solve a non-convex portfolio optimization problem with user-dependent parameters. However, online services typically require latencies of less than 1 second (Crankshaw et al. 2015, Zhao et al. 2018). For example, Amazon estimates each 100 millisecond (ms) delay corresponds to a 1% drop in sales while Google estimates each 500 ms delay corresponds to a 20% traffic loss for their respective mobile applications (Wang et al. 2012). Thus, a mobile service employing mathematical programming for real-time user interfacing will undoubtedly be too slow. Instead, we train a generative model using IPMAN to estimate decisions that replicate the feasibility and optimality performance of a non-convex optimization solver such as Gurobi (Gurobi Optimization 2020). This model can then be deployed by the online service to produce high-quality solutions to contextual non-convex optimization problems in milliseconds. We compare the performance of our model to the strongest version of the predict-then-optimize paradigm, i.e., one that can perfectly predict the right parameters for every context. We find that:

1. IPMAN-trained models generate portfolios in approximately 130 ms, while a predict-then-optimize baseline often requires up to 100 s. Moreover, on average, the portfolios generated by our approach are within a 3% optimality gap.
2. A predict-then-optimize approach requires nearly perfect predictions or else the optimizer produces infeasible solutions with respect to the true constraints. Analogously, even with a noisy oracle that may label decisions incorrectly, our models can make conservative decision estimates that are fairly robust to labelling errors.

In the second case study, we analyze an application that currently involves deep learning to predict decisions. Radiation therapy (RT) is a cancer treatment procedure in which high-energy x-rays deliver a distribution of radiation over cubic segments of the patient's

body (Delaney et al. 2005). Clinics mandate that treatment plans should meet a set of non-convex pass-fail criteria, however for most patients, the criteria cannot all be simultaneously satisfied. Consequently, treatment planners may manually tune the parameter settings of a radiation therapy planning optimization software until they obtain a treatment plan that is approved by the referring oncologist. Automated planning methods, alternatively, directly predict personalized dose distributions in order to reduce the time spent on plan generation (Wu et al. 2017). This involves estimating a dose that satisfies the relevant subset of criteria that an oncologist would have required for the patient. Currently, automated planning methods use conventional supervised learning techniques with a data set of historical patient spatial features (e.g., the sizes and locations of healthy organs and tumors) and clinically delivered dose distributions. Instead, by treating the relevant criteria that must be satisfied for each individual as a set of patient-specific hidden constraints, IPMAN can estimate doses that better meet clinical metrics (i.e., feasibility) and simultaneously deliver lower radiation to healthy tissue (i.e., optimality). We find that:

1. Compared to supervised learning baselines, IPMAN-trained models yield dose decisions that are feasible 7% more often and result in 7% less radiation to healthy tissue.
2. By encoding clinical policies as constraints, IPMAN-trained models can produce treatments tailored to one clinic’s metrics while learning from past decisions of another clinic. IPMAN can thus be used by new clinics that do not yet have sufficient data to train their own models.

7.1. Automated portfolio optimization: A comparison with optimization

Suppose an automated broker must generate portfolio recommendations for different users by modeling a personalized variant of the non-convex, cardinality constrained portfolio optimization problem (see Kim et al. 2019) with $n = 100$ stocks and $p = 5$ contextual features per user. To train a predictive model, the broker may select a training cohort of users, utilize their profile information as features, and collect past portfolios that were recommended to them. Several recently developed predict-then-optimize frameworks have shown exceptional performance estimating parameters for portfolio optimization problems (e.g., Ban et al. 2018, Elmachtoub and Grigas 2017, Ferber et al. 2020, Yu et al. 2020). Thus, we use an *oracle-then-optimize* framework as the baseline model which perfectly predicts the correct parameters for each user and solves a non-convex optimization problem. In comparison, our IPMAN-trained model only learns to directly predict optimal portfolios.

7.1.1. Data and model. We first generate three linear quantities, $\mathbf{b}_r, \mathbf{b}_c, \mathbf{b}_\Delta \in \mathbb{R}_+^p$, that represent the “true relationships” between (i) the risk upper bound $r(\mathbf{u}) = \mathbf{b}_r^\top \mathbf{u}$; (ii) the minimum cardinality $c(\mathbf{u}) = \mathbf{b}_c^\top \mathbf{u} + 1$; and (iii) the cardinality bandwidth $\Delta c(\mathbf{u}) = \mathbf{b}_\Delta^\top \mathbf{u} + 1$. Without loss of generality, these relationships are generated so that the average user desires a low-risk portfolio of between five to fifteen stocks. For experiments with a perfect oracle (see Sec 7.1.2), the users have an average risk of $\mathbb{E}[r(\mathbf{u})] = 0.066$, a minimum cardinality of $\mathbb{E}[c(\mathbf{u})] = 7$, and a minimum bandwidth of $\mathbb{E}[\Delta c(\mathbf{u})] = 4$. For the experiments with a noisy oracle (see Sec 7.1.3), we consider a slightly easier problem that can be solved quicker; the users here have an average risk of $\mathbb{E}[r(\mathbf{u})] = 0.076$, a minimum cardinality of $\mathbb{E}[c(\mathbf{u})] = 12$, and a minimum bandwidth of $\mathbb{E}[\Delta c(\mathbf{u})] = 6$.

Given a data generation model for user behavior, we sample a data set of i.i.d. randomly sampled context vectors $\mathbf{u} \in \mathbb{R}_+^p$ for 7,000 users and randomly split the data into three groups of size 5,000, 1,000, and 1,000 for training, validation, and hold-out-testing, respectively. User parameters input to the personalized portfolio optimization problem

$$\begin{aligned} \text{PPP}(\mathbf{u}) : \quad & \underset{\mathbf{x}}{\text{minimize}} \quad -\boldsymbol{\mu}^\top \mathbf{x} \\ & \text{subject to} \quad \mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x} \leq r(\mathbf{u}) \\ & \quad \lfloor c(\mathbf{u}) \rfloor \leq \text{card}(\mathbf{x}) \leq \lfloor c(\mathbf{u}) \rfloor + \lceil \Delta c(\mathbf{u}) \rceil \\ & \quad \mathbf{x} \in \mathcal{P} := \left\{ \mathbf{x} \mid \mathbf{x}^\top \mathbf{1} \leq 1, \mathbf{x} \geq \mathbf{0} \right\} \end{aligned}$$

where $\boldsymbol{\mu} \in \mathbb{R}_+^n$ is the return vector, $\boldsymbol{\Sigma} \in \mathbb{S}_+^{n \times n}$ is the risk matrix, and $\mathcal{X}(r(\mathbf{u}), c(\mathbf{u}), \Delta c(\mathbf{u})) := \{\mathbf{x} \in \mathcal{P} \mid \mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x} \leq r(\mathbf{u}), \lfloor c(\mathbf{u}) \rfloor \leq \text{card}(\mathbf{x}) \leq \lfloor c(\mathbf{u}) \rfloor + \lceil \Delta c(\mathbf{u}) \rceil\}$ represents the user-dependent hidden feasible set. Note that our data-driven portfolio optimization problem differs from the common setting where $\boldsymbol{\mu}$ and/or $\boldsymbol{\Sigma}$ must be determined. In our case, we assume $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are known but parameters for the constraints must be estimated (Li 2020, Yu et al. 2020). We also assume that for our training set, we queried each user to obtain their true parameters $r(\hat{\mathbf{u}}_i)$, $c(\hat{\mathbf{u}}_i)$, and $\Delta c(\hat{\mathbf{u}}_i)$. In other words, the oracle used for training is:

$$\Psi(\mathbf{x}, \hat{\mathbf{u}}_i) = \begin{cases} 0 & \text{if } \mathbf{x} \notin \mathcal{P} \\ 0 & \text{if } \mathbf{x} \notin \mathcal{X}(\mathbf{b}_r^\top \hat{\mathbf{u}}_i, \mathbf{b}_c^\top \hat{\mathbf{u}}_i + 1, \mathbf{b}_\Delta^\top \hat{\mathbf{u}}_i + 1), \\ 1 & \text{otherwise} \end{cases} \quad \forall \hat{\mathbf{u}}_i \in \hat{\mathcal{U}} \quad (10)$$

To construct our initial data sets of feasible ($\mathcal{D}^{(1)}$) and infeasible ($\bar{\mathcal{D}}^{(1)}$) decisions, we generate 10 feasible and 10 infeasible portfolios per context vector $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$ (for a total of 100,000 data points) by randomly solving convex approximations and relaxations of the problem. We then implement our generator and classifier as two feed-forward neural networks and train IPMAN for $K = 80$ iterations. Rather than reporting a set of J results from each of the dual regularizers, we report only $\lambda = 4 \times 10^{-4}$. This value was selected via cross-validation using our training and validation sets. We implement IPMAN using PyTorch with an Nvidia GTX 1080 GPU. At test time, our generator can predict a decision in approximately 130 ms. For the baseline models, we use Gurobi 8.1.1 to optimally solve $\mathbf{PPP}(\mathbf{u})$ and set a time out for the optimizer at 100 s. Additional details on data generation, model formulation, and the baseline models are provided in EC.4.

7.1.2. Experiment 1: Quickly predicting near-optimal solutions. We evaluate our generator over the set of solvable $\mathbf{PPP}(\hat{\mathbf{u}}_i)$ by evaluating the average optimality gap and the fraction of predictions that satisfy constraints. We define feasibility as meeting all constraints to within a relative tolerance of 5%. Note that Kim et al. (2019) use approximately the same tolerance level for their related work on predicting optimal portfolios.

Figure 5 shows the performance IPMAN on the validation set for each iteration of training. The plots demonstrate that the generator quickly learns to predict high-return portfolios. However, the classifier needs many more iterations to learn acceptable levels of feasibility. This is because, for $K = 1$, the data set contains at most 20 portfolios per user $\hat{\mathbf{u}}_i$. In this small-data regime, it is difficult for the classifier to learn a general function that can act as a barrier for every user; this motivates the use of active learning. As the data set is progressively augmented, the classifier learns a barrier that better approximates the feasible set and the generator learns to more consistently produce optimal portfolios.

Table 1 compares IPMAN with a commercial optimization solver stratified by optimizer runtime. Predicting a decision with IPMAN requires, on average, 130 ms, whereas the optimizer requires, on average, 3 s with many instances requiring several minutes to solve. Furthermore, for problems that require up to 100 s for the commercial optimizer, the generator consistently predicts decisions that are both feasible and optimal, thereby perfectly replicating optimizer performance in three orders of magnitude less time. For problems with optimizer runtimes greater than 100 s, our estimates are often infeasible as these

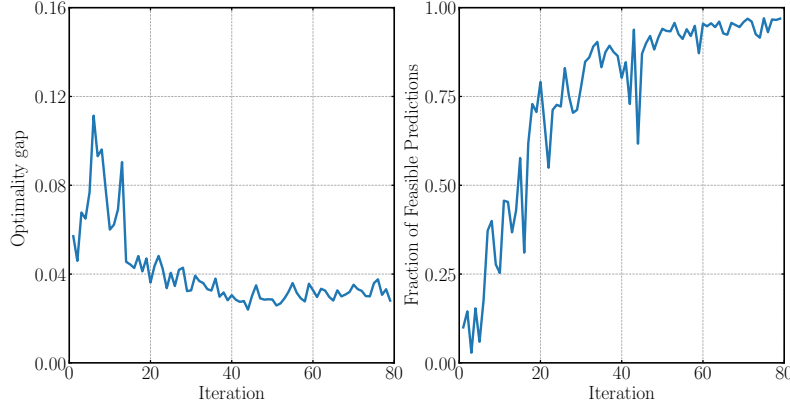


Figure 5 Average optimality gap and fraction of feasible predictions for IPMAN on the validation set obtained during training. Instances where the optimizer fails (> 100 s runtime) are omitted.

Table 1 Summary statistics comparing IPMAN with an optimizer baseline on the testing set. The ‘All points’ test statistic includes instances that have runtimes less than 100 s. Optimality gap is measured over only feasible decisions. The IPMAN runtime is approximately 0.13 s for all problems.

	All points	Optimizer runtime windows				
		0–0.1 s	0.1–1 s	1–10 s	10–100 s	>100 s
Average optimizer runtime (s)	2.67	0.07	0.27	3.11	41.8	–
Optimality gap (%)	2.53	2.00	2.67	4.40	3.37	–
Fraction of predictions $\mathbf{x} \in \mathcal{X}(\mathbf{u})$ (%)	96.0	99.4	96.3	87.0	77.8	5.61

problems typically feature low risk and cardinality bandwidth parameters. We hypothesize that the tight cardinality constraints lead to pathological feasible sets such that it becomes difficult for both prediction and optimization.

Finally, recall that these experiments compare against an oracle-then-optimize baseline that always knows the true parameter values. A realistic predict-then-optimize framework would inevitably introduce estimation errors in the parameters. We next show that prediction errors in constraint parameters lead to large optimization errors whereas IPMAN can generate feasible decisions regardless of how inaccurate the parameter estimates are.

7.1.3. Experiment 2: Predicting optimal solutions when the oracle is noisy. Analogous to an oracle-then-optimize approach, IPMAN assumes a perfect feasibility oracle in

Table 2 Summary statistics with mean and standard deviation over five trials comparing IPMAN with an optimizer baseline on the testing set. Optimality gap is evaluated over only feasible decisions. The best approach for each noise level is shaded.

Noise σ	Average optimality gap (%)		Fraction of predictions feasible (%)	
	IPMAN	Oracle-then-optimize	IPMAN	Oracle-then-optimize
0.0025	1.11 ± 0.2	0 ± 0	92.8 ± 1.1	94.3 ± 11.5
0.0075	1.08 ± 0.3	0.23 ± 0.1	89.1 ± 4.5	74.3 ± 25.1
0.0125	1.07 ± 0.5	0.35 ± 0.4	78.2 ± 9.1	48.0 ± 42.3
0.0175	1.26 ± 0.3	0.8 ± 1.2	84.9 ± 4.2	61.9 ± 37.0

order to perform active learning. In practice, however, there may be a discrepancy between user features and problem parameters. As a result, predict-then-optimize models would introduce estimation errors and IPMAN would not include a perfect feasibility oracle. In this subsection, we explore this issue by considering a noisy oracle for both the baseline and IPMAN. Since the parameter data has a linear relationship with the user’s features ($\mathbf{b}_r$, $\mathbf{b}_c$, and $\mathbf{b}_\Delta$), we replace $\Psi(\mathbf{x}, \hat{\mathbf{u}}_i)$ in (10) with a black-box oracle that employs a noisy linear model $\mathbf{b}_r + \boldsymbol{\epsilon}_r$, $\mathbf{b}_c + \boldsymbol{\epsilon}_c$, and $\mathbf{b}_\Delta + \boldsymbol{\epsilon}_\Delta$ where $\boldsymbol{\epsilon}_r, \boldsymbol{\epsilon}_c, \boldsymbol{\epsilon}_\Delta \sim \mathcal{N}(0, \sigma)$.

We perform five trials by progressively increasing the variance of the noisy oracle, $\sigma \in \{0.0025, 0.0075, 0.0125, 0.0175\}$. In Table 2, we compare IPMAN and the noisy oracle-then-optimize baseline by analyzing the mean and standard deviation of the average optimality gap and the fraction of feasible predictions on the test set for each trial. IPMAN produces solutions with a small optimality gap (around 1%) while ensuring that decisions are largely feasible. Further, no statistically significant degradation in the optimality gap is observed as the variance of the noise increases; the fraction of feasible predictions is also relatively stable. While the noisy oracle-then-optimize baseline produces solutions with almost no optimality gap, a much larger proportion of the decisions are infeasible when considering the true portfolio parameters. The difference in the fraction of feasible predictions between IPMAN and the noisy oracle-then-optimize baseline is stark, e.g., on average, almost 40% of test set solutions produced by the commercial optimizer are infeasible when $\sigma = 0.0175$. Further, as the noise increases, the fraction of feasible predictions decreases while the standard deviation increases. Notably, the standard deviation is almost an order of magnitude larger as compared to IPMAN.

The intuition behind these results is that the optimizer incorrectly assumes that the constraint parameters are perfect. For example, if the noisy oracle aggressively estimates larger parameter values (i.e., $\epsilon_r, \epsilon_c, \epsilon_\Delta > 0$), the optimizer incorrectly operates on a relaxed feasible set leading to infeasible decisions. On the other hand, by minimizing $\mathbf{GBP}(\hat{\mathcal{U}}, \lambda, B^*)$, IPMAN training includes a regularization term in the objective (i.e., the barrier function) which assists in generalization. By conservatively setting λ , we ensure that IPMAN predicts decisions that remain feasible. IPMAN can then maintain effectiveness even when the oracle generates noisy labels.

7.2. Personalized cancer treatment: A comparison with deep learning

In automated treatment planning, a predictive model generates a dose that satisfies the relevant clinical metrics for a given patient. Since oncologists typically consider spatial features such as the size and location of both healthy organs and tumor sites, estimation is typically performed with deep neural networks. Using a supervised learning approach, the neural network learns to map a patient’s contoured computed tomography (CT) image to a 3-D dose distribution (Kearney et al. 2018, Mahmood et al. 2018, Babier et al. 2019). Dose predictions are evaluated on whether they satisfy the same set of pass-fail metrics as the clinical doses, which current supervised learning techniques do not optimize. With IPMAN, we encode the metrics as constraints and the hidden feasible set as the metrics that the real doses must satisfy. Thus, we form a non-convex optimization problem with hidden constraints and learn to generate clinically satisfiable doses. Below, we highlight the setup and key results; the full data set and model, as well as additional results, are provided in the Electronic Companion.

7.2.1. Data and model. We use a data set of clinical dose distributions for 217 patients with head-and-neck cancer to train IPMAN to predict radiation therapy treatment plans with certificates of optimality. We randomly split the data into 100, 67, and 50 patients for training, validation, and hold-out testing, respectively. We then apply a data augmentation step to update the 100 doses into approximately 5,000 doses (i.e., 50 per patient) binned into feasible and infeasible training sets. Each patient is discretized into a tensor whose elements represent voxels (volumetric pixels) of the patient’s geometry. For each patient i , the clinical data set contains a 3-D dose distribution $\hat{\mathbf{x}}_i$ (representing the treatment decision) that was prescribed by the oncologist and a contoured CT image $\hat{\mathbf{u}}_i$ (representing the context vector). Full details on the data and models is given in EC.5.2

Each patient has up to seven structures that must be considered. There are four organs-at-risk (OARs)—Right Parotid, Left Parotid, Larynx, Mandible—that can receive up to a maximum amount of radiation. There are also three tumor sites, called planning target volumes (PTVs)—PTV70, PTV63, and PTV56 where the number indicates the amount of radiation each structure should receive. Let $\mathcal{O}$ and $\mathcal{T}$ index the OARs and PTVs, respectively, and $\mathcal{R} := \mathcal{O} \cup \mathcal{T}$ index all structures of interest. For each structure $r \in \mathcal{R}$, let $\mathcal{V}_r$ index the voxel set, i.e., the elements of $\mathbf{x}$ and $\mathbf{u}$ that correspond to structure r . Each structure is evaluated using a pass-fail clinical criteria $g_r(\mathbf{x}) \leq z_r$ (see column 2 of Table 3). Our non-convex radiation therapy (RT) optimization problem minimizes the sum of average doses measured in grays (Gy) delivered to the OARs subject to patient-specific clinical criteria and a fixed set of known safety constraints on the average dose (see Babier et al. 2018):

$$\begin{aligned} \mathbf{RTP}(\mathbf{u}): \quad & \underset{\mathbf{x}}{\text{minimize}} \quad \frac{1}{|\mathcal{O}|} \sum_{o \in \mathcal{O}} \frac{1}{|\mathcal{V}_o|} \sum_{v \in \mathcal{V}_o} x_v \\ & \text{subject to} \quad \mathbf{x} \in \mathcal{X}_r(\mathbf{u}), \quad \forall r \in \mathcal{R} \\ & \quad \mathbf{x} \in \mathcal{P} := \left\{ \mathbf{x} \left| \underline{x}_r \leq \frac{1}{|\mathcal{V}_r|} \sum_{v \in \mathcal{V}_r} x_v \leq \bar{x}_r, \forall r \in \mathcal{R} \right. \right\} \end{aligned}$$

where each $\mathcal{X}_r(\mathbf{u})$ are hidden feasible sets that enforce the clinical criteria as patient-specific constraints. If a clinician were to review the patient’s CT image and determine that the clinical criteria for structure r is mandatory, $\mathcal{X}_r(\mathbf{u})$ must contain that criteria. Otherwise, the criteria reverts to the polyhedral dose bound. In our problem, we determine whether the criteria is necessary by reviewing the real clinical doses in our data set of treatments, i.e.,

$$\mathcal{X}_r(\hat{\mathbf{u}}) := \left\{ \mathbf{x} \left| \begin{array}{ll} g_r(\mathbf{x}) \leq z_r, & \text{if } g_r(\hat{\mathbf{x}}) \leq z_r, \forall (\hat{\mathbf{x}}, \hat{\mathbf{u}}) \in \mathcal{D} \\ \underline{x}_r \leq \frac{1}{|\mathcal{V}_r|} \sum_{v \in \mathcal{V}_r} x_v \leq \bar{x}_r, & \text{otherwise} \end{array} \right. \right\} \quad \forall r \in \mathcal{R}, \hat{\mathbf{u}} \in \hat{\mathcal{U}}$$

Note that while the otherwise condition represents the case where the clinical constraint can be ignored, we define it above without loss of generality so that $\mathcal{X}_r(\hat{\mathbf{u}}_i) \subset \mathcal{P}$.

Since assessing whether a generated decision is feasible amounts to checking whether the relevant clinical criteria are satisfied, we implement a data-driven rule-based oracle (see Example 1). For any patient in the training set, we first evaluate the clinical dose from the

feasible decision data set and assess whether the generated dose satisfies the same criteria. Due to its computational simplicity, we can easily call the oracle a large number of times in training in order to augment our data set and improve our models.

We implement IPMAN using two convolutional neural network models. Our generator and classifier use a modified architecture from the generative adversarial network (GAN) of Babier et al. (2019). We train four different generative models $F^{(j,k)}(\mathbf{u})$ for $\lambda_j \in \{256, 64, 16, 4\}$ for 11 iterations using our training set. At the end of each iteration, our oracle labels the generator outputs for data augmentation. We then evaluate each of the generators on the validation set of 67 patients. After 11 iterations, error on the validation set stabilizes and we use the hold-out test set to assess the performance of our models. We report below the best performing models and leave the results with all models in EC.5.3 and EC.5.4.

We compare against two state-of-the-art baseline models from the clinical automated planning literature: a U-net convolutional neural network (CNN) (Nguyen et al. 2019) and a generative adversarial network (GAN) (Babier et al. 2019). The CNN is trained by minimizing the mean-squared error of predicted doses from clinical doses. The GAN is composed of two networks that are trained adversarially in a minimax game.

7.2.2. Experiment 1: Predicting feasible and optimal dose distributions. A predicted dose that satisfies all of the same clinical criteria as the ground truth dose for the given patient is a feasible solution that satisfies all hidden constraints for our optimization problem. Thus, we evaluate all models on the percentage of predictions that individually satisfy each of the hidden constraints as well as the percentage that satisfy all constraints simultaneously. Satisfaction is defined as meeting the dose bound to within a 1 Gy relaxation, which is within acceptable limits (Low et al. 1998).

Table 3 shows performance on the hold-out test set for IPMAN at iteration 11 against the two baseline models. IPMAN dominates both baselines in hidden and polyhedral constraint satisfaction as well as the average objective function value. That is, this model predicts dose distributions that deliver lower dose to healthy tissue while better satisfying the clinical criteria. We conclude that IPMAN-trained models produce feasible decisions more often and with a lower objective function value on average as compared to existing state-of-the-art methods.

Table 3 The percentage of predicted decisions on the hold-out test set that satisfy each hidden constraint to 1 Gy relaxation. The best performing models on the summary statistics are highlighted.

Structure r	Criteria (Gy)	Baselines		IPMAN
		GAN	CNN	$\lambda = 4$
	$g_r(\mathbf{x}) \leq z_r$			
Right Parotid	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 26$	85.7	85.7	100
Left Parotid	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 26$	70.0	60.0	100
Larynx	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 45$	93.3	83.3	100
Mandible	$\text{max}\{x_v \mid v \in \mathcal{V}_r\} \leq 73.5$	100	100	100
PTV70	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 70$	97.6	97.6	92.8
PTV63	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 63$	96.3	96.3	96.3
PTV56	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 56$	100	100	100
Fraction of predictions $\mathbf{x} \in (\cap_r \mathcal{X}_r(\mathbf{u}))$ (%)		86.0	78.0	94.0
Fraction of predictions $\mathbf{x} \in \mathcal{P}$ (%)		92.0	90.0	94.0
Average objective function value (Gy)		40.3	41.0	37.8

7.2.3. Experiment 2: Adapting to the policies of different institutions. In RT, different clinics often mandate different sets of clinical criteria (Wu et al. 2017), meaning that a data set of treatments obtained from one clinic may not be indicative of institutional policies at another. Further, new clinics may have limited patient volumes making it difficult to obtain sufficient data to properly train dose estimation models using conventional supervised learning techniques (Boutilier et al. 2016). However, IPMAN can circumvent this issue by encoding specific clinical criteria into the computational oracle and then adaptively learn to generate doses that satisfy the clinic’s policies. For example, IPMAN can use a large data set obtained from a high-volume clinic to train a model that will be applied at a new oncology practice with slightly different oncological specifications.

In this experiment, we consider the clinical criteria obtained from Geretschläger et al. (2015) who pursue more aggressive treatment policies as compared to the clinic that produced our data. Geretschläger et al. (2015) specify that the three tumors should receive 72 Gy, 66 Gy, and 54 Gy, respectively. Thus, we re-label our target sites to PTV72, PTV66, and PTV54 and update the clinical criteria bound z_r . In comparison to Section 7.2.2, two of the criteria have become stricter while the third is now easier to satisfy. Although we

Table 4 The percentage of predicted decisions on the hold-out test set that satisfy each hidden constraint of Geretschläger et al. (2015) to 1 Gy relaxation. The best performing models on the summary statistics are highlighted.

Structure	Criteria (Gy)	Baseline IPMAN	
	$g_r(\mathbf{x}) \leq z_r$	GAN	$\lambda = 256$
Right Parotid	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 26$	83.3	85.7
Left Parotid	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 26$	70.0	50.0
Larynx	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 45$	93.3	86.7
Mandible	$\text{max}\{x_v \mid v \in \mathcal{V}_r\} \leq 73.5$	100	81.0
PTV70	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 70$	7.31	95.2
PTV63	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 63$	77.8	96.3
PTV56	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 56$	100	100
Fraction of predictions $\mathbf{x} \in (\cap_r \mathcal{X}_r(\mathbf{u}))$ (%)		18.3	64.0
Fraction of predictions $\mathbf{x} \in \mathcal{P}$ (%)		93.8	94.0
Average objective function value (Gy)		40.3	42.3

do not know the exact oncologist preferences in determining when a criteria is necessary, we assume that patients for whom the PTV criteria is necessary in our original clinic would require a more aggressive level of radiation at the new clinic. Consequently, we train IPMAN for 11 iterations with updated criteria for the oracle.

Table 4 shows the performance on the hold-out test set for the generator at iteration 11. Here, our best model uses $\lambda = 256$. In comparison to the previous experiment, we prefer a higher λ since it encourages more conservative training with respect to feasibility, which is aligned with our goal of satisfying the clinical criteria. The baseline for comparison is the GAN model which was trained on data from the original institution. Since no new data exists, the GAN cannot be re-trained to recognize the updated clinical criteria. As a result, few plans (18.3%) produced by the GAN baseline satisfy the hidden constraints with only 77.8% and 7.31% of the PTV66 and PTV72 criteria being satisfied, respectively. In contrast, IPMAN is able to learn the new hidden constraints. Overall, hidden constraint satisfaction is 64.0% with 96.3% and 95.2% of plans satisfying the PTV66 and PTV72, constraints, respectively. Nevertheless, as a result of learning higher doses to the PTVs,

constraint satisfaction on the OARs slightly degrades; this can be observed in the slightly higher objective function values.

8. Discussion and conclusion

The approach introduced in this paper addresses three issues that affect the deployment of prediction tools in environments where high-quality decisions are critical (e.g., Lamanna and Byrne 2018). First, although current trends in machine learning research have focused on pursuing interpretable or white-box models (e.g., Wang et al. 2020), in many settings, safety issues may arise due to the sub-optimality of the predicted decision (Bertsimas et al. 2020). Several papers have demonstrated that there is significant value in optimally solving a prediction problem (Bertsimas and Dunn 2017, Hu et al. 2019, Berk and Bertsimas 2019). Nevertheless, in large-data environments, there exist few approaches that provide guarantees on the quality of the predicted solutions. This paper addresses the issue by presenting a generative prediction model that admits a generalization of the classical ϵ -optimality property of interior point methods. In addition, the aforementioned IPMAN algorithm produces these solutions while learning from past decisions that were not necessarily optimal.

Second, there are many instances where workers deviate from standard processes that have been determined by management, even when their actions may have serious consequences (Bendoly 2011, 2013, Cotteleer and Bendoly 2015, Chan et al. 2020b). In these settings, formulating constraints to construct optimization models based on the application of standard decision rules will inevitably widen the gap between established guidelines and what is actually done. Since process deviations may be negatively perceived by management, querying workers on what they do in practice is also not a viable solution. Instead, the IPMAN framework can learn the hidden process deviations directly from the data. The resulting constraints can then be incorporated into the decision model by solely examining the outcome, i.e., the labels for the predicted decision.

Finally, the active learning component of IPMAN represents a simplified way of incorporating expert judgement into the development of automated decision-making processes. For instance, having medical practitioners evaluate the acceptability of the predicted decisions based on their hidden preferences allows them to participate in the creation of the algorithm. Since this process allows practitioners to understand the interactions which led

to the development of the model, they may be more comfortable adopting the tool in their practices (Vollmer et al. 2020, Faes et al. 2020). It also gives them an opportunity to interact with advanced technology without having to understand complex mathematical relations nor high-level programming languages. The result is a model that is tailored to the hidden preferences of the decision-makers (e.g., medical practitioners) while still retaining its ability to generate certifiably optimal decisions.

In summary, this paper presents a new machine learning-based approach, leveraging adversarial and active learning, to generate solutions with optimality guarantees to contextual optimization problems. In developing IPMAN, we generalize results from interior point methods, and present new out-of-sample optimality bounds for any model that predicts solutions to a constrained optimization problem. Our numerical results on two case studies – a portfolio optimization problem and the design of personalized radiation therapy treatments – suggest that significant computational speedups and solution quality improvements are possible against traditional machine learning and operations research approaches to solving contextual optimization problems.

References

- Agrawal A, Amos B, Barratt S, Boyd S, Diamond S, Kolter JZ (2019) Differentiable convex optimization layers. *Advances in neural information processing systems*, 9562–9574.
- Amos B, Kolter JZ (2017) Optnet: Differentiable optimization as a layer in neural networks. *International Conference on Machine Learning*, 136–145.
- Angalakudati M, S B, Calzada J, Chatterjee B, Perakis G, Raad N, Uichanco J (2014) Business analytics for flexible resource allocation under random emergencies. *Management Science* 60(6):1552–1573.
- Arjovsky M, Bottou L (2017) Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862* .
- Babier A, Boutilier JJ, Sharpe MB, McNiven AL, Chan TCY (2018) Inverse optimization of objective function weights for treatment planning using clinical dose-volume histograms. *Physics in Medicine & Biology* 63(10):105004.
- Babier A, Mahmood R, McNiven AL, Diamant A, Chan TCY (2019) Knowledge-based automated planning with three-dimensional generative adversarial networks. *forthcoming in Medical Physics* .
- Badenbroek R, de Klerk E (2018) Complexity analysis of a sampling-based interior point method for convex optimization. *arXiv preprint arXiv:1811.07677* .
- Ban GY, El Karoui N, Lim AEB (2018) Machine learning and portfolio optimization. *Management Science* 64(3):1136–1154.

- Ban GY, Rudin C (2018) The big data newsvendor: Practical insights from machine learning. *Operations Research* 1(67):90–108.
- Bansal S, Pachamanova D (2019) Special issue on nonconvex portfolio optimization. *The Engineering Economist* 64(3):193–195, URL <http://dx.doi.org/10.1080/0013791X.2019.1636177>.
- Bartlett PL, Mendelson S (2002) Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research* 3(Nov):463–482.
- Bello I, Pham H, Le QV, Norouzi M, Bengio S (2017) Neural combinatorial optimization.
- Bendoly E (2011) Linking task conditions to physiology and judgment errors in rm systems. *Production and Operations Management* 20(6):860–876.
- Bendoly E (2013) Real-time feedback and booking behavior in the hospitality industry: Moderating the balance between imperfect judgment and imperfect prescription. *Journal of Operations Management* 31(1-2):62–71.
- Bengio Y, Lodi A, Prouvost A (2020) Machine learning for combinatorial optimization: a methodological tour d’horizon. *European Journal of Operational Research* .
- Benson HY, Shanno DF, Vanderbei RJ (2004) Interior-point methods for nonconvex nonlinear programming: jamming and numerical testing. *Mathematical Programming* 99(1):35–48.
- Berk L, Bertsimas D (2019) Certifiably optimal sparse principal component analysis. *Mathematical Programming Computation* 11(3):381–420.
- Bertsimas D, Dunn J (2017) Optimal classification trees. *Machine Learning* 106(7):1039–1082.
- Bertsimas D, Kallus N (2019) From predictive to prescriptive analytics. *Management Science* 0(0).
- Bertsimas D, McCord C (2018) Optimization over continuous and multi-dimensional decisions with observational data. *Advances in Neural Information Processing Systems*, 2966–2974,.
- Bertsimas D, Orfanoudaki A, Weiner RB (2020) Personalized treatment for coronary artery disease patients: a machine learning approach. *Health Care Management Science* 1–25.
- Boutillier JJ, Craig T, Sharpe MB, Chan TCY (2016) Sample size requirements for knowledge-based treatment planning. *Medical Physics* 43(3):1212–21.
- Brinker K (2003) Incorporating diversity in active learning with support vector machines. *Proceedings of the 20th international conference on machine learning (ICML-03)*, 59–66.
- Bubeck S, Eldan R (2019) The entropic barrier: Exponential families, log-concave geometry, and self-concordance. *Mathematics of Operations Research* 44(1):264–276.
- Castrejon L, Kundu K, Urtasun R, Fidler S (2017) Annotating object instances with a polygon-rnn. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5230–5238.
- Chan CW, Farias VF, Bambos N, Escobar GJ (2012) Optimizing intensive care unit discharge decisions with patient readmissions. *Operations research* 60(6):1323–1341.

- Chan TCY, Diamant A, Mahmood R (2020a) Sampling from the complement of a polyhedron: An mcmc algorithm for data augmentation. *Operations Research Letters* 48(6):744–751.
- Chan TCY, Howard N, Lagzi S, Romero G (2020b) Cherry-picking and its negative spillover effect on system service level: Evidence from a radiology workflow platform. *Available at SSRN 3273494* .
- Chen T, Kornblith S, Swersky K, Norouzi M, Hinton G (2020) Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029* .
- Cotteleer M, Bendoly E (2015) How behavioral factors affect decisions related to work process deviation URL <https://www2.deloitte.com/za/en/insights/focus/behavioral-economics/applying-behavioral-principles>
- Crankshaw D, Bailis P, Gonzalez JE, Li H, Zhang Z, Franklin MJ, Ghodsi A, Jordan MI (2015) The missing piece in complex analytics: Low latency, scalable model management and serving with velox. *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*.
- Delaney G, Jacob S, Featherstone C, Barton M (2005) The role of radiotherapy in cancer treatment. *Cancer* 104(6):1129–1137.
- Donti P, Amos B, Kolter JZ (2017) Task-based end-to-end model learning in stochastic optimization. *Advances in Neural Information Processing Systems*, 5484–5494.
- Elmachtoub AN, Grigas P (2017) Smart “predict, then optimize”. *arXiv preprint arXiv:1710.08005* .
- Engelking R (1977) *General Topology* (Polish Scientific Publishers).
- Faes L, Liu X, Wagner SK, Fu DJ, Balaskas K, Sim DA, Bachmann LM, Keane PA, Denniston AK (2020) A clinician’s guide to artificial intelligence: How to critically appraise machine learning studies. *Translational Vision Science & Technology* 9(2):7–7.
- Ferber A, Wilder B, Dilkina B, Tambe M (2020) Mipaal: Mixed integer program as a layer. *AAAI*, 1504–1511.
- Ferreira KJ, Lee BHA, Simchi-Levi D (2015) Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing & Service Operations Management* 18(1):69–88.
- Foster DJ, Sekhari A, Sridharan K (2018) Uniform convergence of gradients for non-convex learning and optimization. *Advances in Neural Information Processing Systems*, 8759–8770.
- Geretschlager A, Bojaxhiu B, Dal Pra A, Leiser D, Schmucking M, Arnold A, Ghadjar P, Aebbersold DM (2015) Definitive intensity modulated radiotherapy in locally advanced hypopharygeal and laryngeal squamous cell carcinoma: mature treatment results and patterns of locoregional failure. *Radiation Oncology* 10:20.
- Gondzio J (2012) Interior point methods 25 years later. *European Journal of Operational Research* 218(3):587–601.
- Goodfellow I, Bengio Y, Courville A (2016) *Deep Learning*, volume 1 (MIT press Cambridge).

- Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. *Advances in Neural Information Processing Systems*, 2672–2680.
- Gupta V, Rusmevichientong P (2020) Small-data, large-scale linear optimization with uncertain objectives. *Management Science* .
- Gurobi Optimization L (2020) Gurobi optimizer reference manual. URL <http://www.gurobi.com>.
- Hannah L, Powell W, Blei DM (2010) Nonparametric density estimation for stochastic optimization with an observable state variable. *Advances in Neural Information Processing Systems*, 820–828.
- Hinder O, Ye Y (2018) A one-phase interior point method for nonconvex optimization. *arXiv preprint arXiv:1801.03072* .
- Hornik K (1991) Approximation capabilities of multilayer feedforward networks. *Neural Networks* 4(2):251–257.
- Hu X, Rudin C, Seltzer M (2019) Optimal sparse decision trees. *Advances in Neural Information Processing Systems*, 7267–7275.
- Joshi AJ, Porikli F, Papanikolopoulos N (2009) Multi-class active learning for image classification. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2372–2379 (IEEE).
- Kao YH, Roy BV, Yan X (2009) Directed regression. *Advances in Neural Information Processing Systems*, 889–897.
- Kearney V, Chan JW, Haaf S, Descovich M, Solberg TD (2018) Dosenet: a volumetric dose prediction algorithm using 3d fully-convolutional neural networks. *Physics in Medicine & Biology* 63(23):235022.
- Kim Y, Kang D, Jeon M, Lee C (2019) Gan-mp hybrid heuristic algorithm for non-convex portfolio optimization problem. *The Engineering Economist* 64(3):196–226.
- Konda VR, Tsitsiklis JN (2000) Actor-critic algorithms. *Advances in Neural Information Processing Systems*, 1008–1014.
- Kool W, van Hoof H, Welling M (2019) Attention, learn to solve routing problems! *International Conference on Learning Representations*.
- Lamanna C, Byrne L (2018) Should artificial intelligence augment medical decision making? the case for an autonomy algorithm. *AMA journal of ethics* 20(9):902–910.
- Larsen E, Lachapelle S, Bengio Y, Frejinger E, Lacoste-Julien S, Lodi A (2018) Predicting solution summaries to integer linear programs under imperfect information with machine learning. *arXiv preprint arxiv:1807.11876* .
- Li JYM (2020) Inverse optimization of convex risk functions. *Management Science* .
- Low DA, Harms WB, Mutic S, Purdy JA (1998) A technique for the quantitative evaluation of dose distributions. *Medical Physics* 25(5):656–661.

- Mahmood R, Babier A, McNiven A, Diamant A, Chan TCY (2018) Automated treatment planning in radiation therapy using generative adversarial networks. of Machine Learning Research P, ed., *Machine Learning for Health Care*, volume 85.
- Maurer A (2016) A vector-contraction inequality for rademacher complexities. *International Conference on Algorithmic Learning Theory*, 3–17 (Springer).
- Mazyavkina N, Sviridov S, Ivanov S, Burnaev E (2020) Reinforcement learning for combinatorial optimization: A survey. *arXiv preprint arXiv:2003.03600* .
- Mišić VV, Perakis G (2020) Data analytics in operations management: A review. *Manufacturing & Service Operations Management* 22(1):158–169.
- Nesterov Y, Nemirovskii A (1994) *Interior-point polynomial algorithms in convex programming*, volume 13 (SIAM).
- Neyshabur B, Tomioka R, Srebro N (2015) Norm-based capacity control in neural networks. *Conference on Learning Theory*, 1376–1401.
- Nguyen D, Jia X, Sher D, Lin MH, Iqbal Z, Liu H, Jiang S (2019) 3d radiotherapy dose prediction on head and neck cancer patients with a hierarchically densely connected u-net deep learning architecture. *Physics in Medicine & Biology* 64(6):065020.
- Pelikan M, Goldberg DE, Lobo FG (2002) A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications* 21(1):5–20.
- Schroff F, Kalenichenko D, Philbin J (2015) Facenet: A unified embedding for face recognition and clustering. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 815–823.
- Sener O, Savarese S (2018) Active learning for convolutional neural networks: A core-set approach. *International Conference on Learning Representations*.
- Settles B (2009) Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.
- Swezey RME, Charron B (2018) Large-scale recommendation for portfolio optimization. *Proceedings of the 12th ACM Conference on Recommender Systems*, 382–386.
- Vanderbei RJ, Shanno DF (1999) An interior-point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications* 13(1-3):231–252.
- Vinyals O, Fortunato M, Jaitly N (2015) Pointer networks. *Advances in Neural Information Processing Systems*, 2692–2700.
- Vollmer S, Mateen BA, Bohner G, Király FJ, Ghani R, Jonsson P, Cumbers S, Jonas A, McAllister KS, Myles P, et al. (2020) Machine learning and artificial intelligence research for patient benefit: 20 critical questions on transparency, replicability, ethics, and effectiveness. *bmj* 368.
- Wang C, Han B, Patel B, Mohideen F, Rudin C (2020) In pursuit of interpretable, fair and accurate machine learning for criminal recidivism prediction. *arXiv preprint arXiv:2005.04176* .

-
- Wang Z, Lin FX, Zhong L, Chishtie M (2012) How far can client-only solutions go for mobile browser speed? *Proceedings of the 21st international conference on World Wide Web*, 31–40.
- Wu B, Kusters M, Kunze-Busch M, Dijkema T, McNutt T, Sanguineti G, Bzdusek K, Dritschilo A, Pang D (2017) Cross-institutional knowledge-based planning (KBP) implementation and its performance comparison to auto-planning engine (APE). *Journal of the European Society for Therapeutic Radiology and Oncology* 123(1):57–62.
- Yu S, Chen Y, Dong C (2020) Learning time varying risk preferences from investment portfolios using inverse optimization with applications on mutual funds. *arXiv preprint arXiv:2010.01687* .
- Zhao Y, Laser MS, Lyu Y, Medvidovic N (2018) Leveraging program analysis to reduce user-perceived latency in mobile applications. *Proceedings of the 40th International Conference on Software Engineering*, 176–186.

Electronic Companion

EC.1. Proofs of main results

Proof of Theorem 1. For notational simplicity, we use $\mathbf{x}^\lambda$ and $\mathbf{x}^*$ in place of $\mathbf{x}^\lambda(\mathbf{u})$ and $\mathbf{x}^*(\mathbf{u})$, respectively. We first prove that an optimal solution exists for any $\lambda > 0$. From the Weierstrauss Theorem, an optimal solution to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$ exists if there is a sub-level set of the objective that is non-empty, bounded, and closed. Let $\bar{\mathbf{x}}$ be a point in the interior of $\mathcal{X}(\mathbf{u})$; by Assumption 1, $\bar{\mathbf{x}}$ exists. Furthermore, $B_\delta(\mathbf{x}, \mathbf{u}) > 0$ for all $\mathbf{x} \in \mathcal{X}(\mathbf{u})$ implies that $\bar{\mathbf{x}}$ admits a finite objective for $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$. Moreover, the objective for $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$ is only finite within $\{\mathbf{x} \mid B_\delta(\mathbf{x}, \mathbf{u}) > 0\} \subseteq \mathcal{N}_\delta(\mathcal{X}(\mathbf{u}))$. Consequently the sub-level set

$$\left\{ \mathbf{x} \mid \mathbf{c}^\top \mathbf{x} - \lambda \log B_\delta(\mathbf{x}, \mathbf{u}) \leq \mathbf{c}^\top \bar{\mathbf{x}} - \lambda \log B_\delta(\bar{\mathbf{x}}, \mathbf{u}) \right\} \subseteq \mathcal{N}_\delta(\mathcal{X}(\mathbf{u}))$$

is non-empty, bounded, and closed (since the objective is continuous) and $\mathbf{x}^\lambda(\mathbf{u})$ exists.

Suppose we choose $\epsilon = -\lambda \log B_\delta(\mathbf{x}^*, \mathbf{u})$. By definition, $0 < B_\delta(\mathbf{x}^*, \mathbf{u}) < 1$, meaning $C := -\log B_\delta(\mathbf{x}^*, \mathbf{u}) > 0$ is a positive constant. Let $\mathbf{x}^\lambda$ be an optimal solution to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$. We now prove that $\mathbf{c}^\top \mathbf{x}^\lambda - \epsilon < \mathbf{c}^\top \mathbf{x}^*$:

$$\begin{aligned} \mathbf{c}^\top \mathbf{x}^* + \epsilon &= \mathbf{c}^\top \mathbf{x}^* - \lambda \log B_\delta(\mathbf{x}^*, \mathbf{u}) \\ &\geq \mathbf{c}^\top \mathbf{x}^\lambda - \lambda \log B_\delta(\mathbf{x}^\lambda, \mathbf{u}) \\ &> \mathbf{c}^\top \mathbf{x}^\lambda. \end{aligned}$$

The first inequality follows from the optimality of $\mathbf{x}^\lambda$ for $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$ while the second inequality follows from $\log B_\delta(\mathbf{x}^\lambda, \mathbf{u}) < 0$, meaning that $-\lambda \log B_\delta(\mathbf{x}^\lambda, \mathbf{u}) > 0$. Moving ϵ to the right-hand-side gives the lower bound.

The proof of $\mathbf{c}^\top \mathbf{x}^* < \mathbf{c}^\top \mathbf{x}^\lambda + \delta$ has two cases. If $\mathbf{x}^\lambda \in \mathcal{X}(\mathbf{u})$, then by the optimality of $\mathbf{x}^*$, we trivially satisfy $\mathbf{c}^\top \mathbf{x}^* \leq \mathbf{c}^\top \mathbf{x}^\lambda < \mathbf{c}^\top \mathbf{x}^\lambda + \delta$. If $\mathbf{x}^\lambda \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$, then let $\tilde{\mathbf{x}} \in \arg \min_{\mathbf{x} \in \mathcal{X}(\mathbf{u})} \|\mathbf{x}^\lambda - \mathbf{x}\|$ be the projection of $\mathbf{x}^\lambda$ on $\mathcal{X}(\mathbf{u})$. Then,

$$\begin{aligned} \mathbf{c}^\top \mathbf{x}^* - \mathbf{c}^\top \mathbf{x}^\lambda &\leq \mathbf{c}^\top \tilde{\mathbf{x}} - \mathbf{c}^\top \mathbf{x}^\lambda \\ &\leq |\mathbf{c}^\top \tilde{\mathbf{x}} - \mathbf{c}^\top \mathbf{x}^\lambda| \\ &\leq \|\tilde{\mathbf{x}} - \mathbf{x}^\lambda\| \\ &< \delta. \end{aligned}$$

The first inequality follows from the optimality of $\mathbf{x}^*$ over $\tilde{\mathbf{x}}$ for $\mathbf{OP}(\mathbf{u})$. The third inequality follows from the Cauchy-Schwartz inequality and $\|\mathbf{c}\| = 1$, while the fourth inequality follows from the fact that for $\mathbf{x}^\lambda \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u}))$, there exists $\mathbf{x} \in \mathcal{X}(\mathbf{u})$ such that $\|\mathbf{x} - \mathbf{x}^\lambda\| < \delta$ and that $\tilde{\mathbf{x}}$ minimizes this distance. This proves the upper bound. $\square$

Proof of Lemma 1. This result is an application of Urysohn's Smooth Lemma, which states that given two closed and disjoint sets $\mathcal{A}$ and $\mathcal{A}'$, there exists a continuous function $f(\cdot) \in [0, 1]$ for which $f(\mathcal{A}) = 1$ and $f(\mathcal{A}') = 0$ (Engelking 1977). Letting $\mathcal{A} = \text{supp}(\mathbb{P}_{(\mathbf{x}, \mathbf{u})})$, and $\mathcal{A}' = \text{supp}(\bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})})$ means there is a continuous function $B^*(\mathbf{x}, \mathbf{u})$ that satisfies $B^*(\mathbf{x}, \mathbf{u}) = 1$ for all $(\mathbf{x}, \mathbf{u}) \in \text{supp}(\mathbb{P}_{(\mathbf{x}, \mathbf{u})})$ and $B^*(\mathbf{x}, \mathbf{u}) = 0$ for all $(\mathbf{x}, \mathbf{u}) \in \text{supp} \bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})}$. To prove that $B^*(\mathbf{x}, \mathbf{u})$ is a maximum, note that every $B \in \mathcal{B}$ satisfies $B(\mathbf{x}, \mathbf{u}) \in [0, 1]$, meaning that 0 is an upper bound on the optimal value. Substituting $B^*(\mathbf{x}, \mathbf{u})$ into the objective of $\mathbf{S}\text{-FCP}(\mathbb{P}_{(\mathbf{x}, \mathbf{u})}, \bar{\mathbb{P}}_{(\mathbf{x}, \mathbf{u})})$ achieves this value.

To prove Statement 2, consider a fixed $\mathbf{u}$. First note that $B^*(\mathbf{x}, \mathbf{u}) = 1$ for all $\mathbf{x} \in \mathcal{X}(\mathbf{u})$. Because $B^*(\mathbf{x}, \mathbf{u})$ is continuous, we must have $\mathcal{X}(\mathbf{u}) \subset \{\mathbf{x} \mid B^*(\mathbf{x}, \mathbf{u}) > 0\}$. Then, note that $B^{\mathcal{P}}(\mathbf{x}) = 0$ for all $\mathbf{x} \in \mathbb{R}^n \setminus \mathcal{P}$. Consequently,

$$\{\mathbf{x} \mid B^*(\mathbf{x}, \mathbf{u})B^{\mathcal{P}}(\mathbf{x}) > 0\} \subseteq \{\mathbf{x} \mid B^{\mathcal{P}}(\mathbf{x}) > 0\} = \mathcal{P} \subseteq \mathcal{N}_{d_H(\mathcal{X}(\mathbf{u}), \mathcal{P})}(\mathcal{X}(\mathbf{u})).$$

Finally, although $B^*(\mathbf{x}, \mathbf{u}) \in [0, 1]$ rather than in $[0, 1)$ as per the definition of a δ -barrier, this can be rectified by simply scaling the classifier by a multiplicative factor. $\square$

Proof of Theorem 2. Using the same argument as in Theorem 1, as long as $B^*(\mathbf{x}, \mathbf{u})$ is a continuous function, $\mathbf{x}^{\lambda_j}(\hat{\mathbf{u}}_i)$ will exist. For notational simplicity, we use $\mathbf{x}^*$ and $\mathbf{x}^{\lambda_j}$ in place of $\mathbf{x}^*(\hat{\mathbf{u}}_i)$ and $\mathbf{x}^{\lambda_j}(\hat{\mathbf{u}}_i)$, respectively. By the Triangle inequality,

$$|\mathbf{c}^\top F^{(j)}(\hat{\mathbf{u}}_i) - \mathbf{c}^\top \mathbf{x}^*| \leq |\mathbf{c}^\top F^{(j)}(\hat{\mathbf{u}}_i) - \mathbf{c}^\top \mathbf{x}^{\lambda_j}| + |\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \mathbf{c}^\top \mathbf{x}^*|.$$

Even if $B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i)$ is not a δ -barrier for $\mathbf{OP}(\hat{\mathbf{u}}_i)$, we will prove that $\mathbf{x}^\lambda$ is an optimal solution to a barrier problem $\mathbf{BP}(\hat{\mathbf{u}}_i, B_\delta, \lambda_j)$ for some δ -barrier, and therefore, satisfies a (δ, ϵ) -optimality bound $|\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \mathbf{c}^\top \mathbf{x}^*| < \max(\delta, \epsilon)$ for some $\delta > 0$ and $\epsilon > 0$. We split the proof into two separate cases: when $B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) > 0$ and when $B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) = 0$.

First, if $B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) > 0$, we construct a new constrained problem for which $\mathbf{x}^*$ is an optimal solution and show that $B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i)$ is a δ -barrier for the new problem. Let $0 <$

$\varepsilon \leq B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i)$ be any sufficiently small parameter value and consider the optimization problem with a smaller feasible set than $\mathcal{X}(\hat{\mathbf{u}}_i)$, below:

$$\min \left\{ \mathbf{c}^\top \mathbf{x} \mid \mathbf{x} \in \mathcal{X}(\hat{\mathbf{u}}_i), B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) \geq \varepsilon \right\} \quad (\text{EC.1})$$

and note that because $\mathbf{x}^*$ is feasible for (EC.1), it is optimal. Further, $\{\mathbf{x} \mid B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) > 0\}$ is a superset of the feasible set of (EC.1), meaning that $B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i)$ is a δ -barrier for the above problem for some $\delta > 0$. Then, from Theorem 1, $\mathbf{x}^{\lambda_j}$, which is an optimal solution to $\mathbf{BP}(\hat{\mathbf{u}}_i, B^{*,\mathcal{P}}, \lambda_j)$, is (δ, ϵ) -optimal for (EC.1) with $\epsilon = -\lambda \log B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) > 0$.

Second, if $B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) = 0$, then $\mathbf{x}^*$ does not lie in the support of the classifier. Instead, we construct a “test” δ -barrier $B^{\text{test}}(\mathbf{x})$ for $\mathcal{X}(\hat{\mathbf{u}}_i)$ and show that $\mathbf{x}^{\lambda_j}$ is also an optimal solution to $\mathbf{BP}(\hat{\mathbf{u}}_i, B^{\text{test}}, \lambda_j)$, meaning that it satisfies an alternative (δ, ϵ) -optimality bound. Let $\mathbf{x}^{\mathcal{P}} \in \arg \min_{\mathbf{x}} \{\mathbf{c}^\top \mathbf{x} \mid \mathbf{x} \in \mathcal{P}\}$ and let $\bar{B}$ be a constant defined as follows:

$$\bar{B} = B^{*,\mathcal{P}}(\mathbf{x}^{\lambda_j}, \hat{\mathbf{u}}_i) \min \left\{ 1, \exp \left[-\frac{1}{\lambda_j} (\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \mathbf{c}^\top \mathbf{x}^{\mathcal{P}}) \right] \right\}. \quad (\text{EC.2})$$

Note that $\bar{B} \in (0, 1)$. We now define $B^{\text{test}}(\mathbf{x})$ as a continuous function that satisfies:

$$B^{\text{test}}(\mathbf{x}) = \begin{cases} B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) & \forall \mathbf{x} \in \{\mathbf{x} \mid B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) \geq \bar{B}\} \\ \bar{B} & \forall \mathbf{x} \in \{\mathbf{x} \mid B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) < \bar{B}, \mathbf{x} \in \mathcal{X}(\hat{\mathbf{u}}_i)\} \\ \leq \bar{B} & \forall \mathbf{x} \in \{\mathbf{x} \mid B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) < \bar{B}, \mathbf{x} \in \mathcal{P} \setminus \mathcal{X}(\hat{\mathbf{u}}_i)\} \\ 0 & \forall \mathbf{x} \in \mathbb{R}^n \setminus \mathcal{P}. \end{cases}$$

The third condition is left as an inequality since we only need a continuous function $B^{\text{test}}(\mathbf{x})$ to satisfy an inequality in that range. We argue that such a function must exist. Because $B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i)$ is continuous in $\mathbf{x}$, the first region $\{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) \geq \bar{B}\}$ is closed. Furthermore because the support of $B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i)$ is a subset of $\mathcal{P}$, the first region is disjoint from $\mathbb{R}^n \setminus \mathcal{P}$. By Urysohn’s Smooth Lemma, a continuous function within $[0, 1]$ that is equal to 1 for $\mathbf{x} \in \{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) \geq \bar{B}\}$ and 0 for $\mathbf{x} \in \mathbb{R}^n \setminus \mathcal{P}$ exists; scaling this function gives $B^{\text{test}}(\mathbf{x})$.

The support of $B^{\text{test}}(\mathbf{x})$ is a superset of $\mathcal{X}(\hat{\mathbf{u}}_i)$ and $B^{\text{test}}(\mathbf{x})$ is in the range $[0, 1]$, meaning that it is a δ -barrier for $\mathbf{OP}(\hat{\mathbf{u}}_i)$ for some $\delta > 0$. It only remains to prove that $\mathbf{x}^{\lambda_j}$ is an optimal solution for $\mathbf{BP}(\hat{\mathbf{u}}_i, B^{\text{test}}, \lambda_j)$, i.e.,

$$\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \lambda_j \log B^{\text{test}}(\mathbf{x}^{\lambda_j}) \leq \mathbf{c}^\top \mathbf{x} - \lambda_j \log B^{\text{test}}(\mathbf{x}), \quad \forall \mathbf{x} \in \{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) > 0\}.$$

First, consider the region $\{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) \geq \bar{B}\}$. From (EC.2), $\bar{B} \leq B^{*,\mathcal{P}}(\mathbf{x}^{\lambda_j}, \hat{\mathbf{u}}_i)$ meaning $B^{\text{test}}(\mathbf{x}^{\lambda_j}) = B^{*,\mathcal{P}}(\mathbf{x}^{\lambda_j}, \hat{\mathbf{u}}_i)$. Because $\mathbf{x}^{\lambda_j}$ is optimal for $\mathbf{BP}(\hat{\mathbf{u}}_i, B^{*,\mathcal{P}}, \lambda_j)$ and this region is a subset of the support of $B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i)$, we have

$$\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \lambda_j \log B^{\text{test}}(\mathbf{x}^{\lambda_j}) \leq \mathbf{c}^\top \mathbf{x} - \lambda_j \log B^{\text{test}}(\mathbf{x}), \quad \forall \mathbf{x} \in \{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) \geq \bar{B}\}.$$

Now, consider the region $\{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) < \bar{B}\}$. Replacing the minimum in (EC.2) with an inequality and taking the logarithm on both sides yields

$$-\log \bar{B} \geq -\log B^{*,\mathcal{P}}(\mathbf{x}^{\lambda_j}, \hat{\mathbf{u}}_i) + \frac{1}{\lambda_j} (\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \mathbf{c}^\top \mathbf{x}^{\mathcal{P}}).$$

We further re-arrange this inequality to

$$\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \lambda_j \log B^{*,\mathcal{P}}(\mathbf{x}^{\lambda_j}, \hat{\mathbf{u}}_i) \leq \mathbf{c}^\top \mathbf{x}^{\mathcal{P}} - \lambda_j \log \bar{B} \quad (\text{EC.3})$$

$$\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \lambda_j \log B^{\text{test}}(\mathbf{x}^{\lambda_j}) \leq \mathbf{c}^\top \mathbf{x} - \lambda_j \log \bar{B} \quad \forall \mathbf{x} \in \{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) < \bar{B}\} \quad (\text{EC.4})$$

$$\mathbf{c}^\top \mathbf{x}^{\lambda_j} - \lambda_j \log B^{\text{test}}(\mathbf{x}^{\lambda_j}) \leq \mathbf{c}^\top \mathbf{x} - \lambda_j \log B^{\text{test}}(\mathbf{x}) \quad \forall \mathbf{x} \in \{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) < \bar{B}\} \quad (\text{EC.5})$$

We obtain (EC.4) by substituting $B^{\text{test}}(\mathbf{x}^{\lambda_j}) = B^{*,\mathcal{P}}(\mathbf{x}^{\lambda_j}, \hat{\mathbf{u}}_i)$ and noting $\mathbf{c}^\top \mathbf{x}^{\mathcal{P}} \leq \mathbf{c}^\top \mathbf{x}$ for all $\mathbf{x} \in \mathcal{P}$. We obtain (EC.5) because $B^{\text{test}}(\mathbf{x}) < \bar{B}$. Thus, $\mathbf{x}^{\lambda_j}$ is an optimal solution to $\mathbf{BP}(\hat{\mathbf{u}}_i, B^{\text{test}}, \lambda)$ and is (δ, ϵ) -optimal for $\mathbf{OP}(\hat{\mathbf{u}}_i)$ where $\epsilon = -\lambda_j \log \bar{B}$. $\square$

Proof of Corollary 1. The δ term in the bound of (8) is derived from the fact that an optimal solution $\mathbf{x}^{\lambda_j}(\hat{\mathbf{u}}_i)$ to $\mathbf{BP}(\hat{\mathbf{u}}_i, B^{*,\mathcal{P}}, \lambda_j)$ is also an optimal solution to some barrier problem with a δ -barrier. The proof of Theorem 2 invokes two separate cases with two different bounds, i.e., δ takes a different value if $B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) > 0$ or if $B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) = 0$. We consider each case separately and show that δ can be bounded by the above in both cases.

If $B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) > 0$, then the proof of Theorem 2 follows by constructing a new optimization problem (EC.1) with a smaller feasible set and showing that the product classifier is a δ -barrier for that problem, where δ is equal to the Hausdorff distance between this feasible set and the support of the product classifier

$$\delta = d_H(\{\mathbf{x} \mid \mathbf{x} \in \mathcal{X}(\hat{\mathbf{u}}_i), B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) \geq \varepsilon\}, \{\mathbf{x} \mid B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) > 0\}).$$

Because we can choose any sufficiently small value for the ε parameter, we select a value such that $B^{*,\mathcal{P}}(\hat{\mathbf{x}}, \hat{\mathbf{u}}_i) \geq \varepsilon$ for all $(\hat{\mathbf{x}}, \hat{\mathbf{u}}_i) \in \mathcal{D}$. Furthermore, all $\hat{\mathbf{x}}$ in this data set are feasible decisions to $\mathbf{OP}(\hat{\mathbf{u}}_i)$. Thus, we have $\{\mathbf{x} \mid \mathbf{x} \in \mathcal{X}(\hat{\mathbf{u}}_i), B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) \geq \varepsilon\} \supset \{\hat{\mathbf{x}} \mid (\hat{\mathbf{x}}, \hat{\mathbf{u}}_i) \in \mathcal{D}\}$.

Finally, $\{\mathbf{x} \mid B^{*,\mathcal{P}}(\mathbf{x}, \hat{\mathbf{u}}_i) > 0\} \subset \mathcal{P}$. Substituting the subset and superset into the definition of the Hausdorff distance yields the upper bound on δ for this case.

If $B^{*,\mathcal{P}}(\mathbf{x}^*, \hat{\mathbf{u}}_i) = 0$, then the proof of Theorem 2 follows by constructing a new barrier function $B^{\text{test}}(\mathbf{x})$ for $\mathbf{OP}(\hat{\mathbf{u}}_i)$. Then, $\delta = d_H(\mathcal{X}(\hat{\mathbf{u}}_i), \{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) > 0\})$. However, again note that $\mathcal{X}(\hat{\mathbf{u}}_i) \supset \{\hat{\mathbf{x}} \mid (\hat{\mathbf{x}}, \hat{\mathbf{u}}_i) \in \mathcal{D}\}$ and that $\{\mathbf{x} \mid B^{\text{test}}(\mathbf{x}) > 0\} \subseteq \mathcal{P}$. Substituting these two sets into the definition of the Hausdorff distance yields the same upper bound for this case. $\square$

Proof of Proposition 1. When $\mathcal{B}$ is unrestricted, Lemma 1 states that for any k , the optimal value of $\mathbf{FCP}(\mathcal{D}^{(k)}, \bar{\mathcal{D}}^{(k)})$ is 0 and can be achieved when the optimal solution satisfies $B^{(k)}(\hat{\mathbf{x}}, \hat{\mathbf{u}}) = 1$ for all $(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \in \mathcal{D}^{(k)}$ and $B^{(k)}(\bar{\mathbf{x}}, \bar{\mathbf{u}}) = 0$ for all $(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \in \bar{\mathcal{D}}^{(k)}$.

In the $(k+1)^{\text{st}}$ iteration, $\mathcal{D}^{(k+1)} = \mathcal{D}^{(k)} \cup \mathcal{Q}$ and $\bar{\mathcal{D}}^{(k+1)} = \bar{\mathcal{D}}^{(k)} \cup \bar{\mathcal{Q}}$ where $\mathcal{Q}$ and $\bar{\mathcal{Q}}$ are defined as in Algorithm 1. To show that $\mathcal{B}^{(k+1)} \subset \mathcal{B}^{(k)}$, we first prove $\mathcal{B}^{(k+1)} \subseteq \mathcal{B}^{(k)}$ and then present a counter-example which disproves the equivalence.

The objective function of $\mathbf{FCP}(\mathcal{D}^{(k+1)}, \bar{\mathcal{D}}^{(k+1)})$ is

$$\begin{aligned} & \frac{1}{|\mathcal{D}^{(k+1)}|} \sum_{(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \in \mathcal{D}^{(k+1)}} \log B(\hat{\mathbf{x}}, \hat{\mathbf{u}}) + \frac{1}{|\bar{\mathcal{D}}^{(k+1)}|} \sum_{(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \in \bar{\mathcal{D}}^{(k+1)}} \log(1 - B(\bar{\mathbf{x}}, \bar{\mathbf{u}})) \\ &= \frac{\alpha}{|\mathcal{D}^{(k)}|} \sum_{(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \in \mathcal{D}^{(k)}} \log B(\hat{\mathbf{x}}, \hat{\mathbf{u}}) + \frac{1-\alpha}{|\mathcal{Q}|} \sum_{(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \in \mathcal{Q}} \log B(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \\ & \quad + \frac{\alpha'}{|\bar{\mathcal{D}}^{(k)}|} \sum_{(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \in \bar{\mathcal{D}}^{(k)}} \log(1 - B(\bar{\mathbf{x}}, \bar{\mathbf{u}})) + \frac{1-\alpha'}{|\bar{\mathcal{Q}}|} \sum_{(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \in \bar{\mathcal{Q}}} \log(1 - B(\bar{\mathbf{x}}, \bar{\mathbf{u}})), \end{aligned}$$

where $\alpha = |\mathcal{D}^{(k)}|/|\mathcal{D}^{(k+1)}|$ and $\alpha' = |\bar{\mathcal{D}}^{(k)}|/|\bar{\mathcal{D}}^{(k+1)}|$ are the mixture weights defining the ratio of existing to new points in each data set. Because the optimal value of $\mathbf{FCP}(\mathcal{D}^{(k+1)}, \bar{\mathcal{D}}^{(k+1)})$ is 0 and $B(\mathbf{x}, \mathbf{u}) \in [0, 1]$, each of the individual terms must be equal to 0 for an optimal solution. However, the first and third terms define the objective function for $\mathbf{FCP}(\mathcal{D}^{(k)}, \bar{\mathcal{D}}^{(k)})$. Thus, any optimal solution $B^{(k+1)}$ to $\mathbf{FCP}(\mathcal{D}^{(k+1)}, \bar{\mathcal{D}}^{(k+1)})$ must also be optimal for $\mathbf{FCP}(\mathcal{D}^{(k)}, \bar{\mathcal{D}}^{(k)})$ implying $\mathcal{B}^{(k+1)} \subseteq \mathcal{B}^{(k)}$.

To prove the inclusion is strict, consider the sets $\mathcal{D}^{(k)} \cup \bar{\mathcal{Q}}$ and $\bar{\mathcal{D}}^{(k)}$. We can define a function $B^*(\mathbf{x}, \mathbf{u})$ such that $B^*(\hat{\mathbf{x}}, \hat{\mathbf{u}}) = 1$ for all $(\hat{\mathbf{x}}, \hat{\mathbf{u}}) \in \mathcal{D}^{(k)} \cup \bar{\mathcal{Q}}$ and $B^*(\bar{\mathbf{x}}, \bar{\mathbf{u}}) = 0$ for all $(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \in \bar{\mathcal{D}}^{(k)}$, i.e., $B^* \in \mathcal{B}^{(k)}$. However, then $B^*(\bar{\mathbf{x}}, \bar{\mathbf{u}}) = 1$ for all $(\bar{\mathbf{x}}, \bar{\mathbf{u}}) \in \bar{\mathcal{Q}}$ and $B^*(\mathbf{x}, \mathbf{u})$ has an infinite objective function value for $\mathbf{FCP}(\mathcal{D}^{(k+1)}, \bar{\mathcal{D}}^{(k+1)})$. Thus, $B^* \notin \mathcal{B}^{(k+1)}$. $\square$

EC.2. Structural properties of (δ, ϵ) -optimality for the barrier problem

Our learning problem simultaneously trains a classifier and a generative model to learn feasibility and predictive optimal solutions respectively. Alternatively, if we are already

given a δ -barrier $B_\delta(\mathbf{x}, \mathbf{u})$, we may consider directly optimizing $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$. We show how tuning the λ parameter can yield feasible or infeasible solutions of different qualities.

Under a mild regularity assumption, for a sufficiently large λ , an optimal solution $\mathbf{x}^\lambda(\mathbf{u})$ to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$ is guaranteed to lie inside $\mathcal{X}(\mathbf{u})$. Once λ is sufficiently small, the optimal solutions then enter $\mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$. We first state this assumption before characterizing the trajectory of the sequence of points obtained via an IPM.

ASSUMPTION EC.1 (Regularity of the δ -barrier).

1. *There exist $\tilde{\mathbf{x}} \in \text{int}(\mathcal{X}(\mathbf{u}))$ such that $B_\delta(\tilde{\mathbf{x}}, \mathbf{u}) > B_\delta(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \text{cl}(\mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u}))$.*
2. *There exist $\tilde{\mathbf{x}}' \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$ such that $\mathbf{c}^\top \tilde{\mathbf{x}}' < \mathbf{c}^\top \mathbf{x}^*(\mathbf{u})$ and $0 < B_\delta(\tilde{\mathbf{x}}', \mathbf{u}) < B_\delta(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \mathcal{X}(\mathbf{u})$.*

The first statement implies that there exists a point inside $\mathcal{X}(\mathbf{u})$ for which $B_\delta(\mathbf{x}, \mathbf{u})$ is greater than any point outside of $\mathcal{X}(\mathbf{u})$. Similarly, the second statement implies that there exists a point outside of $\mathcal{X}(\mathbf{u})$ for which $B_\delta(\mathbf{x}, \mathbf{u})$ is lower than any point inside $\mathcal{X}(\mathbf{u})$. Intuitively, the barrier yields higher values for points inside $\mathcal{X}(\mathbf{u})$ rather than outside. Furthermore, the existence of $\tilde{\mathbf{x}}'$ for which $\mathbf{c}^\top \tilde{\mathbf{x}} > \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) > \mathbf{c}^\top \tilde{\mathbf{x}}'$ is a direct consequence of the linear objective. Figure EC.1 shows an example of such points for a feasible set where the δ -barrier is a canonical barrier for $\mathcal{P}$. Given a barrier function satisfying Assumption EC.1, λ controls the feasibility of $\mathbf{x}^\lambda(\mathbf{u})$ for $\mathbf{OP}(\mathbf{u})$.

LEMMA EC.1. *If Assumption EC.1 is satisfied, then there exists $\tilde{\lambda}$ such that for all $\lambda \geq \tilde{\lambda}$, the optimal solution to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$ is feasible for $\mathbf{OP}(\mathbf{u})$, i.e., $\mathbf{x}^\lambda(\mathbf{u}) \in \mathcal{X}(\mathbf{u})$.*

Proof of Lemma EC.1. Let $\mathbf{x}^+ \in \arg \sup_{\mathbf{x}} \{B_\delta(\mathbf{x}, \mathbf{u}) \mid \mathbf{x} \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})\}$ and $\mathbf{x}^- \in \arg \inf_{\mathbf{x}} \{\mathbf{c}^\top \mathbf{x} \mid B_\delta(\mathbf{x}, \mathbf{u}) > 0\}$. Then, for $\tilde{\mathbf{x}}$ satisfying Assumption EC.1 Statement 1, we set

$$\tilde{\lambda} = \frac{\mathbf{c}^\top \tilde{\mathbf{x}} - \mathbf{c}^\top \mathbf{x}^-}{\log B_\delta(\tilde{\mathbf{x}}, \mathbf{u}) - \log B_\delta(\mathbf{x}^+, \mathbf{u})}. \quad (\text{EC.6})$$

From the optimality of $\mathbf{x}^-$, we have $\mathbf{c}^\top \tilde{\mathbf{x}} > \mathbf{c}^\top \mathbf{x}^-$. Also, Assumption EC.1 implies that the denominator is positive, and therefore $\tilde{\lambda} > 0$. Rearranging (EC.6) yields

$$\mathbf{c}^\top \tilde{\mathbf{x}} - \tilde{\lambda} \log B_\delta(\tilde{\mathbf{x}}, \mathbf{u}) = \mathbf{c}^\top \mathbf{x}^- - \tilde{\lambda} \log B_\delta(\mathbf{x}^+, \mathbf{u}).$$

By optimality of $\mathbf{x}^+$ and $\mathbf{x}^-$, we have $\mathbf{c}^\top \mathbf{x} \geq \mathbf{c}^\top \mathbf{x}^-$ and $\log B_\delta(\mathbf{x}, \mathbf{u}) \leq \log B_\delta(\mathbf{x}^+, \mathbf{u})$ respectively, for all $\mathbf{x} \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$. Therefore, $\mathbf{c}^\top \tilde{\mathbf{x}} - \tilde{\lambda} \log B_\delta(\tilde{\mathbf{x}}, \mathbf{u}) \leq \mathbf{c}^\top \mathbf{x} - \tilde{\lambda} \log B_\delta(\mathbf{x}, \mathbf{u})$

for all $\mathbf{x} \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$, concluding that the optimal solution to $\mathbf{BP}(\mathbf{u}, B_\delta, \tilde{\lambda})$ must satisfy $\mathbf{x}^{\tilde{\lambda}}(\mathbf{u}) \in \mathcal{X}(\mathbf{u})$.

Now for any $\varepsilon > 0$, observe that

$$\begin{aligned} \mathbf{c}^\top \tilde{\mathbf{x}} - (\tilde{\lambda} + \varepsilon) \log B_\delta(\tilde{\mathbf{x}}, \mathbf{u}) &\leq \mathbf{c}^\top \mathbf{x} - \tilde{\lambda} \log B_\delta(\mathbf{x}, \mathbf{u}) - \varepsilon \log B_\delta(\tilde{\mathbf{x}}, \mathbf{u}), & \forall \mathbf{x} \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u}) \\ &< \mathbf{c}^\top \mathbf{x} - \tilde{\lambda} \log B_\delta(\mathbf{x}, \mathbf{u}) - \varepsilon \log B_\delta(\mathbf{x}, \mathbf{u}), & \forall \mathbf{x} \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u}). \end{aligned}$$

The first line is obtained by adding $\varepsilon \log B_\delta(\tilde{\mathbf{x}}, \mathbf{u})$ to both sides, and the second from $B_\delta(\tilde{\mathbf{x}}, \mathbf{u}) > B_\delta(\mathbf{x}, \mathbf{u})$ for $\mathbf{x} \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$. Thus, $\mathbf{BP}(\mathbf{u}, B_\delta, \tilde{\lambda} + \varepsilon)$ yields feasible solutions to $\mathbf{OP}(\mathbf{u})$. $\square$

LEMMA EC.2. *If Assumption EC.1 is satisfied, then there exists $\tilde{\lambda}'$ such that for all $\lambda \leq \tilde{\lambda}'$, the optimal solution to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda)$ is infeasible for $\mathbf{OP}(\mathbf{u})$, i.e., $\mathbf{x}^\lambda(\mathbf{u}) \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$.*

Proof of Lemma EC.2. Let $\mathbf{x}^\dagger \in \arg \max_{\mathbf{x}} \{B_\delta(\mathbf{x}, \mathbf{u}) \mid \mathbf{x} \in \mathcal{X}(\mathbf{u})\}$. Then, for $\tilde{\mathbf{x}}'$ satisfying Assumption EC.1 Statement 2, let

$$\tilde{\lambda}' = \frac{\mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) - \mathbf{c}^\top \tilde{\mathbf{x}}'}{\log B_\delta(\mathbf{x}^\dagger, \mathbf{u}) - \log B_\delta(\tilde{\mathbf{x}}', \mathbf{u})}. \quad (\text{EC.7})$$

Assumption EC.1 Statement 2 ensures $\mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) > \mathbf{c}^\top \tilde{\mathbf{x}}'$ and $\log B_\delta(\mathbf{x}^\dagger, \mathbf{u}) > \log B_\delta(\tilde{\mathbf{x}}', \mathbf{u})$. Therefore, $\tilde{\lambda}' > 0$. Rearranging (EC.7) gives us

$$\mathbf{c}^\top \tilde{\mathbf{x}}' - \tilde{\lambda}' \log B_\delta(\tilde{\mathbf{x}}', \mathbf{u}) = \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) - \tilde{\lambda}' \log B_\delta(\mathbf{x}^\dagger, \mathbf{u}).$$

By optimality of $\mathbf{x}^*(\mathbf{u})$ and $\mathbf{x}^\dagger$, we have $\mathbf{c}^\top \mathbf{x} \geq \mathbf{c}^\top \mathbf{x}^*(\mathbf{u})$ and $\log B_\delta(\mathbf{x}, \mathbf{u}) \leq \log B_\delta(\mathbf{x}^\dagger, \mathbf{u})$ respectively, for all $\mathbf{x} \in \mathcal{X}(\mathbf{u})$. Therefore $\mathbf{c}^\top \tilde{\mathbf{x}}' - \tilde{\lambda}' \log B_\delta(\tilde{\mathbf{x}}', \mathbf{u}) \leq \mathbf{c}^\top \mathbf{x} - \tilde{\lambda}' \log B_\delta(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \mathcal{X}(\mathbf{u})$, concluding that the optimal solution to $\mathbf{BP}(\mathbf{u}, B_\delta, \tilde{\lambda}')$ must satisfy $\mathbf{x}^{\tilde{\lambda}'}(\mathbf{u}) \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$.

Now for any $\varepsilon > 0$, observe that

$$\begin{aligned} \mathbf{c}^\top \tilde{\mathbf{x}}' - (\tilde{\lambda}' - \varepsilon) \log B_\delta(\tilde{\mathbf{x}}', \mathbf{u}) &\leq \mathbf{c}^\top \mathbf{x} - \tilde{\lambda}' \log B_\delta(\mathbf{x}, \mathbf{u}) + \varepsilon \log B_\delta(\tilde{\mathbf{x}}', \mathbf{u}), & \forall \mathbf{x} \in \mathcal{X}(\mathbf{u}) \\ &< \mathbf{c}^\top \mathbf{x} - \tilde{\lambda}' \log B_\delta(\mathbf{x}, \mathbf{u}) + \varepsilon \log B_\delta(\mathbf{x}, \mathbf{u}), & \forall \mathbf{x} \in \mathcal{X}(\mathbf{u}). \end{aligned}$$

The first line is obtained by subtracting $\varepsilon \log B_\delta(\tilde{\mathbf{x}}', \mathbf{u})$ to both sides, and the second from $B_\delta(\tilde{\mathbf{x}}', \mathbf{u}) < B_\delta(\mathbf{x}, \mathbf{u})$ for all $\mathbf{x} \in \mathcal{X}(\mathbf{u})$. Thus, $\mathbf{BP}(\mathbf{u}, B_\delta, \tilde{\lambda}' - \varepsilon)$ yields infeasible solutions to $\mathbf{OP}(\mathbf{u})$. $\square$

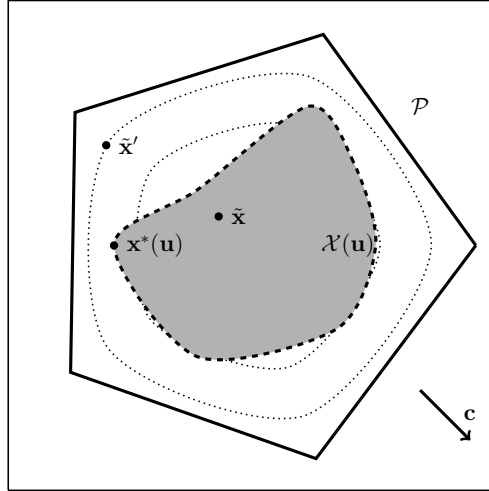


Figure EC.1 The canonical barrier $B^{\mathcal{P}}(\mathbf{x})$ where the dotted lines are level sets. $\mathbf{x}^*(\mathbf{u})$ is optimal for $\mathbf{OP}(\mathbf{u})$ while $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ satisfy Lemmas EC.1 and EC.2 respectively.

Lemma EC.2 and Assumption EC.1 explore the case where the barrier problem produces undesirable results. Otherwise, if $\mathbf{c}^\top \tilde{\mathbf{x}}' > \mathbf{c}^\top \mathbf{x}^*(\mathbf{u})$ and $B_\delta(\tilde{\mathbf{x}}', \mathbf{u}) \geq B_\delta(\mathbf{x}, \mathbf{u})$ for all $\tilde{\mathbf{x}}' \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$ and $\mathbf{x} \in \mathcal{X}(\mathbf{u})$, $\mathbf{OP}(\mathbf{u})$ could be solved by classical IPMs.

Lemmas EC.1 and EC.2 state that when λ is set sufficiently high (or low), the corresponding optimal solution $\mathbf{x}^\lambda(\mathbf{u})$ is a certifiably feasible (or infeasible) solution to $\mathbf{OP}(\mathbf{u})$. Furthermore, there exists a trajectory, i.e., feasibility (or infeasibility) is guaranteed for all λ sufficiently high (or low). Assuming access to an oracle $\Psi(\mathbf{x}, \mathbf{u})$, we can construct a simple IPM (see Algorithm 2) to obtain optimal solutions to $\mathbf{OP}(\mathbf{u})$. We initialize with a large λ_0 that satisfies Lemma EC.1. We define a decay rate $\nu < 1$ and a number of iterations $j \in 0, \dots, J$. Then, for each j , we simply let $\lambda_j = \lambda_0 \nu^j$ and solve $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda_j)$ to obtain a new (δ, ϵ) -optimal solution in each iteration. At the end of each iteration, the oracle checks if the solution is still feasible, and terminates when the solution exits the feasible set.

Recall that we always have access to the canonical barrier $B^{\mathcal{P}}(\mathbf{x})$ and therefore, we only consider any δ -barrier where $\delta \leq d_H(\mathcal{X}(\mathbf{u}), \mathcal{P})$. We prove the optimality bound for solutions obtained via Algorithm 2.

PROPOSITION EC.1. *Consider a δ -barrier where $\delta \leq d_H(\mathcal{X}(\mathbf{u}), \mathcal{P})$. Suppose that $\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2 \in \mathcal{X}(\mathbf{u})$ and $\tilde{\mathbf{x}}'_1, \tilde{\mathbf{x}}'_2 \in \mathcal{N}_\delta(\mathcal{X}(\mathbf{u})) \setminus \mathcal{X}(\mathbf{u})$ satisfy Statements 1 and 2 of Assumption EC.1, respectively. Assume without loss of generality $B_\delta(\tilde{\mathbf{x}}_1, \mathbf{u}) > B_\delta(\tilde{\mathbf{x}}_2, \mathbf{u})$ and $\mathbf{c}^\top \tilde{\mathbf{x}}'_1 > \mathbf{c}^\top \tilde{\mathbf{x}}'_2$. Let $\mathbf{x}^{\mathcal{P}} \in \arg \min_{\mathbf{x}} \{f(\mathbf{x}) \mid \mathbf{x} \in \mathcal{P}\}$. For $J > 0$ and $j \in \{0, \dots, J\}$, consider*

$$\lambda_0 = \frac{\mathbf{c}^\top \tilde{\mathbf{x}}_1 - \mathbf{c}^\top \mathbf{x}^{\mathcal{P}}}{\log B_\delta(\tilde{\mathbf{x}}_1, \mathbf{u}) - \log B_\delta(\tilde{\mathbf{x}}_2, \mathbf{u})}, \quad \nu = \left(\frac{\mathbf{c}^\top \tilde{\mathbf{x}}'_1 - \mathbf{c}^\top \tilde{\mathbf{x}}'_2}{-\lambda_0 \log B_\delta(\tilde{\mathbf{x}}'_1, \mathbf{u})} \right)^{1/J}, \quad \lambda_j = \lambda_0 \nu^j$$

Algorithm 2 Interior Point Method with a δ -barrier**Input:** δ -barrier $B_\delta(\mathbf{x}, \mathbf{u})$; Initial dual variable λ_0 and decay rate $\nu < 1$; Oracle $\Psi(\mathbf{x}, \mathbf{u})$.**Output:** Optimal solution $\mathbf{x}^\lambda(\mathbf{u})$ to the barrier problem.

```

1: for  $j = 0$  to  $J$  do
2:   Solve  $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda_j)$  to obtain optimal solution  $\mathbf{x}^{\lambda_j}(\mathbf{u})$ .
3:   if  $\Psi(\mathbf{x}^{\lambda_j}(\mathbf{u}), \mathbf{u}) = 0$  then
4:     return Previous optimal solution  $\mathbf{x}^{\lambda_{j-1}}(\mathbf{u})$ .
5:   end if
6: end for

```

Then, the following statements are true:

1. An optimal solution $\mathbf{x}^{\lambda_0}(\mathbf{u})$ to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda_0)$ is a feasible solution for $\mathbf{OP}(\mathbf{u})$.
2. There exists $1 \leq j^* \leq J$ such that for all $j < j^*$, an optimal solution $\mathbf{x}^{\lambda_j}(\mathbf{u})$ to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda_j)$ is feasible for $\mathbf{OP}(\mathbf{u})$ and for all $j \geq j^*$, $\mathbf{x}^{\lambda_j}(\mathbf{u})$ is infeasible for $\mathbf{OP}(\mathbf{u})$.
3. For any $j < j^*$, an optimal solution $\mathbf{x}^{\lambda_j}(\mathbf{u})$ is $(0, \epsilon_j)$ -optimal for $\mathbf{OP}(\mathbf{u})$ where

$$\epsilon_j = (\mathbf{c}^\top \tilde{\mathbf{x}}'_1 - \mathbf{c}^\top \tilde{\mathbf{x}}'_2) \nu^{j-J}.$$

Further, for any $j \geq j^*$, $\mathbf{x}^{\lambda_j}(\mathbf{u})$ is (δ, ϵ_j) -optimal for $\mathbf{OP}(\mathbf{u})$, where $\delta \leq d_H(\mathcal{X}(\mathbf{u}), \mathcal{P})$.

Proof of Proposition EC.1. We first make several observations about the parameters. Note that because $\mathcal{X}(\mathbf{u}) \subset \mathcal{P}$ relaxes the feasible set, we have $\mathbf{c}^\top \mathbf{x}^\mathcal{P} \leq \mathbf{c}^\top \mathbf{x}^*(\mathbf{u})$. Next for all $j \leq M$, $\lambda_j = \lambda_0 \nu^j$ and specifically $\lambda_J = \lambda_0 \nu^M = -(\mathbf{c}^\top \tilde{\mathbf{x}}'_1 - \mathbf{c}^\top \tilde{\mathbf{x}}'_2) / \log B_\delta(\tilde{\mathbf{x}}'_1, \mathbf{u})$.

To prove the first statement, we show that $\lambda_0 > \tilde{\lambda}$ where $\tilde{\lambda}$ is defined as in (EC.6) and constructed using $\tilde{\mathbf{x}}_1$. Note that $\mathbf{c}^\top \mathbf{x}^\mathcal{P} \leq \mathbf{c}^\top \mathbf{x}^-$ and by Assumption EC.1, $\log B_\delta(\tilde{\mathbf{x}}_2, \mathbf{u}) > \log B_\delta(\mathbf{x}^+, \mathbf{u})$. We substitute $\mathbf{c}^\top \mathbf{x}^\mathcal{P}$ and $\log B_\delta(\tilde{\mathbf{x}}_2, \mathbf{u})$ in λ_0 and prove $\lambda_0 > \tilde{\lambda}$. By Lemma EC.1, Statement 1 must hold.

We use a similar argument using $\tilde{\mathbf{x}}'_1$ to show $\lambda_J < \tilde{\lambda}'$ as defined in (EC.7). By Lemma EC.2, an optimal solution $\mathbf{x}^{\lambda_J}$ must be infeasible for $\mathbf{OP}(\mathbf{u})$. Given that λ_j decreases every iteration and using the first statement, there must exist a cutoff point $1 \leq j^* \leq M$ for which $\lambda_{j^*} < \tilde{\lambda}'$ and $\lambda_{j^*-1} \geq \tilde{\lambda}'$. Therefore, Statement 2 must also hold.

In order to prove the third statement, recall that $\delta \leq d_H(\mathcal{X}(\hat{\mathbf{u}}), \mathcal{P})$ for all j . We first prove $(\Delta(\mathbf{u}), \epsilon_j)$ -optimality when $j = J$, and then prove for $j < J$. Let $\epsilon_J = \mathbf{c}^\top \tilde{\mathbf{x}}'_1 - \mathbf{c}^\top \tilde{\mathbf{x}}'_2$. Note that

$$\lambda_J = \frac{\mathbf{c}^\top \tilde{\mathbf{x}}'_1 - \mathbf{c}^\top \tilde{\mathbf{x}}'_2}{-\log B_\delta(\tilde{\mathbf{x}}'_1, \mathbf{u})} = \frac{\epsilon_J}{-\log B_\delta(\tilde{\mathbf{x}}'_1, \mathbf{u})} < \frac{\epsilon_J}{-\log B_\delta(\mathbf{x}^*, \mathbf{u})}.$$

The second equality follows from substituting the value of ϵ_M and the inequality from $B_\delta(\tilde{\mathbf{x}}'_1, \mathbf{u}) < B_\delta(\mathbf{x}^*(\mathbf{u}), \mathbf{u})$ (i.e., Assumption EC.1). We next show that $\mathbf{x}^{\lambda_J}$ satisfies $(\Delta(\mathbf{u}), \epsilon_J)$ -optimality,

$$\begin{aligned} \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) + \epsilon_J &> \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) - \lambda_J \log B_\delta(\mathbf{x}^*(\mathbf{u}), \mathbf{u}) \\ &\geq \mathbf{c}^\top \mathbf{x}^{\lambda_J}(\mathbf{u}) - \lambda_J \log B_\delta(\mathbf{x}^{\lambda_J}(\mathbf{u}), \mathbf{u}) \\ &> \mathbf{c}^\top \mathbf{x}^{\lambda_J}(\mathbf{u}). \end{aligned}$$

The first line follows from substituting the value of ϵ_J and the second from the optimality of $\mathbf{x}^{\lambda_J}(\mathbf{u})$ for $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda_J)$. The third line follows from $-\lambda_J \log B_\delta(\mathbf{x}^{\lambda_J}(\mathbf{u}), \mathbf{u}) > 0$.

For each $j < M$, we have $\lambda_j = \lambda_J \nu^{j-M}$. Then, we write $\epsilon_j = (\mathbf{c}^\top \tilde{\mathbf{x}}'_1 - \mathbf{c}^\top \tilde{\mathbf{x}}'_2) \nu^{j-M}$. The same steps used for the $j = M$ case are repeated to obtain $(\Delta(\mathbf{u}), \epsilon_j)$ -optimality certificates. Finally, note that from Statement 2, for all $j < j^*$, the optimal solutions $\mathbf{x}^{\lambda_j}(\mathbf{u})$ are feasible for $\mathbf{OP}(\mathbf{u})$. By optimality of $\mathbf{x}^*(\mathbf{u})$ for $\mathbf{OP}(\mathbf{u})$, we have $\delta = 0$ for all $j < j^*$. $\square$

Proposition EC.1 first provides parameters $\lambda_0 > \tilde{\lambda}$ and $\lambda_J < \tilde{\lambda}'$ for which the optimal solutions to $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda_0)$ and $\mathbf{BP}(\mathbf{u}, B_\delta, \lambda_J)$ lie inside and outside of $\mathcal{X}(\mathbf{u})$, respectively. Next, it shows that the sequence of λ_j produces a sequence of optimal solutions $\{\mathbf{x}^{\lambda_j}(\mathbf{u})\}$ that start within the feasible set $\mathcal{X}(\mathbf{u})$ and proceed to move outside. Finally, it derives a sequence of corresponding $\{\epsilon_j\}$ such that the sequence of solutions are $(\Delta(\mathbf{u}), \epsilon_j)$ -optimal for $\mathbf{OP}(\mathbf{u})$. This implies the final solution is $(0, (\mathbf{c}^\top \tilde{\mathbf{x}}'_1 - \mathbf{c}^\top \tilde{\mathbf{x}}'_2) \nu^{j^*-1-J})$ -optimal for $\mathbf{OP}(\mathbf{u})$.

The above proposition summarizes an IPM for solving $\mathbf{OP}(\mathbf{u})$ when given a δ -barrier $B_\delta(\mathbf{x}, \mathbf{u})$ at least as good as the canonical barrier and an oracle $\Psi(\mathbf{x}, \mathbf{u})$. The IPM behaves predictably; by initializing with large λ , we ensure that we obtain feasible solutions, but by decreasing λ , we know that the solution will ultimately be infeasible. An oracle could identify the point of termination immediately before the IPM leaves the feasible set. We can from here obtain a tight bound on the (δ, ϵ) -optimality of the final solution.

While direct optimization is desirable for its structural properties, this IPM approach is reliant on access to a δ -barrier. On the other hand, IPMAN learns a classifier that approximates a δ -barrier after several iterations. Therefore, unless we are given an a priori δ -barrier (e.g., a canonical barrier for $\mathcal{P}$), this IPM approach is not necessarily feasible from the onset. A potential fix would be to first train IPMAN until a δ -barrier is obtained and then use the δ -barrier IPM to solve subsequent problems. This ties to the second

difference between the two approaches; IPMAN is ultimately a predictive model and is therefore subject to prediction error. On the other hand, prediction from a trained model is much faster than direct optimization. Therefore, in cases where the problem is large and an IPM would be difficult to solve or require numerous queries from an oracle, the predictive power of IPMAN yields more practical benefits.

EC.3. Proof of the generalization bound (Theorem 3)

The proof of the generalization bound uses a Generalization Lemma of Bertsimas and Kallus (2019) to bound the (δ, ϵ) -bound in terms of an empirical risk objective function error between $F^*(\mathbf{u})$ versus $\mathbf{x}^\lambda(\mathbf{u})$ and Markov's inequality to translate this bound to a probabilistic (δ, ϵ) -optimality certificate. However, in order to use the lemma in this way, we first require an auxiliary result to relate F^* with $\mathfrak{R}_{N_u}(\mathcal{F})$.

Assumption 2 states that the generative model F^* is a composition; we project the classifier output to $\mathcal{P}$ whenever $F^{(j,k)}(\mathbf{u}) \notin \mathcal{P}$. Although $F^{(j,k)} \in \mathcal{F}$, the final model $F^*(\mathbf{u}) := \text{proj}(F(\mathbf{u})) = \arg \min_{\mathbf{x}} \{\|\mathbf{x} - F(\mathbf{u})\| \mid \mathbf{x} \in \mathcal{P}\}$ may not be a member of $\mathcal{F}$. We first bound the Rademacher complexity of models composed from projection below.

LEMMA EC.3. *Let $\mathcal{F} = \{F : \mathcal{U} \rightarrow \mathbb{R}^n\}$ be a model class and $\text{proj}(\mathcal{F}) = \{\text{proj}(F) \mid F \in \mathcal{F}\}$ be the class of models composed by a projection to a polyhedron $\mathcal{P}$. Then for any $\hat{\mathcal{U}} \sim \mathbb{P}_{\mathbf{u}}$, $\hat{\mathfrak{R}}_{N_u}(\text{proj}(\mathcal{F}), \hat{\mathcal{U}}) \leq \sqrt{2n} \hat{\mathfrak{R}}_{N_u}(\mathcal{F}, \hat{\mathcal{U}})$.*

Proof of Lemma EC.3. We want to show for any fixed $\hat{\mathcal{U}}$ that

$$\mathbb{E}_{\boldsymbol{\sigma} \sim p_{\boldsymbol{\sigma}}} \left[\frac{2}{N_u} \sup_{F \in \mathcal{F}} \sum_{i=1}^{N_u} \boldsymbol{\sigma}_i^\top \text{proj}(F(\hat{\mathbf{u}}_i)) \right] \leq \sqrt{2n} \mathbb{E}_{\boldsymbol{\sigma} \sim p_{\boldsymbol{\sigma}}} \left[\frac{2}{N_u} \sup_{F \in \mathcal{F}} \sum_{i=1}^{N_u} \boldsymbol{\sigma}_i^\top F(\hat{\mathbf{u}}_i) \right]. \quad (\text{EC.8})$$

By conditioning and iterating, it suffices to prove the following inequality for any function $\Xi(F) : \mathcal{F} \rightarrow \mathbb{R}$,

$$\mathbb{E}_{\boldsymbol{\sigma} \sim p_{\boldsymbol{\sigma}}} \left[\sup_{F \in \mathcal{F}} \boldsymbol{\sigma}^\top \text{proj}(F) + \Xi(F) \right] \leq \mathbb{E}_{\boldsymbol{\sigma} \sim p_{\boldsymbol{\sigma}}} \left[\sup_{F \in \mathcal{F}} \sqrt{2n} \boldsymbol{\sigma}^\top F + \Xi(F) \right]. \quad (\text{EC.9})$$

We first prove inequality (EC.9), before returning to the main lemma.

As $\boldsymbol{\sigma} \sim p_{\boldsymbol{\sigma}}$ is a random vector of i.i.d. Rademacher variables, it is supported over the (ordered) set $\{(-1, \dots, -1, -1), (-1, \dots, -1, 1), \dots, (1, \dots, 1, 1)\}$ all with equal probability.

Let $\hat{\sigma}_\ell$ denote the ℓ -th element of this set. By iterating over all values, we expand the left-hand-side of (EC.9) out to:

$$\mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sigma^\top \text{proj}(F) + \Xi(F) \right] = \frac{1}{2^n} \sum_{\ell=1}^{2^n} \left(\sup_{F \in \mathcal{F}} \hat{\sigma}_\ell^\top \text{proj}(F) + \Xi(F) \right) \quad (\text{EC.10})$$

$$= \frac{1}{2^n} \sum_{\ell=1}^{2^{n-1}} \left(\sup_{F \in \mathcal{F}} \{ \hat{\sigma}_\ell^\top \text{proj}(F) + \Xi(F) \} + \sup_{F \in \mathcal{F}} \{ -\hat{\sigma}_\ell^\top \text{proj}(F) + \Xi(F) \} \right) \quad (\text{EC.11})$$

$$= \frac{1}{2^n} \sum_{\ell=1}^{2^{n-1}} \left(\sup_{F_1, F_2 \in \mathcal{F}} \hat{\sigma}_\ell^\top (\text{proj}(F_1) - \text{proj}(F_2)) + \Xi(F_1) + \Xi(F_2) \right). \quad (\text{EC.12})$$

Equation (EC.10) follows by letting $\hat{\sigma}_\ell$ iterate over the support of the distribution. Equation (EC.11) follows from the symmetry of the Rademacher distribution. That is, for every $\hat{\sigma}_\ell$, there exists $-\hat{\sigma}_\ell$ with equal probability, and we need to only characterize half of the elements in the support. (EC.12) merges the suprema.

By the Obtuse Angle Criterion, projection to a convex set is a non-expansive operation (i.e., $\|\text{proj}(F_1) - \text{proj}(F_2)\| \leq \|F_1 - F_2\|$). We use the Cauchy-Schwarz inequality and the non-expansiveness property (in (EC.13) and (EC.14) below, respectively) to remove the dependency on the projection operator:

$$\text{RHS (EC.12)} \leq \frac{1}{2^n} \sum_{\ell=1}^{2^{n-1}} \left(\sup_{F_1, F_2 \in \mathcal{F}} \|\hat{\sigma}_\ell\| \|\text{proj}(F_1) - \text{proj}(F_2)\| + \Xi(F_1) + \Xi(F_2) \right) \quad (\text{EC.13})$$

$$\leq \frac{1}{2^n} \sum_{\ell=1}^{2^{n-1}} \left(\sup_{F_1, F_2 \in \mathcal{F}} \|\hat{\sigma}_\ell\| \|F_1 - F_2\| + \Xi(F_1) + \Xi(F_2) \right) \quad (\text{EC.14})$$

$$\leq \frac{1}{2^n} \sum_{\ell=1}^{2^{n-1}} \left(\sup_{F_1, F_2 \in \mathcal{F}} \sqrt{n} \|F_1 - F_2\| + \Xi(F_1) + \Xi(F_2) \right) \quad (\text{EC.15})$$

$$\leq \frac{1}{2} \left(\sup_{F_1, F_2 \in \mathcal{F}} \sqrt{n} \|F_1 - F_2\| + \Xi(F_1) + \Xi(F_2) \right) \quad (\text{EC.16})$$

$$\leq \frac{1}{2} (\sqrt{n} \|F_1^* - F_2^*\| + \Xi(F_1^*) + \Xi(F_2^*)). \quad (\text{EC.17})$$

Inequality (EC.15) follows by noting $\|\sigma\| \leq \sqrt{n}$ for all $\sigma \sim p_\sigma$ and (EC.16) from the fact that the dependency on $\hat{\sigma}_\ell$ has been removed. We obtain (EC.17) by letting F_1^* and F_2^* be the two values that attain the supremum.

We use the Khintchine inequality to bound $\|F_1^* - F_2^*\| \leq \sqrt{2}\mathbb{E}_{\sigma \sim p_\sigma} [|\sigma^\top(F_1^* - F_2^*)|]$. We then rearrange the terms as follows:

$$\text{RHS (EC.17)} \leq \frac{1}{2} \left(\sqrt{2n}\mathbb{E}_{\sigma \sim p_\sigma} [|\sigma^\top(F_1^* - F_2^*)|] + \Xi(F_1^*) + \Xi(F_2^*) \right) \quad (\text{EC.18})$$

$$= \frac{1}{2} \left(\mathbb{E}_{\sigma \sim p_\sigma} \left[\sqrt{2n} |\sigma^\top(F_1^* - F_2^*)| + \Xi(F_1^*) + \Xi(F_2^*) \right] \right) \quad (\text{EC.19})$$

$$\leq \frac{1}{2} \left(\mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F_1, F_2 \in \mathcal{F}} \sqrt{2n} |\sigma^\top(F_1 - F_2)| + \Xi(F_1) + \Xi(F_2) \right] \right) \quad (\text{EC.20})$$

$$= \frac{1}{2} \left(\mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \left\{ \sqrt{2n}\sigma^\top F + \Xi(F) \right\} + \sup_{F \in \mathcal{F}} \left\{ -\sqrt{2n}\sigma^\top F + \Xi(F) \right\} \right] \right) \quad (\text{EC.21})$$

$$= \mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sqrt{2n}\sigma^\top F + \Xi(F) \right]. \quad (\text{EC.22})$$

Inequality (EC.19) brings all of the terms inside the expectation. (EC.20) upper bounds by the supremum. Because $\Xi(F_1) + \Xi(F_2)$ is invariant under the exchange of F_1 and F_2 , the supremum will be obtained when $\sigma^\top(F_1 - F_2)$ is positive, meaning we can remove the absolute value and separate the supremum in (EC.21). Finally, the symmetry of the random variable σ implies that the two suprema are equal, thereby giving (EC.22).

To complete the proof, we use a standard conditioning argument (see Maurer (2016)) to show (EC.8) decomposes to (EC.9). For any $0 \leq m \leq N_{\mathbf{u}}$, we prove the following by induction:

$$\mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sum_{i=1}^{N_{\mathbf{u}}} \sigma_i^\top \text{proj}(F(\hat{\mathbf{u}}_i)) \right] \leq \mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sum_{i=1}^m \sqrt{2n}\sigma_i^\top F(\hat{\mathbf{u}}_i) + \sum_{i=m+1}^{N_{\mathbf{u}}} \sigma_i^\top \text{proj}(F(\hat{\mathbf{u}}_i)) \right].$$

The case for $m = 0$ is an identity. Now for fixed values of $\hat{\sigma}_i, \forall i \neq m$, let

$$\Xi(F) = \sum_{i=1}^{m-1} \sqrt{2n}\hat{\sigma}_i^\top F(\hat{\mathbf{u}}_i) + \sum_{i=m+1}^{N_{\mathbf{u}}} \hat{\sigma}_i^\top \text{proj}(F(\hat{\mathbf{u}}_i)).$$

Then, assuming the inequality holds for $m - 1$, we show

$$\begin{aligned} \mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sum_{i=1}^{N_{\mathbf{u}}} \sigma_i^\top \text{proj}(F(\hat{\mathbf{u}}_i)) \right] &\leq \mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sum_{i=1}^{m-1} \sqrt{2n}\sigma_i^\top F(\hat{\mathbf{u}}_i) + \sum_{i=m}^{N_{\mathbf{u}}} \sigma_i^\top \text{proj}(F(\hat{\mathbf{u}}_i)) \right] \\ &= \mathbb{E}_{\sigma \sim p_\sigma} \left[\mathbb{E}_{\sigma_m \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sigma_m^\top \text{proj}(F(\hat{\mathbf{u}}_m)) + \Xi(F) \mid \{\hat{\sigma}_i, \forall i \neq m\} \right] \right] \\ &\leq \mathbb{E}_{\sigma \sim p_\sigma} \left[\mathbb{E}_{\sigma_m \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sqrt{2n}\sigma_m^\top F(\hat{\mathbf{u}}_m) + \Xi(F) \mid \{\hat{\sigma}_i, \forall i \neq m\} \right] \right] \\ &= \mathbb{E}_{\sigma \sim p_\sigma} \left[\sup_{F \in \mathcal{F}} \sum_{i=1}^m \sqrt{2n}\sigma_i^\top F(\hat{\mathbf{u}}_i) + \sum_{i=m+1}^{N_{\mathbf{u}}} \sigma_i^\top \text{proj}(F(\hat{\mathbf{u}}_i)) \right]. \end{aligned}$$

The second inequality comes from substituting (EC.9). When $m = N_{\mathbf{u}}$, the proof is complete. $\square$

Lemma EC.3 can be seen as an extension of the main theorem of Maurer (2016) and is proved using a similar sequence of steps. There, the authors showed that composition of a Lipschitz scalar-valued vector function onto a vector-valued model class bounds the Rademacher complexity of the composed class by $\sqrt{2L}$. In the above, we compose the projection operator, a vector-valued function, to the vector-valued model class and bound the Rademacher complexity by $\sqrt{2n}$. Although we only specifically consider the projection operator, the proof easily extends to any vector-valued function, so long as it is L -Lipschitz, whereupon we would reintroduce L back into the bound.

Before proving Theorem 3, we re-state the Generalization Lemma of Bertsimas and Kallus (2019).

LEMMA EC.4 (Bertsimas and Kallus (2019)). *Consider a function $z(\mathbf{x}, \mathbf{u}) : \mathcal{P} \times \mathcal{U} \rightarrow \mathbb{R}$ that is bounded and L_∞ -Lipschitz continuous in $\mathbf{x}$ using the $\|\cdot\|_\infty$ norm,*

$$\sup_{\mathbf{x} \in \mathcal{P}, \mathbf{u} \in \mathcal{U}} z(\mathbf{x}, \mathbf{u}) \leq K, \quad \sup_{\mathbf{x}_1 \neq \mathbf{x}_2 \in \mathcal{P}, \mathbf{u} \in \mathcal{U}} \frac{z(\mathbf{x}_1, \mathbf{u}) - z(\mathbf{x}_2, \mathbf{u})}{\|\mathbf{x}_1 - \mathbf{x}_2\|_\infty} \leq L_\infty.$$

For any $\beta > 0$, with probability at least $1 - \beta$ with respect to the sampling of $\hat{\mathcal{U}}$,

$$\mathbb{E}_{\mathbf{u} \sim \mathbb{P}_{\mathbf{u}}} \left[z(F(\mathbf{u}), \mathbf{u}) \right] \leq \frac{1}{N_{\mathbf{u}}} \sum_{i=1}^{N_{\mathbf{u}}} z(F(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) + K \sqrt{\frac{\log(1/\beta)}{2N_{\mathbf{u}}}} + L_\infty \mathfrak{R}_{N_{\mathbf{u}}}(\text{proj}(\mathcal{F})), \quad \forall F \in \text{proj}(\mathcal{F}).$$

We are now ready to prove Theorem 3.

Proof of Theorem 3. The proof follows by first applying Lemma EC.4, before applying Markov's inequality. We let $z(\mathbf{x}, \mathbf{u}) = |\mathbf{c}^\top \mathbf{x} - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})|$, as a function of $\mathbf{x} \in \mathcal{P}$ and $\mathbf{u} \in \mathcal{U}$, and show it is bounded from above

$$\sup_{\mathbf{x} \in \mathcal{P}, \mathbf{u} \in \mathcal{U}} z(\mathbf{x}, \mathbf{u}) = \sup_{\mathbf{x} \in \mathcal{P}, \mathbf{u} \in \mathcal{U}} |\mathbf{c}^\top \mathbf{x} - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})| \quad (\text{EC.23})$$

$$\leq \max_{\mathbf{x} \in \mathcal{P}} \mathbf{c}^\top \mathbf{x} - \min_{\mathbf{x} \in \mathcal{P}} \mathbf{c}^\top \mathbf{x} = K. \quad (\text{EC.24})$$

Because $\mathcal{P}$ is compact, (EC.24) is bounded. We set K to be equal to RHS (EC.24).

We next show L_∞ -Lipschitz continuity,

$$\sup_{\mathbf{x}_1 \neq \mathbf{x}_2 \in \mathcal{P}, \mathbf{u} \in \mathcal{U}} \frac{z(\mathbf{x}_1, \mathbf{u}) - z(\mathbf{x}_2, \mathbf{u})}{\|\mathbf{x}_1 - \mathbf{x}_2\|_\infty} = \sup_{\mathbf{x}_1 \neq \mathbf{x}_2 \in \mathcal{P}, \mathbf{u} \in \mathcal{U}} \frac{|\mathbf{c}^\top \mathbf{x}_1 - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})| - |\mathbf{c}^\top \mathbf{x}_2 - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})|}{\|\mathbf{x}_1 - \mathbf{x}_2\|_\infty} \quad (\text{EC.25})$$

$$\leq \sup_{\mathbf{x}_1 \neq \mathbf{x}_2 \in \mathcal{P}, \mathbf{u} \in \mathcal{U}} \frac{|\mathbf{c}^\top \mathbf{x}_1 - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u}) - \mathbf{c}^\top \mathbf{x}_2 + \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})|}{\|\mathbf{x}_1 - \mathbf{x}_2\|_\infty} \quad (\text{EC.26})$$

$$= \sup_{\mathbf{x}_1 \neq \mathbf{x}_2 \in \mathcal{P}} \frac{|\mathbf{c}^\top \mathbf{x}_1 - \mathbf{c}^\top \mathbf{x}_2|}{\|\mathbf{x}_1 - \mathbf{x}_2\|_\infty} = L_\infty \quad (\text{EC.27})$$

Inequality (EC.26) follows from the Reverse Triangle Inequality. (EC.27) follows from the fact that $f(\mathbf{x})$ is linear and therefore, Lipschitz continuous using the $\|\cdot\|_\infty$ norm. We let L_∞ be the Lipschitz constant of $f(\mathbf{x})$.

Because $z(\mathbf{x}, \mathbf{u})$ satisfies the bounded and Lipschitz continuity assumptions, we apply Lemma EC.4 to obtain

$$\mathbb{E}_{\mathbf{u} \sim \mathbb{P}_{\mathbf{u}}} [z(F(\mathbf{u}), \mathbf{u})] \leq \frac{1}{N_{\mathbf{u}}} \sum_{i=1}^{N_{\mathbf{u}}} z(F(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) + K \sqrt{\frac{\log(1/\beta)}{2N_{\mathbf{u}}}} + L_\infty \mathfrak{R}_{N_{\mathbf{u}}}(\text{proj}(\mathcal{F})), \quad \forall F \in \text{proj}(\mathcal{F}).$$

Specifically, this bound holds for $F^* \in \text{proj}(\mathcal{F})$. By Lemma EC.3, we can bound $\mathfrak{R}_{N_{\mathbf{u}}}(\text{proj}(\mathcal{F})) \leq \sqrt{2n} \mathfrak{R}_{N_{\mathbf{u}}}(\mathcal{F})$.

The remainder of the proof follows from Markov's inequality. For $\gamma > 0$,

$$\begin{aligned} \mathbb{P}_{\mathbf{u}} \left\{ z(F^*(\mathbf{u}), \mathbf{u}) > \gamma \right\} &= \mathbb{P}_{\mathbf{u}} \left\{ |\mathbf{c}^\top F^*(\mathbf{u}) - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})| > \gamma \right\} \\ &\leq \frac{\mathbb{E}_{\mathbf{u} \sim \mathbb{P}_{\mathbf{u}}} \left[|\mathbf{c}^\top F^*(\hat{\mathbf{u}}_i) - \mathbf{c}^\top \mathbf{x}^\lambda(\hat{\mathbf{u}}_i)| \right]}{\gamma}. \end{aligned}$$

From the Law of Total Probability, we obtain

$$\begin{aligned} \mathbb{P}_{\mathbf{u}} \left\{ |\mathbf{c}^\top F^*(\mathbf{u}) - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})| \leq \gamma \right\} &= 1 - \mathbb{P}_{\mathbf{u}} \left\{ |\mathbf{c}^\top F^*(\mathbf{u}) - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})| > \gamma \right\} \\ &\geq 1 - \frac{\mathbb{E}_{\mathbf{u} \sim \mathbb{P}_{\mathbf{u}}} \left[|\mathbf{c}^\top F^*(\hat{\mathbf{u}}_i) - \mathbf{c}^\top \mathbf{x}^\lambda(\hat{\mathbf{u}}_i)| \right]}{\gamma}, \\ &\geq 1 - \frac{\frac{1}{N_{\mathbf{u}}} \sum_{i=1}^{N_{\mathbf{u}}} |\mathbf{c}^\top F^*(\hat{\mathbf{u}}_i) - \mathbf{c}^\top \mathbf{x}^\lambda(\hat{\mathbf{u}}_i)| + K \sqrt{\frac{\log(1/\beta)}{2N_{\mathbf{u}}}} + \sqrt{2n} L_\infty \mathfrak{R}_{N_{\mathbf{u}}}(\mathcal{F})}{\gamma}, \end{aligned}$$

with probability $1 - \beta$. The second and third line follow from Markov's inequality and substituting the bound from Lemma EC.4, respectively. Given that we have a probabilistic bound for the error of $F^*(\mathbf{u})$ from $\mathbf{x}^\lambda(\mathbf{u})$, we bound the error to $\mathbf{x}^*(\mathbf{u})$. Recall that $\mathbf{x}^\lambda(\mathbf{u})$ is (δ, ϵ) -optimal. There are two cases to consider. First, if $\mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u}) \leq \mathbf{c}^\top F(\mathbf{u}) \leq \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u}) + \gamma$, then by substitution,

$$\mathbf{c}^\top F^*(\mathbf{u}) - \epsilon - \gamma < \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) < \mathbf{c}^\top F^*(\mathbf{u}) + \delta.$$

Alternatively, if $\mathbf{c}^\top F^*(\mathbf{u}) \leq \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u}) \leq \mathbf{c}^\top F^*(\mathbf{u}) + \gamma$, then by substitution,

$$\mathbf{c}^\top F^*(\mathbf{u}) - \epsilon < \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) < \mathbf{c}^\top F^*(\mathbf{u}) + \delta + \gamma.$$

Note that both of these events can be covered by adding and subtracting γ to both the upper and lower bounds respectively. Then,

$$\mathbb{P}_{\mathbf{u}} \left\{ \mathbf{c}^\top F^*(\mathbf{u}) - \epsilon - \gamma < \mathbf{c}^\top \mathbf{x}^*(\mathbf{u}) < \mathbf{c}^\top F^*(\mathbf{u}) + \delta + \gamma \right\} \geq \mathbb{P}_{\mathbf{u}} \left\{ |\mathbf{c}^\top F^*(\mathbf{u}) - \mathbf{c}^\top \mathbf{x}^\lambda(\mathbf{u})| \leq \gamma \right\},$$

completing the proof. $\square$

EC.4. Supplementary content for automated portfolio optimization

We consider a personalized version of the non-convex portfolio optimization problem **PPP**($\mathbf{u}$) of Kim et al. (2019), where the objective is to maximize the return subject to constraints on the portfolio risk and the cardinality (i.e., the total number of stocks in which one can invest). Each user possesses their own problem parameters and requires an optimal solution according to their personalized problem.

In order to generate our initial data sets and oracle, we translate the cardinality constraint into a form that is amenable for optimization solvers. Kim et al. (2019) assume that any stock that is allocated less than 0.001 of the total portfolio does not affect the cardinality (i.e., $x_i \leq 0.001$ is equivalent to $x_i \approx 0$ for $i \in \{1, \dots, n\}$). We increase the bound to 0.005 and reformulate **PPP**($\mathbf{u}$) to a mixed-integer quadratic optimization problem:

$$\underset{\mathbf{x}, \mathbf{y}}{\text{minimize}} \quad -\boldsymbol{\mu}^\top \mathbf{x} \tag{EC.28a}$$

$$\text{subject to} \quad \mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x} \leq r(\mathbf{u}) \tag{EC.28b}$$

$$x_i \geq 0.005 y_i \quad \forall i \in \{1, \dots, n\} \tag{EC.28c}$$

$$x_i \leq y_i \quad \forall i \in \{1, \dots, n\} \tag{EC.28d}$$

$$\sum_{i=1}^n y_i \geq \lfloor c(\mathbf{u}) \rfloor \tag{EC.28e}$$

$$\sum_{i=1}^n y_i \leq \lfloor c(\mathbf{u}) \rfloor + \lceil \Delta c(\mathbf{u}) \rceil \tag{EC.28f}$$

$$y_i \in \{0, 1\} \quad \forall i \in \{1, \dots, n\} \tag{EC.28g}$$

Note that problem (EC.28) is non-convex only due to the binary variable $\mathbf{y}$. Thus, for any fixed $\mathbf{y}$ that satisfies the cardinality constraints (EC.28e) and (EC.28f), the corresponding problem (EC.28) becomes a convex quadratic program.

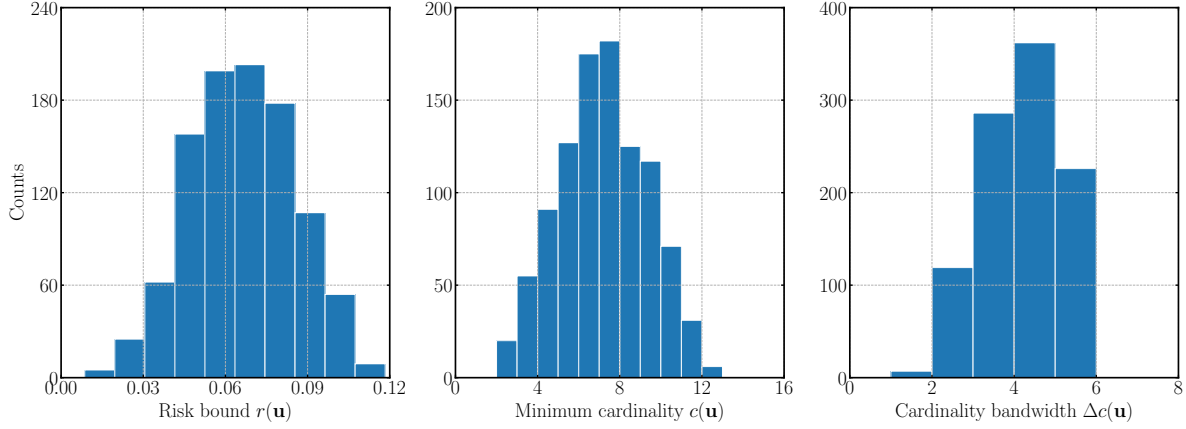
EC.4.1. Data set generation

We consider $\mathbf{PPP}(\mathbf{u})$ with $n = 100$ stocks and $p = 5$ features (i.e., $\mathcal{U} = \mathbb{R}^p$). We randomly generate a return vector $\boldsymbol{\mu} \in \mathbb{R}^n$ where $\mu_i \sim U[0, 1]$ is uniformly sampled i.i.d., as well as a random positive definite matrix $\boldsymbol{\Sigma} \in \mathbb{S}^{n \times n}$.

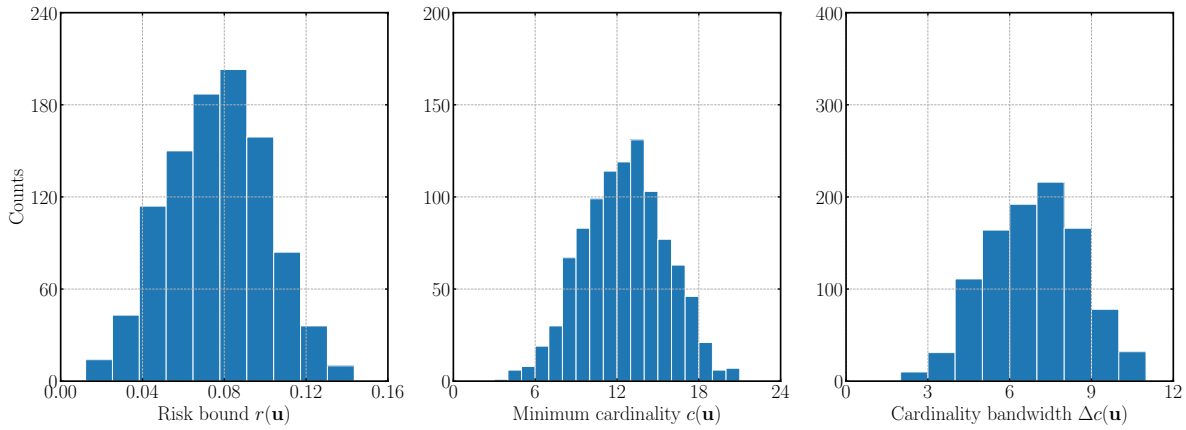
The histograms of the final parameter distributions for both experiments are given in Figure EC.2. We use the same data generation approach towards both experiments but with different parameter values. First, to validate the usefulness of IPMAN in predicting optimal decisions faster than an optimizer, we seek parameter settings for $r(\mathbf{u})$, $c(\mathbf{u})$, and $\Delta c(\mathbf{u})$ such that $\mathbf{PPP}(\mathbf{u})$ is difficult to solve within the latency requirements of an online application. The non-convex problem typically becomes more challenging for smaller values of these parameters. We randomly generate three linear functions $\mathbf{b}_r, \mathbf{b}_c, \mathbf{b}_\Delta \in \mathbb{R}^p$ where each element $b_i \sim U[0, 1]$ is uniformly sampled i.i.d. We then normalize each of these models such that $\|\mathbf{b}_r\| = 1/15$, $\|\mathbf{b}_c\| = 10$, and $\|\mathbf{b}_\Delta\| = 5$. Then, the “true models” that map the context vectors to each parameter are $r(\mathbf{u}) = \mathbf{b}_r^\top \mathbf{u}$, $c(\mathbf{u}) = \mathbf{b}_c^\top \mathbf{u} + 1$, and $\Delta(\mathbf{u}) = \mathbf{b}_\Delta^\top \mathbf{u} + 1$. We add 1 in the latter cases to prevent these parameters from being equal to 0. Under these settings, the average test set user has a risk tolerance $\mathbb{E}[r(\mathbf{u})] = 0.066$, minimum number of investments $\mathbb{E}[c(\mathbf{u})] = 7$, and minimum bandwidth $\mathbb{E}[\Delta c(\mathbf{u})] = 4$.

In the second experiment, we evaluate the robustness of IPMAN when given “noisy” versions of the true linear oracles. In order to reduce the overall runtimes over multiple trials, we consider true parameter settings that lead to easier optimization problems (i.e., with higher risk, minimum cardinality, and cardinality bandwidth parameters). Here, we use the same linear model, except by setting $\|\mathbf{b}_r\| = 1/12$, $\|\mathbf{b}_c\| = 16$, and $\|\mathbf{b}_\Delta\| = 8$. Under these settings, the average test user has a risk tolerance $\mathbb{E}[r(\mathbf{u})] = 0.076$, minimum number of investments $\mathbb{E}[c(\mathbf{u})] = 12$, and minimum bandwidth $\mathbb{E}[\Delta c(\mathbf{u})] = 6$.

We randomly generate a data set of 7000 users split between training, validation, and testing. For each user, we randomly generate a context vector $\hat{\mathbf{u}}_i$. Then, $\hat{\mathcal{U}}$ is the collection of 5000 context vectors for the training set users. We then randomly generate $\mathcal{D}$ and $\bar{\mathcal{D}}$ of 50000 points each by solving relaxations of problem (EC.28). To generate feasible decisions for each user, we randomly sample a vector $\mathbf{y}$ that satisfies constraints (EC.28e) and (EC.28f), fix this variable, and then solve problem (EC.28). To generate infeasible decisions for each user, we randomly sample a vector $\mathbf{y}$ that fails to satisfy the cardinality constraints, i.e., $\sum_{i=1}^n y_i > \lfloor c(\mathbf{u}) \rfloor + \lceil \Delta c(\mathbf{u}) \rceil$ or $\sum_{i=1}^n y_i < \lfloor c(\mathbf{u}) \rfloor$. We randomly generate 10 feasible and 10 infeasible decisions per user in the training set.



(a) Parameters for the experiment in Section 7.1.2. The average user in the test set has a risk tolerance of 0.066, minimum 7 investments, and up to 11 investments.



(b) Parameters for the experiment in Section 7.1.3. The average user in the tests set has a risk tolernace of 0.076, minimum 12 investments, and up to 18 investment.

Figure EC.2 Histogram of parameter values of $\text{PPP}(\mathbf{u})$ for the test set in the two experiments.

Table EC.1 Overview of the generator architecture. BN refers to batch normalization; LR and softmax refer to Leaky ReLU (0.2 slope) and Softmax activations, respectively.

Layer	Input shape	Activation
1	5	LR
2	200	BN-LR
3	200	BN-LR
4	200	softmax
Output	100	—

EC.4.2. Neural network architecture

Table EC.2 Overview of the classifier architecture. LR and sigmoid refer to Leaky ReLU (0.2 slope) and Sigmoid activations, respectively.

Layer	Input shape	Activation
1	105	LR
2	100	LR
3	100	LR
4	100	LR
5	100	sigmoid
Output	1	—

The architectures for the generator and classifier are described in Tables EC.1 and EC.2, respectively. Both the generator and classifier are fully-connected neural networks. The generator takes as input the context vector $\mathbf{u} \in \mathbb{R}^5$ and is composed of four layers. Each layer has a Leaky ReLU activation and the inner layers also incorporate batch normalization. The final layer of the generator is passed through a softmax activation function which ensures that $F(\mathbf{u})$ is a vector of positive elements which sum up to one. Consequently, the generator will always output a valid portfolio that satisfies the known polyhedral constraints and the core challenge is to learn to satisfy the hidden constraints. The classifier takes as input $(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^{105}$ and is composed of five layers with Leaky ReLU activations between each layer. The final output is passed through a Sigmoid activation to ensure that $B(\mathbf{x}, \mathbf{u}) \in [0, 1]$.

EC.4.3. Implementation of the IPMAN algorithm

We describe details of our IPMAN training algorithm, which follows the procedure described in Algorithm 1. All models are trained using the Adam optimizer with $(\beta_1, \beta_2) = (0.9, 0.999)$. We train the classifier with a batch size of 2000 data points, a learning rate of 1×10^{-2} , and plateau-scheduling decay. We train the generator with a batch size of 1000 data points, a learning rate of 5×10^{-3} , and plateau-scheduling delay. In each iteration of the IPMAN algorithm, we train the generator and classifier for 10 epochs each.

Just as classical IPMs require a good initial point (i.e., lying within the feasible set) in order to construct a trajectory of points leading to an optimal solution, IPMAN can be made more efficient by ensuring that the generative model is initialized to predict points

that are likely to be feasible. This initialization can greatly improve the training time and stability of the algorithm. Consequently, we first pre-train both the generator and the classifier at the beginning of the algorithm (Goodfellow et al. 2016).

The classifier is pre-trained using the initial data set of feasible $\mathcal{D}^{(0)}$ and infeasible decisions $\bar{\mathcal{D}}^{(0)}$ with the same learning rate and settings as used in IPMAN, but for 200 epochs. This initial pre-training ensures that the classifier is warm-started to a good barrier function from the offset of the IPMAN algorithm and therefore reduces the overall training time required. That is, without pre-training the classifier, we would require running more iterations of IPMAN. Note that pre-training the classifier is equivalent to training the classifier in the first iteration of IPMAN for 200 epochs and subsequently training for 10 epochs per iteration in subsequent iterations.

The generator is pre-trained using a sub-set of $\mathcal{D}^{(0)}$ constructed by randomly selecting a single feasible decision $\hat{\mathbf{x}}$ per context vector $\hat{\mathbf{u}}$ in $\mathcal{D}^{(0)}$ (i.e., a total of 5000 points). With this sub-set, we train the classifier to minimize mean-squared error, similar to prior approaches to predicting decisions (Larsen et al. 2018). We use the same training settings for pre-training the generator as we do in IPMAN and train for 200 epochs.

EC.5. Supplementary content for personalized cancer treatment

RT treatment is delivered by a linear accelerator (LINAC) that projects high-energy X-rays from different angles to a patient’s tumor. The patient’s body is discretized into voxels (i.e., $5\text{mm} \times 5\text{mm} \times 2\text{mm}$ volumetric pixels) and the dose delivered to each of these voxels is used to assess the quality of a treatment. The design of an RT treatment plan is typically done by mathematical optimization where the decision variable has two components (i) beamlet intensity and (ii) dose delivered (in Gy). Most automated planning systems involve predicting doses that are clinically desirable for a given patient. In practical implementations, the appropriate beamlets that can reconstruct a predicted dose distribution can be obtained via a secondary optimization process (Babier et al. 2018). In this work, we focus on constructing clinically desirable dose distributions.

Each patient contains seven organs-at-risk (OARs) (i.e., brainstem, spinal cord, right parotid, left parotid, larynx, esophagus, and mandible) to which we minimize the average dose. Each patient also contains up to three planning target volumes (PTVs) with different prescription doses (i.e., PTV56, PTV63, and PTV70 with 56 Gy, 63 Gy, and

70Gy as prescription doses, respectively). We remark that constraints to the brainstem, spinal cord, and esophagus are generally easily satisfied by all predictions. Consequently, we focus specifically on the right parotid, left parotid, larynx, mandible, PTV56, PTV63, and PTV70.

In the RT optimization problem, each of the OARs and targets require polyhedral upper and lower bound constraints to the mean dose delivered to that structure. Furthermore, there exists a hidden “clinical criteria” constraint for each OAR and target that must be satisfied at the discretion of an oncologist. That is, if the ground truth treatment plan for a given patient from the data set satisfies a hidden constraint, then any generated plan for that patient must also satisfy that constraint. The hidden constraint for each OAR is an upper bound on either the mean or maximum dose delivered to that structure, while the hidden constraint for each target is a lower bound on the value-at-risk, i.e., minimum dose delivered to 90-th percentile of the target structure. The oracle $\Psi(\mathbf{x}, \mathbf{u})$ is a look-up table that compares the dose generated by our model with the ground truth (i.e., what was actually delivered). In particular, for each structure, $\Psi(\mathbf{x}, \mathbf{u})$ checks whether the input dose satisfies all the constraints (i.e., two polyhedral and one hidden constraints). We expand on the classification of feasibility in the sections below.

EC.5.1. Neural network architecture

We use a modified version of the generative adversarial network (GAN) of (Babier et al. 2019), where two networks learn to predict dose distributions. The architectures for $F(\mathbf{u})$ and $B(\mathbf{x}, \mathbf{u})$ are described in Tables EC.3 and EC.4, respectively.

The generator takes as input a tensor $\mathbf{u} \in \mathbb{R}^{128 \times 128 \times 128 \times 8}$, where the first three dimensions correspond to a voxel in the patient’s geometry. The fourth dimension is a concatenation of the CT image greyscale and a one-hot encoded vector in $\{0, 1\}^7$ whose elements label whether the voxel belongs to one of the seven contoured structures. The generator then outputs a tensor $\mathbf{x} \in \mathbb{R}^{128 \times 128 \times 128}$ whose elements specify the dose to be delivered to each voxel of the patient.

The classifier is trained to predict whether a given dose distribution satisfies all of the constraints (both hidden and polyhedral) for each structure of the patient. This network takes as input the concatenated tensor $(\mathbf{x}, \mathbf{u})$ and outputs a vector in $[0, 1]^7$, whose elements each indicate the classifier’s belief of whether the given dose distribution has satisfied all of the constraints for each specific structure. Consequently, learning feasibility becomes a

Table EC.3 Overview of the generator architecture. BN refers to batch normalization; LR, R, and tanh refer to Leaky ReLU (0.2 slope), ReLU, and Tanh activations, respectively; AP refers to a mean pool; and D refers to dropout.

Layer	Concatenate with	Input shape	Block	Activation
1	—	$128 \times 128 \times 128 \times 8$	conv3d	BN-LR
2	—	$64 \times 64 \times 64 \times 64$	conv3d	BN-LR
3	—	$32 \times 32 \times 32 \times 128$	conv3d	BN-LR
4	—	$16 \times 16 \times 16 \times 256$	conv3d	BN-LR
5	—	$8 \times 8 \times 8 \times 512$	conv3d	BN-LR
6	—	$4 \times 4 \times 4 \times 512$	conv3d	BN-LR
7	—	$2 \times 2 \times 2 \times 512$	deconv3d	LR
8	layer 5 output	$4 \times 4 \times 4 \times 1024$	deconv3d	BN-R
9	layer 4 output	$8 \times 8 \times 8 \times 1024$	deconv3d	BN-D-R
10	layer 3 output	$16 \times 16 \times 16 \times 512$	deconv3d	BN-D-R
11	layer 2 output	$32 \times 32 \times 32 \times 256$	deconv3d	BN-R
12	layer 1 output	$64 \times 64 \times 64 \times 128$	deconv3d	AP-tanh
Output	—	$128 \times 128 \times 128 \times 1$	—	—

multi-label classification problem and the classifier acts as seven separate classifiers each predicting feasibility with respect to an individual structure, but whose model parameters are shared with each other. For any structure, in order to classify a dose distribution as satisfying the relevant constraints, the classifier must: (i) first determine from the dose whether the polyhedral constraints are satisfied, (ii) determine from the CT image whether the patient requires a hidden constraint to be satisfied, and (iii) determine from the dose whether the hidden constraint is satisfied if this constraint is required for the patient. Overall, a dose distribution is feasible only if all constraints are satisfied.

EC.5.2. Implementation of the IPMAN algorithm

We describe the exact implementation of the IPMAN algorithm used in our experiments. We summarize the steps in Algorithm 4. All models are trained using the Adam optimizer with $(\beta_1, \beta_2) = (0.5, 0.999)$. We train the classifier with a learning rate of 1×10^{-3} and the generator with a learning rate of 2×10^{-5} . We remark on several modifications of the IPMAN algorithm that we implement for these experiments.

Table EC.4 Overview of the classifier architecture. BN refers to batch normalization; LR, R, and sigmoid refer to Leaky ReLU (0.2 slope), ReLU, and Sigmoid activations, respectively.

Layer	Input shape	Block	Activation
1	$128 \times 128 \times 128 \times 9$	conv3d	LR
2	$64 \times 64 \times 64 \times 64$	conv3d	BN-LR
3	$32 \times 32 \times 64 \times 128$	conv3d	BN-LR
4	$16 \times 16 \times 16 \times 256$	conv3d	BN-LR
5	$8 \times 8 \times 8 \times 512$	conv3d	sigmoid
Output	7	—	—

Algorithm 3 Generator pre-training and data augmentation

Input: Feasible and input data sets $\mathcal{D} = \{(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i)\}_{i=1}^{N_u}$, $\hat{\mathcal{U}} = \{\hat{\mathbf{u}}_i\}_{i=1}^{N_u}$, infeasible data set $\bar{\mathcal{D}} = \emptyset$, Pre-training number of epochs E_{ST} .

Output: Pre-trained generative model $F^{(0,0)}$, Feasible and infeasible data sets $\mathcal{D}, \bar{\mathcal{D}}$.

- 1: Initialize generator and discriminator F, D
 - 2: **for** $e = 1$ **to** E_{ST} **do**
 - 3: Update generator and discriminator $F^*, D^* \leftarrow \text{Adam}(\nabla L_{ST})$.
 - 4: **for all** $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$ **do**
 - 5: Append $\mathcal{D} \leftarrow \mathcal{D} \cup (F^*(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i)$ if $\Psi(F^*(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) = 1$ else $\bar{\mathcal{D}} \leftarrow \bar{\mathcal{D}} \cup (F^*(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i)$.
 - 6: **end for**
 - 7: **end for**
 - 8: **return** $F^{(0,0)} \leftarrow F^*, \mathcal{D}, \bar{\mathcal{D}}$.
-

EC.5.2.1. Pre-training as a GAN Similar to the experiments on personalized portfolio optimization, we first pre-train the generative model and subsequently apply transfer learning at the beginning of the algorithm (Goodfellow et al. 2016). Pre-training amounts to training the generative model first as a Style Transfer GAN to learn to predict dose distributions from CT images as in Mahmood et al. (2018). The steps are summarized in Algorithm 3. In order to pre-train our generative model, we introduce a discriminator network $D(\mathbf{x}, \mathbf{u}) : \mathbb{R}^{128 \times 128 \times 128} \times \mathbb{R}^{128 \times 128 \times 128 \times 8} \rightarrow \mathbb{R}$. As in a Style Transfer GAN, we train the discriminator using $\mathcal{D}$ to classify whether a given dose and CT image pair belongs to the data distribution and a generative model to predict dose distributions that fool the

Algorithm 4 IPMAN

Input: Data sets of decisions $\mathcal{D}$, $\bar{\mathcal{D}}$, and inputs $\hat{\mathcal{U}} = \{\hat{\mathbf{u}}_i\}_{i=1}^{N_{\mathbf{u}}}$, Set of dual variables $\{\lambda_j\}_{j=0}^M$,Number of iterations K , Number of epochs E_B, E_F , Subset sampling rate s **Output:** Final generative models $F^{(j,K)}$ for $j \in \{0, \dots, M\}$

```

1: Pre-train generator using Algorithm 3.
2: Initialize generator  $F^{(j,0)} \leftarrow F^*$  for  $j \in \{0, \dots, M\}$ , classifier  $B$ .
3: for  $k = 1$  to  $K$  do
4:   Sample subsets to train  $\mathcal{D}^{(k)} = \sigma(\mathcal{D}; s)$ ,  $\bar{\mathcal{D}}^{(k)} = \sigma(\bar{\mathcal{D}}; s|\mathcal{D}|/|\bar{\mathcal{D}}|)$ .
5:   for  $e = 0$  to  $E_B$  do
6:     Update classifier  $B^{(k)} \leftarrow \text{Adam}(\nabla L_B)$ .
7:   end for
8:   for  $j = 0$  to  $M$  do
9:     for  $e = 0$  to  $E_F$  do
10:      Update generator  $F^{(j,k)} \leftarrow \text{Adam}(\nabla L_F)$ .
11:    end for
12:    for all  $\hat{\mathbf{u}}_i \in \hat{\mathcal{U}}$  do
13:      Append  $\mathcal{D} \leftarrow \mathcal{D} \cup (F^{(j,k)}(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i)$  if  $\Psi(F^{(j,k)}(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) = 1$  else  $\bar{\mathcal{D}} \leftarrow \bar{\mathcal{D}} \cup$ 
         $(F^{(j,k)}(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i)$ .
14:    end for
15:  end for
16: end for
17: return  $F^{(j,K)}$  for  $j \in \{0, \dots, M\}$ 

```

discriminator. Specifically,

$$\min_F \max_D \left\{ L_{ST} := \frac{1}{N_{\mathbf{u}}} \sum_{(\hat{\mathbf{u}}_i, \hat{\mathbf{x}}_i) \in \mathcal{D}} \log D(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i) + \log(1 - D(F(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i)) + \lambda_{ST} \|F(\hat{\mathbf{u}}_i) - \hat{\mathbf{x}}_i\|_1 \right\}.$$

The l_1 -loss is a regularization term that ensures that the generator predicts dose distributions that resemble the ground truth and λ_{ST} is the regularization parameter. GANs are trained by iterative gradient descent between $F(\mathbf{u})$ and $D(\mathbf{x}, \mathbf{u})$. In our implementation, we set $\lambda_{ST} = 90$ and train the GAN for 50 epochs, following the practice from Babier et al. (2019). At the end of pre-training, we discard $D(\mathbf{x}, \mathbf{u})$ and let $F^{(0,0)}(\mathbf{u})$ denote the trained generative model.

EC.5.2.2. Sampling an infeasible data set of decisions $\bar{\mathcal{D}}$. Training IPMAN requires an initial data set of infeasible decisions $\bar{\mathcal{D}}$. In practice, a data set of infeasible decisions would not be available a priori and, instead, is generated by sampling. Note, however, that in every epoch of the pre-training step, the generative model generates candidate solutions $F(\hat{\mathbf{u}}_i)$ to attempt to fool the discriminator. We save the generated decisions during pre-training and label them afterwards as feasible or infeasible using the oracle. By training using 100 patients for 50 epochs, we generate a total of 5000 dose distributions that are labelled as feasible or infeasible and then binned in the appropriate $\mathcal{D}$ or $\bar{\mathcal{D}}$, respectively. The steps are summarized in Algorithm 3.

EC.5.2.3. Learning multi-label feasibility with sub-sampled data sets. Training IPMAN for multiple iterations can produce a large quantity of generated data points. Furthermore, because we consider feasibility for each structure separately, training the classifier quickly becomes prohibitively expensive. In order to reduce training time, we do not use the entire data sets $\mathcal{D}$ and $\bar{\mathcal{D}}$ but rather smaller sampled subsets. Let $\sigma(\cdot; s)$ be a random sampling operator (without replacement) where s is the fraction of points to sample. For example, $\sigma(\mathcal{D}; 0.5)$ denotes a randomly sampled subset of size $0.5|\mathcal{D}|$. In our implementation, we set $s = 0.3$ and trained the classifier using $\mathcal{D}^{(k)} = \sigma(\mathcal{D}; 0.3)$ and $\bar{\mathcal{D}}^{(k)} = \sigma(\bar{\mathcal{D}}; 0.3|\mathcal{D}|/|\bar{\mathcal{D}}|)$; this reduced the training time to 24 hours.

We next define the multi-label classification problem. For any $(\hat{\mathbf{u}}_i, \hat{\mathbf{x}}_i)$ in $\mathcal{D}$ or $\bar{\mathcal{D}}$, let $\psi_{i,r}$ be a label determining whether the dose distribution had satisfied the polyhedral and (conditional) hidden constraints for structure r . That is, if the ground truth dose for $\hat{\mathbf{u}}_i$ satisfied the hidden constraints, $\psi_{i,r} = 1$ if the polyhedral and hidden constraints were satisfied and zero otherwise. If the clinical dose did not satisfy the hidden constraint, then $\psi_{i,r} = 1$ if only the polyhedral constraints were satisfied; here, the hidden constraint is inactive for this patient. Then, let $[B(\mathbf{x}, \mathbf{u})]_r$ denote the r -th element of the classifier output. The classifier problem is

$$\max_{B \in \mathcal{B}} \left\{ L_B := \frac{1}{N_{\mathbf{x}} + \bar{N}_{\mathbf{x}}} \sum_{(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i) \in \mathcal{D}^{(k)} \cup \bar{\mathcal{D}}^{(k)}} \sum_{r \in \mathcal{R}} \left(\psi_{i,r} \log [B(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i)]_r + (1 - \psi_{i,r}) \log \left(1 - [B(\hat{\mathbf{x}}_i, \hat{\mathbf{u}}_i)]_r \right) \right) \right\}.$$

The above problem specializes to **FCP**($\mathcal{D}^{(k)}, \bar{\mathcal{D}}^{(k)}$) in the single-class setting (i.e., $|\mathcal{R}| = 1$). For a dose distribution to be classified feasible, $B(\mathbf{x}, \mathbf{u})_r = 1$ for all $r \in \mathcal{R}$. This approach

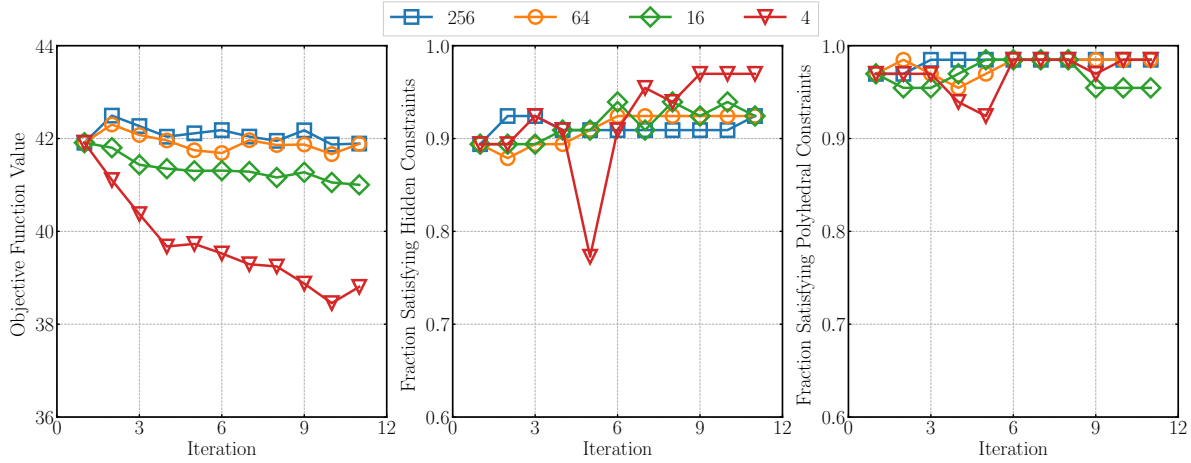
of separating the constraint satisfaction along all structures individually is equivalent to modeling the optimization problem via a barrier function for each structure. Furthermore, the barriers are approximated by a neural network classifier with shared weights except in the last layer. As we describe later below, the objective of the barrier optimization problem is obtained by summing all of the separate barriers, i.e., $f(\mathbf{x}) - \lambda \sum_{r \in \mathcal{R}} \log[B(\mathbf{x}, \mathbf{u})]_r$. Finally, we minimize L_B using the Adam optimizer for $E_B = 10$ epochs in every iteration. Note that it is essential to ensure that the classifier accurately predicts feasibility in order to be able to approximate a δ -barrier.

EC.5.2.4. Regularized barrier optimization problem. We include an l_1 regularization term in training. This term is equivalent to the one used in the pre-training stage and is useful to ensure that predicted dose distributions do not deviate too far from the ground truth. Note that we only use this regularization in the first set of experiments (Section 7.2.2) and remove it in the second set of experiments (Section 7.2.3). There, the ground truth plans may not be feasible, meaning that it would be incorrect to replicate ground truth behavior. However, a consequence of removing a regularization term is that certain models may become unstable and deviate significantly if the classifier is not a complete δ -barrier. We observe this behavior in $\lambda = 4$ where the model minimizes dose while ignoring feasibility.

Let $\mathbf{c}$ denote the linear cost vector denoting the average of average doses to the OARs. Then, the generative barrier problem is

$$\min_{F \in \mathcal{F}} \left\{ L_F := \frac{1}{N_{\mathbf{u}}} \sum_{\hat{\mathbf{u}}_i \in \tilde{\mathcal{U}}} \left(\frac{1}{\lambda_j} \mathbf{c}^\top F(\hat{\mathbf{u}}_i) + \lambda_{ST} \|F(\hat{\mathbf{u}}_i) - \hat{\mathbf{x}}_i\|_1 - \sum_{r \in \mathcal{R}} \left[B^{(k)}(F(\hat{\mathbf{u}}_i), \hat{\mathbf{u}}_i) \right]_r \right) \right\}.$$

We minimize L_F using the Adam optimizer for $E_F = 1$ epoch in every iteration. It is important to ensure that the classifier is trained close to optimality to ensure that it approximates a δ -barrier. Furthermore, given the nature of training the classifier, it is often the case that the classifier's support is uneven and may have areas of local optimality that the generator may abuse. A standard practice in the GAN literature is to control the training duration of the two networks; we employ a similar strategy by training the generative model for a shorter duration than the classifier in order to ensure that the generator does not overfit and abuse local optima caused by the classifier. As previously mentioned, we set $\lambda_{ST} = 50$ for the first set of experiments but require that $\lambda_{ST} = 0$ for the second set.



(a) Objective function and fraction of feasible plans with respect to the hidden and polyhedral constraints.

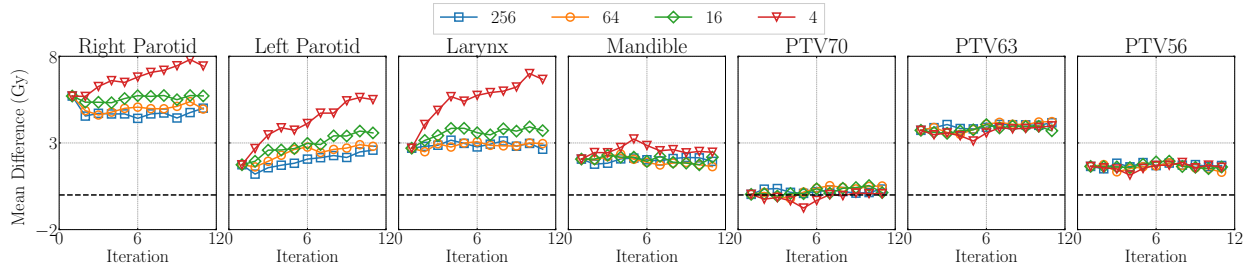
(b) Average difference from the hidden constraint bound z_r . The dashed line is 0 Gy. Above 0 suggests plans satisfy the constraints on average.**Figure EC.3** Statistics on the validation set obtained during training on criteria from our institution.**EC.5.3. Supplementary results for learning to predict feasible and optimal dose distributions**

Figure EC.3(a) displays the average objective function value and the fraction of plans that satisfy the hidden and polyhedral constraints over training iterations. We find that in training, the objective function value improves as a function of the number of iterations, while hidden constraint satisfaction also increases. For example, in the first iteration, 89% of predictions in the validation set satisfy all of their hidden constraints, whereas this fraction increases to 97% by iteration 11. Polyhedral constraint satisfaction also increases from 88% to 95%. This suggests that the IPMAN algorithm trains the model to generate fewer infeasible doses. In other words, the classifier is learning to produce a tighter characterization of the feasible set (see Proposition 1).

Figure EC.3(b) shows the average difference from the boundary of the hidden constraint for each structure. If the difference is positive, doses on average satisfy the hidden criteria. We observe two important phenomena. First, the four leftmost plots are associated with

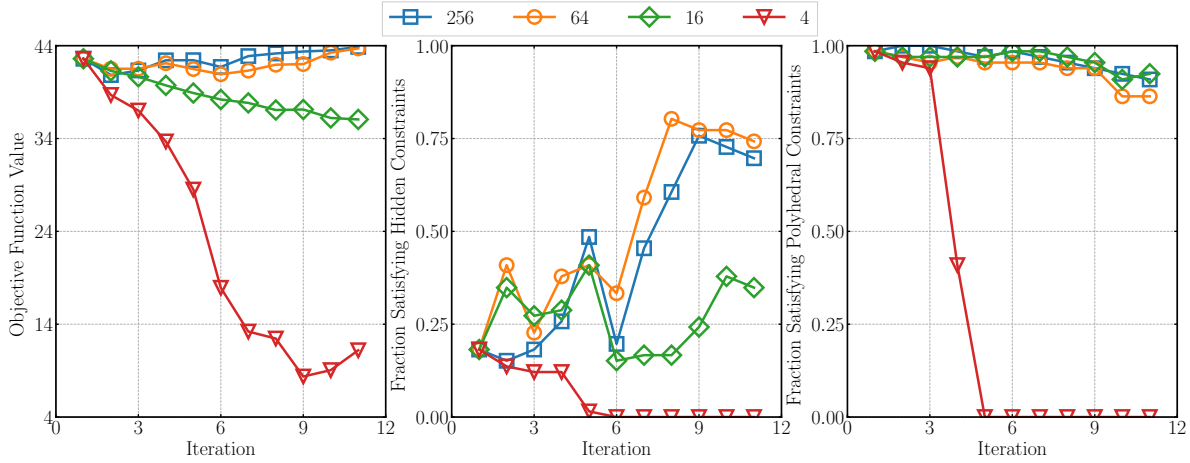
Table EC.5 The percentage of predicted decisions on the hold-out test set that satisfy each hidden constraint to 1 Gy relaxation. The best performing models on the summary statistics are highlighted.

Structure r	Criteria (Gy)	Baselines		IPMAN (λ)			
		GAN	CNN	256	64	16	4
	$g_r(\mathbf{x}) \leq z_r$						
Right Parotid	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 26$	85.7	85.7	86.2	90.0	93.3	100
Left Parotid	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 26$	70.0	60.0	70.0	90.0	90.0	100
Larynx	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 45$	93.3	83.3	89.7	89.7	93.3	100
Mandible	$\text{max}\{x_v \mid v \in \mathcal{V}_r\} \leq 73.5$	100	100	100	100	100	100
PTV70	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 70$	97.6	97.6	97.6	97.6	95.2	92.8
PTV63	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 63$	96.3	96.3	96.3	96.3	96.3	96.3
PTV56	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 56$	100	100	100	100	100	100
Fraction of predictions $\mathbf{x} \in (\cap_r \mathcal{X}_r(\mathbf{u}))$ (%)		86.0	78.0	82.0	88.0	88.0	94.0
Fraction of predictions $\mathbf{x} \in \mathcal{P}$ (%)		92.0	90.0	94.0	92.0	90.0	94.0
Average objective function value (Gy)		40.3	41.0	41.0	41.0	40.0	37.8

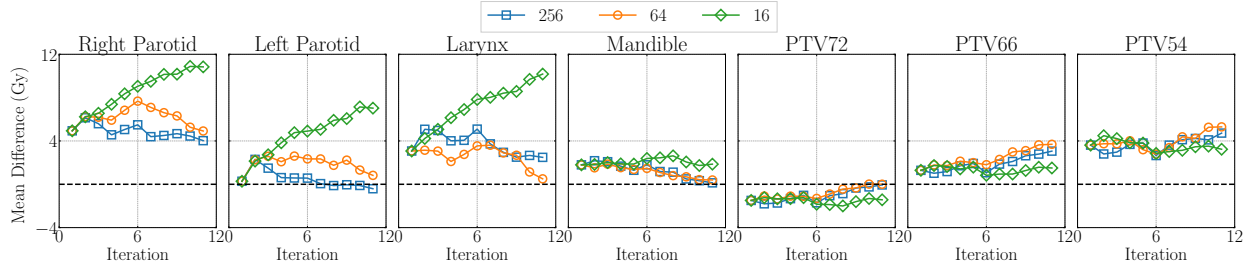
OAR constraints. By minimizing the objective, the associated OAR constraints see progressively better adherence as expected. Note that the Mandible and PTV70 structures often overlap, meaning their constraints conflict with each other, preventing improvement for this organ. Second, the PTV constraints show small but sustained improvement as the number of training iterations increase. This is because they are solely associated with feasibility and are not part of the objective function. In particular, the PTV70 constraint is typically the hardest to satisfy in practice; IPMAN learns this difficulty and makes predictions that lie close to the boundary of the feasible set.

Table EC.5 extends Table 3 and shows performance on the hold-out test set for IPMAN at iteration 11 against the baseline models for all values of λ . In general, all IPMAN models with $\lambda \leq 64$ satisfy the hidden constraints better than the baselines and IPMAN with $\lambda = 4$ dominates all other models. We observe that with higher values of λ , constraint satisfaction comes with a price; the objective function value is higher.

Recall that optimal solutions to the barrier problem satisfy an (δ, ϵ) -optimality guarantee. At high λ , this translates to a non-trivial upper bound with respect to the optimal value of the true problem. For low values of λ , the guarantee translates to a non-trivial



(a) Objective function and fraction of feasible plans with respect to the hidden and polyhedral constraints.

(b) Average difference from the hidden constraint bound z_r . The dashed line is 0 Gy. Above 0 suggests plans satisfy the constraints on average. $\lambda = 4$ is omitted to preserve scale.**Figure EC.4** Statistics on the validation set obtained during training on criteria from Geretschlager et al. (2015).

lower bound (see Theorem 1 and EC.2). Because the generator learns via empirical risk minimization of the barrier problem, the corresponding predictions should also satisfy similar upper and lower bounds (see Theorem 2 and 3). While we may expect a low λ to yield predictions that are infeasible (i.e., satisfying a non-trivial lower bound with respect to the optimal value), we find that $\lambda = 4$ yields the best performance on out-of-sample data.

EC.5.4. Supplementary results for adapting to the clinical constraints of a different institution

Figure EC.4(a) displays the objective function value and the fraction of plans that satisfied the hidden and polyhedral constraints. The models trained for $\lambda \leq 16$ decrease in objective function value and constraint satisfaction as the algorithm progresses. In the early stages, the classifier (which is initially trained mainly using the clinical data and doses sampled from the same distribution) has not yet observed a sufficient and diverse number of feasible

Table EC.6 The percentage of predicted decisions on the hold-out test set that satisfy each hidden constraint of Geretschl ger et al. (2015) to 1 Gy relaxation. The best performing models on the summary statistics are highlighted.

Structure	Criteria (Gy)	Baseline	IPMAN (λ)				
			$g_r(\mathbf{x}) \leq z_r$	GAN	256	64	16
Right Parotid	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 26$	83.3	85.7	85.7	100	100	
Left Parotid	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 26$	70.0	50.0	60.0	100	100	
Larynx	$\text{mean}\{x_v \mid v \in \mathcal{V}_r\} \leq 45$	93.3	86.7	76.8	100	100	
Mandible	$\text{max}\{x_v \mid v \in \mathcal{V}_r\} \leq 73.5$	100	81.0	90.5	100	100	
PTV70	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 70$	7.31	95.2	95.2	14.3	0	
PTV63	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 63$	77.8	96.3	96.3	85.2	0	
PTV56	$\text{VaR}_{90}\{x_v \mid v \in \mathcal{V}_r\} \geq 56$	100	100	100	96.0	0	
Fraction of predictions $\mathbf{x} \in (\cap_r \mathcal{X}_r(\mathbf{u}))$ (%)			18.3	64.0	66.0	26.0	0
Fraction of predictions $\mathbf{x} \in \mathcal{P}$ (%)			93.8	94.0	90.0	88.0	0
Average objective function value (Gy)			40.3	42.3	42.3	34.3	10.0

plans. Therefore, the classifier is not yet a sufficient δ -barrier, allowing the generator to leave the feasible set.

Generally at high λ , optimal solutions to the barrier problem are less aggressive in terms of minimizing the objective and instead lie well in the interior of the feasible set (see EC.2). As the classifier improves, particularly after iteration 6, the generative models for $\lambda \geq 64$ quickly learn to predict solutions that are more likely to satisfy the hidden constraints. In particular, the distance from the PTV72 boundary in Figure EC.4(b) begins to increase from the 6th iteration and passes 0 (i.e., satisfy the hidden constraint) by the 10th iteration. This result demonstrates that our model is learning a new constraint, the PTV72 hidden criteria, which cannot be learned by naively using the training data. Recall that no clinical dose in the original data set reached 72 Gy on the PTV70. Thus, learning the PTV72 constraint is entirely attributable to the IPMAN procedure.

Table EC.6 shows the performance of all IPMAN models on the hold-out test set for the generator at iteration 11. Overall, hidden constraint satisfaction is 64.0% with 96.3% and 95.2% of plans satisfying the PTV66 and PTV72, constraints, respectively, when $\lambda = 256$ or

64. Nevertheless, as a result of learning higher doses to the PTVs, constraint satisfaction on the OARs slightly degrades; this can be observed in the slightly higher objective function values.