Learning-Augmented Algorithms for Online Concave Packing and Convex Covering Problems

Elena Grigorescu
University of Waterloo

Young-San Lin
Melbourne Business School

Maoyuan Song
Purdue University

Abstract

Learning-augmented algorithms have been extensively studied in the computer science community recently, particularly in the context of online problems, in which machine-learning predictors can help provide additional information about the future, in order to overcome classical impossibility results. Such algorithms use *advice* prudently to improve the performance of classical algorithms, while ensuring robustness against inaccurate advice. In this paper, we present learning-augmented algorithmic frameworks for two fundamental optimizations settings, extending and generalizing prior works. For *online packing with concave objectives*, we present a simple but overarching strategy that *switches* between the advice and the state-of-the-art online algorithm. For *online covering with convex objectives*, we greatly extend primal-dual methods for online convex covering programs and previous learning-augmented frameworks for online covering linear programs from the literature to many new applications. We show that our algorithms break impossibility results when the advice is accurate, while maintaining comparable performance with state-of-the-art classical online algorithms even when the advice is erroneous.

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

1 INTRODUCTION

In the classical online model, algorithms are given parts of the input over time, and must make irrevocable decisions to process the input elements as they arrive, without knowledge of the future. The performance of an online algorithm is often measured by its "competitive ratio", which is defined as the ratio between the "cost" or "value" of the algorithm's solution and that of the optimal solution in hindsight, or equivalently that of the solution of an optimal offline algorithm. Due to the uncertainty around future inputs, online problems are traditionally hard, and often exhibit impossibility results, which are lower bounds on the competitive ratio of any online algorithm (e.g., Karlin et al., 1994; Alon et al., 2009; Balseiro and Gur, 2019). We refer the readers to Buchbinder and Naor (2009b) and Hoi et al. (2021) for excellent surveys on online algorithms.

The on-going success of machine learning has led to the break-through concept of using predictions (Purohit et al., 2018; Lykouris and Vassilvitskii, 2021), i.e., additional information inferred from historical data about future inputs or the problem instance as a whole, and incorporating them into classical online algorithms, to achieve better performance and break impossibility results. However, due to the probabilistic nature of machine learning, as well as the inability of online algorithms to verify the accuracy of these predictions, blindly trusting and following the advice can lead to undesirable performance compared to even online algorithms without predictions, as established by Szegedy et al. (2013) and Bamas et al. (2020b). As a result, the focus of the community has shifted to *learning-augmented algorithms*, namely algorithms that utilize the advice prudently, maintaining rigorous guarantees and high performance, both when the advice is accurate, in which case we say that it maintains *consistency*, and when the advice is arbitrarily inaccurate, in which case we say that it satisfies *robustness*. We refer the

readers to Mitzenmacher and Vassilvitskii (2022) for a survey on learning-augmented algorithms.

Online covering and packing problems. A recent line of work studies learning-augmented algorithms in context of online covering linear programs (LPs). The seminal work of Bamas et al. (2020b) built upon the primal-dual framework of Buchbinder and Naor (2009a) and presented the first primal-dual learning-augmented (PDLA) algorithm for online set cover. Combining their algorithms with ideas from Elad et al. (2016), the work of Grigorescu et al. (2022a) generalized the primal-dual learning-augmented framework to solve general online covering LPs and semidefinite programs, allowing for fractional cost, constraint, and advice. The PDLA framework utilizes the duality of covering and packing linear programs, and maintains a primal covering solution as well as a dual packing solution simultaneously, while fine-tuning the growth rate of each individual variable using the advice provided to the algorithm. Compared to many other learning-augmented algorithms for specific online problems, the PDLA framework is general-purpose, and can apply to a variety of problems oblivious of structure.

Our contributions at a glance. In this paper, we extend the line of work on online covering and packing problems, by designing algorithmic frameworks for online packing maximization with concave objectives, and online covering minimization with convex objectives, further generalizing the setting of Grigorescu et al. (2022a). Our settings and frameworks are general-purpose, and can be directly applied to a variety of problems, agnostic of problem structure. Concave packing and convex covering are traditionally considered strongly associated "dual" problems to each other in operations research and online optimization; Nonetheless, we show that these two settings admit vastly different learning-augmented algorithms despite their similarity.

For online packing with concave objectives, we present a simple class of algorithms utilizing a similar "switching" strategy to that outlined in Grigorescu et al. (2022b) (attributed to Roie Levin), complementing their observation that even such simple strategies can outperform sophisticated algorithms for online covering linear programs. Switching between different algorithms has been a classical design philosophy in classical online algorithms (Fiat et al., 1991), but is much less prevalent in learning-augmented algorithms (e.g., Antoniadis et al., 2023).

For online covering with convex objectives, it appears that "switching" strategies do not admit efficient algorithms, thus we instead present a primal-dual-based framework, taking inspiration from the classical primal-dual method for online convex covering in Azar et al. (2016). As an extended result, we adapt our PDLA framework to the online covering setting with ℓ_q-norm objectives studied in Shen and Nagarajan (2020), utilizing problem structure to obtain better analyses. Finally, we apply our proposed frameworks to a variety of online optimization problems, including network utility maximization, optimization under inventory constraints, mixed covering and packing, and buy-at-bulk network design.

Conceptually, from our observations around the "switching" paradigm, we raise some basic questions: What properties enable these switching strategies to perform well, that are present in concave packing, but absent in convex covering? More generally, which online problems allow such simple algorithms and solutions to exist? We believe that understanding these conceptual questions are important for the field of learning-augmented algorithms.

Additional prior works. We present an overview of additional related works in Appendix A.

Organization. We state necessary background knowledge in Section 1.1, and outline our contributions in more detail in Section 1.2. In Section 2, we present a simple "switching"-based overarching framework for online concave packing. In Section 3, we present an overview for our primal-dual learning-augmented framework for online convex covering, and its full analysis in Appendix B. In Appendix C, we present an extension of our PDLA framework onto online covering with ℓ_q-norm objectives. In Appendix D, we apply our algorithmic frameworks onto a variety of well-motivated practical problems. We conclude our paper with closing remarks and discussions of future directions in Section 4.

1.1 Preliminaries

We denote by $\mathbf{0}_{(n)}$ and $\mathbf{1}_{(n)}$ the vector of all zeroes and all ones of length n, respectively. When the length of the vector is unambiguous, we omit the subscript and use $\mathbf{0}$ and $\mathbf{1}$. We use $x \geq y$ for $x, y \in \mathbb{R}^n_{\geq 0}$ to denote the relation that $x_i \geq y_i$ for all $i \in [n]$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is monotone if for all $x \geq x'$, $f(x) \geq f(x')$ as well[1].

[1]Such functions are usually called 'monotone non-decreasing'. Since all monotone functions in this paper are non-decreasing, we omit the classifier for simplicity.

We use $A := \{a_{ij}\}_{i\in[m],j\in[n]} \in \mathbb{R}_{\geq 0}^{m\times n}$ to denote the constraint matrix of the covering and packing problems we study, and we use $i \in [m]$ and $j \in [n]$ to denote the row and column index of A, respectively.

Online concave packing. The online packing setting with concave objectives we study is defined as follows:

$$\max\ g(y) \text{ over } y \in \mathbb{R}_{\geq 0}^{m} \text{ subject to } A^T y \leq b. \quad (1)$$

Here, $g : \mathbb{R}_{\geq 0}^{m} \to \mathbb{R}_{\geq 0}$ is a concave and monotone objective function with $g(\mathbf{0}) = 0$, and $b \in \mathbb{R}_{>0}^{n}$ denotes an upper bound vector for the linear constraints.

In the online setting, the objective function g, the values in b, and the number of constraints n are given in advance. In each round $i \in [m]$, a new packing variable y_i is introduced, along with all the associated coefficients a_{ij} for all $j \in [n]$ (the i-th row of A). The number of packing variables m might be unknown, and each packing constraint is gradually revealed column-wise to the algorithm. In round $i \in [m]$, the algorithm can only irrevocably assign a value for y_i. The goal is to approximately maximize $g(y)$ by assigning values to the variables online while maintaining (approximate) feasibility.

Online convex covering. The online covering setting with convex objectives we study is defined as follows:

$$\min\ f(x) \text{ over } x \in \mathbb{R}_{\geq 0}^{n} \text{ subject to } Ax \geq \mathbf{1}. \quad (2)$$

Here, $f : \mathbb{R}_{\geq 0}^{n} \to \mathbb{R}_{\geq 0}$ is a convex, monotone, and *differentiable* function, with $f(\mathbf{0}) = 0$. We make the technical assumption that the gradient of f, ∇f, is monotone as well, and later show that it is possible to remove this assumption in a more structured setting.

The online setting for convex covering is similar to that for concave packing: The objective function f and the numbers of covering variables n are given in advance, but the constraint matrix A and the number of constraints m is unknown to the algorithm. In each round $i \in [m]$, the i-th row of A arrives online, and the algorithm must update x_j for all $j \in [n]$ in a non-decreasing fashion to satisfy the constraint, while approximately minimizing $f(x)$.

Online optimization with advice. In the learning-augmented setting, the algorithm is additionally given an advice on the variables of interest: $y' \in \mathbb{R}_{\geq 0}^{m}$ for packing, and $x' \in \mathbb{R}_{\geq 0}^{n}$ for covering. We do not make any additional assumptions about the advice, and their objective values $g(y')$ or $f(x')$ can be arbitrarily worse compared to the optimal solution. Our frameworks additionally utilize a hyper-parameter $\lambda \in [0, 1]$, denoted the *confidence parameter*, chosen by the user a priori.

The advice can be interpreted as a suggestion on what the solution, and thus what the main variables at the end of the algorithm's execution should be, and the confidence parameter λ indicates how much the user trusts this advice. A value of λ close to 0 indicates that the advice is trusted and thus likely to be accurate, while a value close to 1 indicates that the advice should not be trusted. The algorithm's goal is thus to incorporate the advice prudently, obtaining a final solution $\bar{x}$ or $\bar{y}$ whose performance is comparable to the advice when λ is small, and comparable to the state-of-the-art online algorithm when λ is large.

We formalize the metrics we use to measure the performance of learning-augmented algorithms via two notions, *consistency* and *robustness*:

Definition 1.1. An online (covering) solution $\bar{x}$ is $C(\lambda)$-consistent if $f(\bar{x}) \leq C(\lambda) \cdot f(x')$, where $f(x')$ is the cost of the advice. A learning-augmented algorithm is $C(\lambda)$-consistent if the solution it generates is $C(\lambda)$-consistent.

Definition 1.2. An online (covering) solution $\bar{x}$ is $R(\lambda)$-robust if $f(\bar{x}) \leq R(\lambda) \cdot \mathsf{OPT}$, where OPT is the cost of the optimal offline solution. A learning-augmented algorithm is $R(\lambda)$-robust if the solution it generates is $R(\lambda)$-robust.

The packing version of these definitions follows symmetrically: A solution $\bar{y}$ is C-consistent and R-robust if $g(\bar{y}) \geq \frac{1}{C} g(y')$ and $g(\bar{y}) \geq \frac{1}{R}\mathsf{OPT}$. For packing maximization problems, it is common practice to allow the solutions found by an online algorithm to violate the packing constraints by a certain factor; an exactly feasible solution can be recovered by scaling down the approximately feasible solution. We make this notion of approximate feasibility explicit in the following definition:

Definition 1.3. An online (packing) solution $\bar{y}$ is $V(\lambda)$-feasible if $A^T \bar{y} \leq V(\lambda) \cdot b$. An online algorithm is $V(\lambda)$-feasible if the solution it generates is $V(\lambda)$-feasible.

Intuitively, when we are confident in the advice, we should follow it as much as possible and obtain a solution that is close to the advice, so $C(\lambda)$ should tend to 1 as λ tends to 0. On the other hand, when the advice is possibly inaccurate, our algorithm should follow the non-learning-augmented classical online al-

gorithm, so $R(\lambda)$ should tend to the competitive ratio of the state-of-the-art online algorithm as λ tends to 1. The role of the confidence hyper-parameter λ is thus to control the *tradeoff* between consistency and robustness.

1.2 Our Contributions

In this paper, we devise and analyze learning-augmented algorithmic frameworks for online packing problems with concave objectives, and online covering problems with convex objectives. We note that our algorithms solve for fractional solutions. While many specific problems require integral solutions, these integrality constraints are not in scope of the general-purpose settings we study. The user can apply the appropriate rounding schemes of their interest to the fractional solutions of our frameworks independently.

1.2.1 Online Packing with Concave Objectives

Our result for the online packing problem with concave objectives is a general framework for devising learning-augmented algorithms for all online concave packing problems. We present a simple algorithm utilizing a switching strategy, which is reminiscent of the switching algorithm mentioned in Grigorescu et al. (2022b), but ultimately relies on properties of maximization problems distinct to that for covering minimization problems. The algorithm uses any state-of-the-art classical online algorithm as a black-box subroutine, and obtains a solution that matches both the value of the advice and the value of the subroutine online algorithm asymptotically:

Theorem 1.4 (Informal)**.** *Given an instance of the online concave packing problem (1), there exists an online algorithm that takes an α-competitive β-feasible online algorithm as a subroutine, an advice $y' \in \mathbb{R}^n_{\geq 0}$ which is β-feasible, and a confidence parameter λ. The algorithm is $\frac{1}{1-\lambda}$-consistent, $\frac{\alpha}{\lambda}$-robust, and $(2-\lambda)\beta$-feasible.*

The formal version of Theorem 1.4 is Theorem 2.1. We state and analyze our algorithm in Section 2.

We remark that our algorithm is a general-purpose framework, and does not rely on any specific classical online algorithm. This property absolves users of our framework from the responsibility of understanding potentially sophisticated online algorithm used as a black-box subroutine. In addition, future advancements in the field of classical online problems would immediately imply advances in their variants augmented by advice.

Our advice model on the packing variables also generalizes many prior models (e.g., Im et al., 2021, for online knapsack problems), since any form of advice that suggests a course of action can be simulated by our packing program formulation by setting the advice entries to appropriate values. The exact values of the advice entries depend on the exact problem and advice formulation: We give some examples of these reductions in Appendix D.1.

1.2.2 Online Covering with Convex Objectives

As our main result, we present a general framework for designing learning-augmented algorithms for online covering problems with convex objective functions (2) and possibly non-integral advice and constraints, which generalizes previous works such as online set cover (Bamas et al., 2020b), and online general covering LPs (Grigorescu et al., 2022a). We employ the standard assumption that the objective function f is convex, monotone, differentiable, with $f(\mathbf{0}) = 0$. We additionally assume that the gradient ∇f is monotone, which is a technical assumption made in works studying the classical online version of the problem as well (Buchbinder et al., 2014; Azar et al., 2016).

We point out that the switching strategy for online covering linear programs in Grigorescu et al. (2022b) crucially relies on the linearity (more specifically, subadditivity) of the objective function, which is not satisfied by convex functions in general, even with the monotone gradient assumption. The cost of "switching" between solutions is at most a multiplicative factor of 2 in the linear case, but may be unbounded for convex objectives. Thus, such simple switching strategies do not apply to our setting.

Motivated by this, our algorithm for online convex covering returns to more sophisticated methods and follows the primal-dual learning-augmented (PDLA) paradigm (Bamas et al., 2020b; Grigorescu et al., 2022a), taking the advice into consideration while carefully tuning the growth rate of each variable, in order to approximately minimize the cost of the solution. The PDLA algorithms we present are both robust and consistent, with performance close to that of the optimal offline solution when the advice is accurate, and close to that of the state-of-the-art online algorithm when the advice is inaccurate. The performance of our algorithm is (informally) characterized by the following theorem:

Theorem 1.5 (Informal)**.** *Given an instance of the*

online convex covering problem (2), there exists an online algorithm that takes an advice $x' \in \mathbb{R}^n_{\geq 0}$ and a confidence parameter λ. The algorithm is $O(\frac{1}{1-\lambda})$-consistent and $O((p \log \frac{d}{\lambda})^p)$-robust. Here, $p := \sup_{x \geq \mathbf{0}} \frac{\langle x, \nabla f(x) \rangle}{f(x)}$, and d is the row sparsity of the constraint matrix.

The formal version of Theorem 1.5 is Theorem 3.2 in Section 3, which addresses the case when the advice x' is infeasible. We remark that our consistency ratio tends to 1 as λ tends to 0, and our robustness ratio tends to $O((p \log d)^p)$ as λ tends to 1, matching the competitiveness of the online algorithm presented in Azar et al. (2016), meeting the ideal expectations. While our algorithms and analyses resemble their counterparts in prior works, we employ some subtle yet vital modifications, since a more direct application of Grigorescu et al. (2022a)'s learning-augmentation model onto Azar et al. (2016)'s primal-dual framework suffers from various technical issues and cannot yield satisfactory results.

1.2.3 Online Covering with ℓ_q-norm Objectives

Our technical assumption that the gradient is monotone follows from an identical assumption in Buchbinder et al. (2014) and Azar et al. (2016). Subsequently, Shen and Nagarajan (2020) restricted their attention to online covering problems with ℓ_q-norm objectives, and presented new analyses of a variant of Azar et al. (2016)'s algorithm that removed the monotone gradient assumption, using the structural properties of ℓ_q-norms. Specifically, they consider covering problems of the following form:

$$\min \sum_{e=1}^{r} c_e ||x(S_e)||_{q_e} \text{ over } x \in \mathbb{R}^n_{\geq 0} \text{ subject to } Ax \geq 1 \tag{3}$$

where each $S_e \subseteq [n]$ is a subset of indices, $q_e \geq 1$, $c_e \geq 0$, $x(S)$ denote the vector x restricted to indices in S, and $||x(S)||_q$ is the ℓ_q-norm, i.e., $\left(\sum_{i \in S} x_i^q\right)^{1/q}$.

We adapt the analysis in Shen and Nagarajan (2020) to our PDLA framework for general online convex covering problems, and show that it is also consistent and robust for online covering problems with ℓ_q-norm objectives, similarly in lieu of the monotone gradient assumptions in Azar et al. (2016). The performance of our algorithm for online covering problems with ℓ_q-norm objectives is (informally) described by the following theorem:

Theorem 1.6 (Informal). *Given an instance of the online covering problem with ℓ_q-norm objectives (3), there exists an online algorithm that takes an advice $x' \in \mathbb{R}^n_{\geq 0}$ and a confidence parameter λ. The algorithm is $O(\frac{1}{1-\lambda})$-consistent and $O(\log \frac{\kappa d}{\lambda})$-robust. Here, κ is the condition number of the constraint matrix, and d is the row sparsity of the constraint matrix.*

The formal version of Theorem 1.6 is Theorem C.1 in Appendix C which addresses the case when the advice x' is infeasible.

1.2.4 When is switching optimal?

Designing simple solutions to natural problems of wide general interest is an ultimate goal of both theory (as they can be taught even in undergraduate courses!) and practice (as they can be implemented with a few lines of code, and would have strong provable guarantees!). However, while the study of learning-augmented algorithm is currently flourishing, many solutions and formulations are often somewhat ad-hoc, and hence there is a need for general frameworks, techniques, and paradigms.

Inspired by the switching strategy noted in Grigorescu et al. (2022b) and our own observations in this paper, we raise a basic question:

Can one characterize the space of online problems augmented with advice, for which one may use classical online algorithms as a black-box to obtain optimal solutions?

In particular, what are the necessary and sufficient conditions for such simple switching strategies to be close to being optimal? What features should a problem exhibit that would allow it to be solvable in a black-box fashion in the advice setting? Recently, Daneshvaramoli et al. (2024) showed that a similar switching-based approach can achieve optimal constants in online knapsack with succinct predictions, which also raises similar questions. We believe that a better understanding of this conceptual direction may lead to unifying frameworks in the study of algorithms with advice.

In this work, we make progress on finding the sufficient conditions for the question above. In particular, we study online concave packing problems with advice and show that the switching framework is close to being optimal in this context. As a step towards necessary conditions, we show that switching strategies, at least ones noted in Grigorescu et al. (2022b) and our algorithmic framework for online concave packing problems, are not applicable to on-

line convex covering problems, which instead require more sophisticated methods to solve in general.

1.2.5 Applications

We explicitly study the application of our frameworks to well-motivated problems in both online concave packing and online convex covering settings in Appendix D. Our algorithms match the state-of-the-art algorithms by setting $\lambda = 1$ and can potentially outperform the best-known online algorithms when the advice is accurate.

We apply Theorem 1.4 to a variety of packing problems, including knapsack, resource management benefit, throughput maximization, network utility maximization, and optimization with inventory constraints. We then present an application of Theorem 1.5 to online mixed covering and packing, and by extension a variety of sub-problems such as capacity-constrained facility location and capacitated multicast, and an application of Theorem 1.6 to online buy-at-bulk network design.

2 SWITCHING ALGORITHMS FOR ONLINE CONCAVE PACKING

We present a simple, possibly folklore, learning-augmented online algorithm for concave packing. Let $\mathcal{O}$ be an α-competitive β-feasible online algorithm for the packing problem (1), for some $\alpha, \beta \geq 1$.

The algorithm keeps track of two candidate solutions: the advice y', and $y^{\mathcal{O}}$ from the non-learning-augmented online algorithm $\mathcal{O}$, and combines them. Whenever a row of A arrives online, the algorithm obtains a solution from both the online algorithm and the advice and sets y_i to a weighted interpolation between the two values, $\lambda \cdot y_i^{\mathcal{O}} + (1-\lambda) \cdot y'_i$. Once the algorithm finds out that the advice violates any constraint by a factor of β, it discards the advice for this round and follows the online algorithm only. The algorithm is presented in Algorithm 1.

We formally characterize the performance of Algorithm 1 in Theorem 2.1, presented below:

Theorem 2.1. *For the learning-augmented online concave packing problem, there exists an online algorithm that takes an α-competitive β-feasible online algorithm $\mathcal{O}$, an advice $y' \in \mathbb{R}^n_{\geq 0}$, and a confidence parameter $\lambda \in [0, 1]$, and generates a solution $\bar{y}$ satisfying $A^T \bar{y} \leq (2 - \lambda)\beta b$ such that $g(\bar{y}) \geq \frac{\lambda}{\alpha} \mathsf{OPT}$. Additionally, if y' is β-feasible, i.e., $A^T y' \leq \beta b$, then $g(\bar{y}) \geq (1 - \lambda) g(y')$.*

Algorithm 1 A Simple Learning-Augmented Online Algorithm for the Packing Problem (1)

Input: Any online algorithm algorithm $\mathcal{O}$, advice y', confidence parameter λ.
Output: The online solution y.

```
for i = 1, 2, ... do
    Update A by adding a new row i.
    Run O for round i and obtain y_i^O.
    if A^T y' ≤ βb then
        y_i ← λ y_i^O + (1 − λ) y'_i.
    else
        y_i ← y_i^O.
```

Proof. Denote by y'_{trim} as the advice y' with all entries discarded by Algorithm 1 set to 0 instead. The final packing solution $\bar{y}$ is at least $\lambda y^{\mathcal{O}} + (1-\lambda) y'_{trim}$. By the monotonicity of g, we have also $g(\bar{y}) \geq g(\lambda y^{\mathcal{O}} + (1 - \lambda) y'_{trim})$. It immediately holds that $g(\bar{y}) \geq g(\lambda y^{\mathcal{O}}) \geq \lambda \cdot g(y^{\mathcal{O}}) \geq \frac{\lambda}{\alpha} \mathsf{OPT}$, since g is concave and $g(\mathbf{0}) = 0$.

If y' is β-feasible, the algorithm will not trim any entry of y', and it similarly holds that $g(\bar{y}) \geq (1 - \lambda) g(y'_{trim}) = (1 - \lambda) g(y')$.

For feasibility, observe that $A^T y'_{trim} \leq \beta b$ and $A^T y^{\mathcal{O}} \leq \beta b$. Thus, $A^T \bar{y} \leq A^T (1-\lambda) y'_{trim} + A^T y^{\mathcal{O}} \leq (2 - \lambda)\beta b$, as desired. □

Remark 2.2. We note that depending on the problem of interest, β might not be a fixed parameter. For online packing linear programs, β is a conditional number non-decreasing over time, which depends on A, so it is not necessarily the case that the advice is only used in earlier rounds. See Appendix D.1 for a more detailed discussion.

3 PRIMAL-DUAL FRAMEWORK FOR ONLINE CONVEX COVERING

In this section we briefly outline a consistent and robust primal-dual learning-augmented (PDLA) algorithm for online covering problems with convex objectives, based on the primal-dual framework of Buchbinder and Naor (2009a) and its extension to the learning-augmented setting in Grigorescu et al. (2022a). In the classical online setting, online convex covering problems are studied in Buchbinder et al. (2014) and subsequently Azar et al. (2016), in which the authors presented an extension of Buchbinder and Naor (2009a)'s primal-dual framework to convex non-linear objective functions; our PDLA frame-

works take inspiration from their work as well.

The primal-dual method is a class of algorithms that simultaneously consider the dual packing program. Specifically, we choose the Fenchel dual program formulated as:

$$\max \sum_{i=1}^{m} y_i - f^*(\mu) \text{ over } y \in \mathbb{R}^m_{\geq 0} \text{ subject to } y^T A \leq \mu^T \tag{4}$$

where $f^* : \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ is the Fenchel dual of f, defined as $f^*(\mu) = \sup_z \{\mu^T z - f(z)\}$. f^* is always convex; for more properties of Fenchel dual functions, we refer the readers to Borwein and Lewis (2006).

As with covering and packing linear programs, the convex covering program and its Fenchel dual also exhibits a form of duality:

Fact 3.1 (Buchbinder et al. (2014)). *Let x and (y, μ) be any feasible solution to the primal covering program (2) and the dual packing program (4), respectively. Then*

$$f(x) \geq \sum_{i=1}^{m} y_i - f^*(\mu)$$

Our algorithm maintains a variable τ denoting the continuous flow of time, and upon the arrival of each constraint t, increment each covering variable x_j for $j \in [n]$ with differing rates, dependent on the advice x', until the constraint is satisfied. We increment the dual packing variable y_t as well, but also potentially decrement some other packing variable to ensure that the packing solution is feasible.

To paint a picture of our strategy in more detail, let A_i denote the i-th row of A. In round t, upon the arrival of constraint t and row A_t, we check if the advice is feasible for this constraint. If the advice is feasible, i.e., $A_t x' \geq 1$, we increment each variable x_j with rate

$$\frac{a_{tj}}{\nabla_j f(x^{(\tau)})} \left(x_j^{(\tau)} + \frac{\lambda}{a_{tj} d} + \frac{(1-\lambda) x'_j \mathbf{1}_{x_j^{(\tau)} < x'_j}}{A_t x'_c} \right)$$

where $x_j^{(\tau)}$ and $x^{(\tau)}$ denote the value of x_j and x at time τ, respectively, and $d = \max_{i \in [m]} |\{a_{ij} | a_{ij} > 0\}|$ is the row sparsity of A. x'_c is the advice restricted to entries in which the corresponding variables has not reached the advice yet. Equivalently, the j-th entry of x'_c is equal to the j-th entry of x' if $x_j^{(\tau)} < x'_j$, and 0 otherwise. $\mathbf{1}_{x_j^{(\tau)} < x'_j}$ is the indicator variable with value 1 if $x_j^{(\tau)} < x'_j$ and 0 otherwise.

Intuitively, the two additive terms, $\frac{\lambda}{a_{tj} d}$ and $\frac{(1-\lambda) x'_j \mathbf{1}_{x_j^{(\tau)} < x'_j}}{A_t x'_c}$, correspond to the contribution of the classical online algorithm and the advice to the growth rate, respectively. Our algorithm applies the additional growth rate attributed to the advice if the corresponding variable x_j has not reached the value suggested by the advice x'_j yet. If x_j has reached the suggested value, our algorithm instead zeroes the additional growth rate, and relies only on the component attributed to the online primal-dual method.

We remark that this form of the growth rate of our primal variables x_j is reoccurring in existing literature, including Grigorescu et al. (2022a) and Azar et al. (2016). The vital difference between our frameworks and those of prior works is the "normalizing factors", the denominators in the additive terms to normalize the contribution of the coefficients of the problem instances to the growth rate. For the classical online term, Grigorescu et al. (2022a) adopts $\frac{\lambda}{A_t \mathbf{1}}$, using the sum of all entries of the arriving constraint as the normalizing factor, which naturally extends into the matching denominator for the advice term, $A_t x'_c$, which allows for a clean normalization of the advice term. This choice, however, implicitly introduced a dependency: the growth rate of each primal variable x_j will depend on the coefficients of the constraint matrix, $a_{tj'}$, even for $j' \neq j$, i.e., corresponding to a different primal variable $x_{j'}$. On the other hand, the choice of Azar et al. (2016) for the classical online term is $a_{tj} d$, which introduced more variance, but the independence between the growth rate of each primal variable allowed for a tighter analysis.

While applying the choice of Azar et al. (2016) onto both additive terms directly does not admit a proper normalization of the advice term, we show that it is in fact possible to navigate the delicate paths connecting these two prior frameworks together, to apply the normalizing factor of Azar et al. (2016) onto the classical online term to obtain the desired level of consistency and robustness, while keeping the choice of Grigorescu et al. (2022a) for the advice term to allow for a clean analysis.

If the advice is not feasible, i.e., $A_t x' < 1$, we disregard the advice temporarily, and increment x_j with rate

$$\frac{a_{tj}}{\nabla_j f(x^{(\tau)})} \left(x_j^{(\tau)} + \frac{1}{a_{tj} d} \right)$$

In this case, our rate of growth coincides with that of both the classical, non-learning-augmented online primal-dual method, and the case when the advice

is feasible above with λ set to 1. Since the advice is infeasible and cannot provide valuable insight, we consider only the contribution from the online algorithm for constraint t.

The increment scheme for the dual variables is as follows. In round t, we initialize the packing variable y_t corresponding to the arriving constraint t to 0, and increment it with rate

$$r := \frac{\partial y_t^{(\tau)}}{\partial \tau} = \frac{\delta}{\log(1 + \frac{2}{\lambda}d^2)}$$

for some value of δ we choose later that depends only on p, d, and λ.

Combining the ideas described above suffices to yield an $O(\log \frac{\kappa d}{\lambda})$-robust learning-augmented algorithm, where κ is the condition number of the constraint matrix A, defined as the ratio between the largest and the smallest non-zero entry in A. However, since the entries of A arrives online, κ is unknown to the algorithm up front and can be arbitrarily large. We follow the idea of Buchbinder et al. (2014) and Azar et al. (2016) to additionally decrease some of the packing variables, to ensure that each dual packing constraint in (4) is exactly satisfied. Specifically, for each dual packing constraint j (corresponding to the covering variable x_j) that is tight, i.e., $\sum_{i=1}^{t} a_{ij} y_i = \mu_j$, we identify the variable $y_{m_j^*}$ with the largest coefficient $a_{(m_j^*)j}$, and decrement it with rate

$$\frac{\partial y_{m_j^*}}{\partial \tau} = -\frac{a_{tj}}{a_{(m_j^*)j}} \cdot r$$

This ensures that the net growth rate of $\sum_{i=1}^{t} a_{ij} y_i$ is 0, so that the constraint is not violated.

Note that the technique of decreasing the dual variables y is unusual, in that it violates the monotonicity of the dual variables, which is a common feature of primal-dual methods, such as in Buchbinder et al. (2007); Bamas et al. (2020b); Grigorescu et al. (2022a). The purpose of this property is for the primal-dual method to be a general-purpose method that solves both covering programs and packing programs simultaneously. However, in our context, we are exclusively concerned with the online convex covering problem, and not its dual packing problem. In fact, learning-augmented algorithms for covering and packing problems that are usually dual to each other often have differing specification and advice forms, and do not share simple and elegant algorithms that serve both problems.

For the dual variables μ, we set μ to $\delta \nabla f(x)$, which is monotone due to the monotonicity of x and the assumption that $\nabla f(x)$ is monotone as well.

We present our learning-augmented primal-dual algorithm for online convex covering problems during round t in Algorithm 2. While the algorithm description increments x and y, and sets μ simultaneously during execution, the update rule for the primal covering variables, x, in fact does not depend on y or μ. Since we are not concerned about maintaining the dual solution, (y, μ), and use it for purely analytical purposes, we can instead increment y and set $\mu = \delta \nabla f(\bar{x})$ in hindsight after computing the final solution $\bar{x}$ to the primal covering problem 2, simplifying the algorithm and its analysis.

Algorithm 2 PDLA algorithm for online convex covering problems in round t

Input: $x_1, \cdots, x_n$, $y_1, \cdots, y_{t-1}$, μ: current solution, A_{tj} for $1 \le j \le n$: current coefficients, (x', λ): advice and confidence parameter.
Output: Updated x and (y, μ).

$\mu \leftarrow \delta \nabla f(\bar{x})$.
$y_t \leftarrow 0$.
$\tau \leftarrow$ current time.
if $A_t x' \ge 1$ **then** ▷ if x' is feasible
    $D_j^{(t)} \leftarrow \frac{\lambda}{a_{tj} d} + \frac{(1-\lambda) x'_j \mathbf{1}_{x_j < x'_j}}{A_i x'_c}$.
else
    $D_j^{(t)} \leftarrow \frac{1}{a_{tj} d}$.
while $A_t x < 1$ **do** ▷ while unsatisfied
    Increase τ at rate 1.
    for each $j \in [n]$ s.t. $a_{tj} > 0$ **do**
        Increase x_j at rate

$$\frac{\partial x_j}{\partial \tau} = \frac{a_{tj}}{\nabla_j f(x)}(x_j + D_j^{(t)})$$

        Increase y_t at rate

$$r := \frac{\partial y_t}{\partial \tau} = \frac{\delta}{\log(1 + \frac{2}{\lambda}d^2)}$$

    for each $j \in [n]$ s.t. $\sum_{i=1}^{t} a_{ij} y_i = \mu_j$ **do**
        $m_j^* = \arg\max_{i=1}^{t} \{a_{ij} | y_i > 0\}$.
        Decrease $y_{m_j^*}$ at rate $\frac{a_{tj}}{a_{(m_j^*)j}} \cdot r$.

While the algorithm increments the variables in a continuous fashion, it is possible to be implemented discretely, up to any desired precision, via binary searches.

We state the performance of Algorithm 2 in Theorem 3.2, presented below:

Theorem 3.2. *For the learning-augmented online*

convex covering problem with the monotone gradient assumption, there exists an online algorithm that takes a problem instance and an advice x', and generates a solution $\bar{x}$ such that

$$f(\bar{x}) \leq \min\left\{O\left(\frac{1}{1-\lambda}\right) f(x') + O((p\log d)^p)\mathit{OPT}, \right.$$
$$\left. O\left(\left(p\log\frac{d}{\lambda}\right)^p\right)\mathit{OPT}\right\}$$

Additionally, if x' is feasible, i.e., $Ax' \geq \mathbf{1}$, then $f(\bar{x}) \leq O\left(\frac{1}{1-\lambda}\right) f(x')$. Here, $d = \max_{i\in[m]} |\{a_{ij}|a_{ij} > 0\}|$ is the row sparsity of the constraint matrix A, and $p := \sup_{x\geq\mathbf{0}} \frac{\langle x,\nabla f(x)\rangle}{f(x)}$.

Proof sketch. We briefly outline our strategy here, and present a full proof in Appendix B. To prove Theorem 3.2, it suffices to prove the two components, $f(\bar{x}) \leq O((p\log\frac{d}{\lambda})^p)\mathsf{OPT}$ (robustness) and $f(\bar{x}) \leq O(\frac{1}{1-\lambda})f(x') + O((p\log d)^p)\mathsf{OPT}$ (consistency), separately.

Towards robustness, we use the classical strategy of showing that both the covering and the packing solutions are feasible, and that the growth rate between the objectives are appropriately bounded. We conclude using the weak duality of Fact 3.1 to associate the covering objective value with the optimal value.

Towards consistency, we follow the strategy of Grigorescu et al. (2022a) and partition the growth rate, $\frac{\partial f(x^{(\tau)})}{\partial\tau}$, into two parts, one attributed to the advice, and the other attributed to the online algorithm. With the observation that integrating the advice component of the growth rate over time recovers the value of the advice, $f(x')$, we conclude by bounding the online component with $O(\frac{1}{1-\lambda})$ times the advice component at all times. □

Remark 3.3. The consistency guarantees in Theorem 3.2 may seem extremely brittle at first glance, reversing back to the competitiveness of the classical online algorithm, $O((p\log d)^p)$, even if the advice is slightly infeasible. We remark that our framework utilizes the advice as much as possible when it remains feasible, and with a more fine-grained analysis, we can obtain an improved consistency bound of

$$f(\bar{x}) \leq O\left(\frac{1}{1-\lambda}\right) + O((p\log d)^p)\mathsf{OPT}_u$$

where OPT_u denotes the optimal cost to only cover the iterations in which the advice is infeasible. Note that without any assumptions on the quality of the advice, OPT_u can be as large as OPT; Thus we choose to omit this fine-grained bound in favor of simplicity and generality.

4 CONCLUSION

In this paper we present two unifying learning-augmented algorithmic frameworks: A switching-based simple framework for online concave packing problems, using any state-of-the-art online algorithm as a black-box, and a primal-dual framework for online convex covering problems based on the ideas of Grigorescu et al. (2022b) and Azar et al. (2016).

As we show in Appendix C, our PDLA framework for convex covering is adaptable to specific forms of objective functions, and can obtain better performance and analysis. It would be exciting to see other settings for which our framework can be applied to. Removing the technical assumption that the gradient is monotone is an interesting direction as well. While the selection of our hyper-parameter λ is not usually in scope of the field of learning-augmented algorithms, future work on how to accurately estimate λ or lessen our algorithm's dependence on it will complement our results nicely as well.

We also raised the conceptual question of characterizing online problems with advice that can utilize classical online algorithms as black-boxes to obtain optimal performances. We believe that while meta-questions like these are hard to formulate and study, they are vital to furthering our understanding of the fundamental power of learning-augmentation, and we would like to invite more researchers to explore these directions.

5 ACKNOWLEDGEMENT

We are grateful to Roie Levin, Kent Quanrud, Sandeep Silwal, Paul Valiant, Samson Zhou, and anonymous reviewers for helpful discussions. Elena Grigorescu was supported in part by NSF CCF-2228814. Maoyuan Song is supported in part by NSF CCF-2127806 and NSF CCF-2228814.

References

Alon, N., Awerbuch, B., Azar, Y., Buchbinder, N., and Naor, J. (2006). A general approach to online network optimization problems. ACM Transactions on Algorithms (TALG), 2(4):640–660. 14, 27

Alon, N., Awerbuch, B., Azar, Y., Buchbinder, N., and Naor, J. (2009). The online set cover problem. SIAM Journal on Computing, 39(2):361–370. 1, 14

Anand, K., Ge, R., Kumar, A., and Panigrahi, D. (2022). Online algorithms with multiple predictions. In International Conference on Machine Learning, pages 582–598. PMLR. 14

Antonakopoulos, S. (2010). Approximating directed buy-at-bulk network design. In International Workshop on Approximation and Online Algorithms, pages 13–24. Springer. 38

Antoniadis, A., Coester, C., Eliáš, M., Polak, A., and Simon, B. (2023). Online metric algorithms with untrusted predictions. ACM transactions on algorithms, 19(2):1–34. 2

Antoniadis, A., Gouleakis, T., Kleer, P., and Kolev, P. (2020). Secretary and online matching problems with machine learned advice. Advances in Neural Information Processing Systems, 33:7933–7944. 14

Awerbuch, B., Azar, Y., and Plotkin, S. (1993). Throughput-competitive on-line routing. In Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science, pages 32–40. IEEE. 34

Azar, Y., Buchbinder, N., Chan, T. H., Chen, S., Cohen, I. R., Gupta, A., Huang, Z., Kang, N., Nagarajan, V., Naor, J., et al. (2016). Online algorithms for covering and packing problems with convex objectives. In 2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS), pages 148–157. IEEE. 2, 4, 5, 6, 7, 8, 9, 19, 21

Azar, Y., Panigrahi, D., and Touitou, N. (2022). Online graph algorithms with predictions. In Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 35–66. SIAM. 14

Balcan, M. (2020). Data-driven algorithm design. CoRR, abs/2011.07177. 14

Balkanski, E., Perivier, N., Stein, C., and Wei, H.-T. (2024). Energy-efficient scheduling with predictions. Advances in Neural Information Processing Systems, 36. 14

Balseiro, S. R. and Gur, Y. (2019). Learning in repeated auctions with budgets: Regret minimization and equilibrium. Management Science, 65(9):3952–3968. 1

Bamas, É., Maggiori, A., Rohwedder, L., and Svensson, O. (2020a). Learning augmented energy minimization via speed scaling. In Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, NeurIPS. 14

Bamas, E., Maggiori, A., and Svensson, O. (2020b). The primal-dual method for learning augmented algorithms. Advances in Neural Information Processing Systems, 33:20083–20094. 1, 2, 4, 8, 14, 20

Banerjee, S., Cohen-Addad, V., Gupta, A., and Li, Z. (2023). Graph searching with predictions. In 14th Innovations in Theoretical Computer Science Conference (ITCS 2023). Schloss-Dagstuhl-Leibniz Zentrum für Informatik. 14

Borwein, J. and Lewis, A. (2006). Convex Analysis. Springer. 7

Boyar, J., Favrholdt, L. M., and Larsen, K. S. (2022). Online unit profit knapsack with untrusted predictions. In 18th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2022). Schloss-Dagstuhl-Leibniz Zentrum für Informatik. 14, 33

Brand, J. v. d., Forster, S., Nazari, Y., and Polak, A. (2024). On dynamic graph algorithms with predictions. In Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 3534–3557. SIAM. 14

Buchbinder, N., Chen, S., Gupta, A., Nagarajan, V., et al. (2014). Online packing and covering framework with convex objectives. arXiv preprint arXiv:1412.8347. 4, 5, 6, 7, 8, 15, 21, 22, 26, 35

Buchbinder, N., Jain, K., and Naor, J. S. (2007). Online primal-dual algorithms for maximizing ad-auctions revenue. In European Symposium on Algorithms, pages 253–264. Springer. 8

Buchbinder, N. and Naor, J. (2006). Improved bounds for online routing and packing via a primal-dual approach. In 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), pages 293–304. IEEE. 34

Buchbinder, N. and Naor, J. (2009a). Online primal-dual algorithms for covering and packing. Mathematics of Operations Research, 34(2):270–286. 2, 6, 14, 32

Buchbinder, N. and Naor, J. S. (2009b). The design of competitive online algorithms via a primal–dual approach. Foundations and Trends® in Theoretical Computer Science, 3(2–3):93–263. 1, 14

Cao, Y., Sun, B., and Tsang, D. H. (2022). Online network utility maximization: Algorithm, competitive analysis, and applications. IEEE Transactions on Control of Network Systems, 10(1):274–284. 34

Chakrabarty, D., Ene, A., Krishnaswamy, R., and Panigrahi, D. (2018). Online buy-at-bulk network design. SIAM J. Comput., 47(4):1505–1528. 37, 38

Chekuri, C., Hajiaghayi, M. T., Kortsarz, G., and Salavatipour, M. R. (2010). Approximation algorithms for nonuniform buy-at-bulk network design. SIAM Journal on Computing, 39(5):1772–1798. 38

Chen, J. Y., Eden, T., Indyk, P., Lin, H., Narayanan, S., Rubinfeld, R., Silwal, S., Wagner, T., Woodruff, D. P., and Zhang, M. (2022a). Triangle and four cycle counting with predictions in graph streams. In The Tenth International Conference on Learning Representations, ICLR. 14

Chen, J. Y., Silwal, S., Vakilian, A., and Zhang, F. (2022b). Faster fundamental graph algorithms via learned predictions. In International Conference on Machine Learning, ICML, pages 3583–3602. 14

Cohen, I. R. and Panigrahi, D. (2023). A general framework for learning-augmented online allocation. In 50th International Colloquium on Automata, Languages, and Programming (ICALP 2023). Schloss-Dagstuhl-Leibniz Zentrum für Informatik. 14

Daneshvaramoli, M., Karisani, H., Lechowicz, A., Sun, B., Musco, C., and Hajiesmaili, M. (2024). Competitive algorithms for online knapsack with succinct predictions. 5

Daneshvaramoli, M., Karisani, H., Lechowicz, A., Sun, B., Musco, C. N., and Hajiesmaili, M. (2023). Online fractional knapsack with predictions. 14, 33

Diakonikolas, I., Kontonis, V., Tzamos, C., Vakilian, A., and Zarifis, N. (2021). Learning online algorithms with distributional advice. In Proceedings of the 38th International Conference on Machine Learning, ICML, pages 2687–2696. 14

Dinitz, M., Im, S., Lavastida, T., Moseley, B., and Vassilvitskii, S. (2021). Faster matchings via learned duals. In Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems, NeurIPS, pages 10393–10406. 14

Dütting, P., Lattanzi, S., Paes Leme, R., and Vassilvitskii, S. (2021). Secretaries with advice. In Proceedings of the 22nd ACM Conference on Economics and Computation, pages 409–429. 14

Elad, N., Kale, S., and Naor, J. S. (2016). Online semidefinite programming. In 43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2, 14

Ergun, J. C., Feng, Z., Silwal, S., Woodruff, D. P., and Zhou, S. (2022). Learning-augmented k-means clustering. In The Tenth International Conference on Learning Representations, ICLR. 14

Fiat, A., Karp, R. M., Luby, M., McGeoch, L. A., Sleator, D. D., and Young, N. E. (1991). Competitive paging algorithms. Journal of Algorithms, 12(4):685–699. 2

Goemans, M. X. and Williamson, D. P. (1995). Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. J. ACM, 42(6):1115–1145. 14

Golrezaei, N., Jaillet, P., and Zhou, Z. (2023). Online resource allocation with convex-set machine-learned advice. arXiv preprint arXiv:2306.12282. 14

Grigorescu, E., Lin, Y.-S., Silwal, S., Song, M., and Zhou, S. (2022a). Learning-augmented algorithms for online linear and semidefinite programming. Advances in Neural Information Processing Systems, 35:38643–38654. 2, 4, 5, 6, 7, 8, 9, 14, 20

Grigorescu, E., Lin, Y.-S., Silwal, S., Song, M., and Zhou, S. (2022b). Learning-augmented algorithms for online linear and semidefinite programming. 2, 4, 5, 9

Henzinger, M., Saha, B., Seybold, M. P., and Ye, C. (2024). On the complexity of algorithms with predictions for dynamic graph problems. In 15th Innovations in Theoretical Computer Science Conference (ITCS 2024). Schloss-Dagstuhl-Leibniz Zentrum für Informatik. 14

Hoi, S. C., Sahoo, D., Lu, J., and Zhao, P. (2021). Online learning: A comprehensive survey. Neurocomputing, 459:249–289. 1

Hsu, C.-Y., Indyk, P., Katabi, D., and Vakilian, A. (2019). Learning-based frequency estimation algorithms. In International Conference on Learning Representations. 14

Im, S., Kumar, R., Montazer Qaem, M., and Purohit, M. (2021). Online knapsack with frequency predictions. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, Advances in Neural Information Processing Systems, volume 34, pages 2733–2743. Curran Associates, Inc. 4, 14, 33

Jiang, S. H.-C., Liu, E., Lyu, Y., Tang, Z. G., and Zhang, Y. (2021). Online facility location with predictions. In International Conference on Learning Representations. 14

Jin, B. and Ma, W. (2022). Online bipartite matching with advice: Tight robustness-consistency tradeoffs for the two-stage model. Advances in Neural Information Processing Systems, 35:14555–14567. 14

Karlin, A. R., Manasse, M. S., McGeoch, L. A., and Owicki, S. S. (1994). Competitive randomized algorithms for non-uniform problems. Algorithmica, 11(6):542–571. 1, 14

Kevi, E. and Thang, N. K. (2023a). Online primal-dual algorithm with predictions for non-linear covering problems. 14

Kevi, E. and Thang, N. K. (2023b). Primal-dual algorithms with predictions for online bounded allocation and ad-auctions problems. In Algorithmic Learning Theory. 14

Khodak, M., Balcan, M.-F. F., Talwalkar, A., and Vassilvitskii, S. (2022). Learning predictions for algorithms with predictions. Advances in Neural Information Processing Systems, 35:3542–3555. 14

Lattanzi, S., Lavastida, T., Moseley, B., and Vassilvitskii, S. (2020). Online scheduling via learned weights. In Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pages 1859–1877. SIAM. 14

Lechowicz, A., Christianson, N., Sun, B., Bashir, N., Hajiesmaili, M., Wierman, A., and Shenoy, P. (2024). Online conversion with switching costs: Robust and learning-augmented algorithms. ACM SIGMETRICS Performance Evaluation Review, 52(1):45–46. 14

Leonardi, S. and Marchetti-Spaccamela, A. (1995). On-line resource management with applications to routing and scheduling. In International Colloquium on Automata, Languages, and Programming, pages 303–314. Springer. 33

Lin, Q., Yi, H., Pang, J., Chen, M., Wierman, A., Honig, M., and Xiao, Y. (2019). Competitive online optimization under inventory constraints. Proceedings of the ACM on Measurement and Analysis of Computing Systems, 3(1):1–28. 35

Lindermayr, A. and Megow, N. (2022). Algorithms with predictions. https://algorithms-with-predictions.github.io/. Accessed: 2024-05-22. 14

Lucic, M., Faulkner, M., Krause, A., and Feldman, D. (2018). Training gaussian mixture models at scale via coresets. Journal of Machine Learning Research, 18(160):1–25. 14

Lykouris, T. and Vassilvitskii, S. (2021). Competitive caching with machine learned advice. Journal of the ACM (JACM), 68(4):1–25. 1, 14

Ma, W., Simchi-Levi, D., and Zhao, J. (2019). The competitive ratio of threshold policies for online unit-density knapsack problems. arXiv preprint arXiv:1907.08735. 33

Mahdian, M., Nazerzadeh, H., and Saberi, A. (2012). Online optimization with uncertain information. ACM Transactions on Algorithms (TALG), 8(1):1–29. 14

Marchetti-Spaccamela, A. and Vercellis, C. (1995). Stochastic on-line knapsack problems. Mathematical Programming, 68(1):73–104. 33

Martello, S. and Toth, P. (1987). Algorithms for knapsack problems. North-Holland Mathematics Studies, 132:213–257. 33

Mitzenmacher, M. (2018). A model for learned bloom filters and optimizing by sandwiching. In Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems, NeurIPS, pages 462–471. 14

Mitzenmacher, M. and Vassilvitskii, S. (2022). Algorithms with predictions. Communications of the ACM, 65(7):33–35. 2

Purohit, M., Svitkina, Z., and Kumar, R. (2018). Improving online algorithms via ml predictions. Advances in Neural Information Processing Systems, 31. 1, 14

Rohatgi, D. (2020). Near-optimal bounds for online caching with machine learned advice. In Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 1834–1845. 14

Salkin, H. M. and De Kluyver, C. A. (1975). The knapsack problem: a survey. Naval Research Logistics Quarterly, 22(1):127–144. 33

Shen, X. and Nagarajan, V. (2020). Online covering with ℓ_q-norm objectives and applications to network design. Mathematical Programming, 184(1):155–182. 2, 5, 26, 27, 28, 29, 32, 37

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199. 1

Thang, N. K. and Dürr, C. (2021). Online primal-dual algorithms with predictions for packing problems. CoRR. 14

A ADDITIONAL PRIOR WORK

Learning-augmented algorithms. Learning-augmented algorithms have been extensively studied for many fundamental online problems. Rohatgi (2020) and Lykouris and Vassilvitskii (2021) showed that accurate predictions can lead to competitive ratios better than classical impossibility results for online caching, and subsequent works studied problems such as set cover (Bamas et al., 2020b), ski rental (Purohit et al., 2018; Bamas et al., 2020b), clustering (Ergun et al., 2022), graph problems (Banerjee et al., 2023; Brand et al., 2024; Henzinger et al., 2024), facility location (Jiang et al., 2021), knapsack (Im et al., 2021; Boyar et al., 2022; Daneshvaramoli et al., 2023), matching (Dinitz et al., 2021; Jin and Ma, 2022), and others (Hsu et al., 2019; Antoniadis et al., 2020; Bamas et al., 2020a,b; Lattanzi et al., 2020; Diakonikolas et al., 2021; Dütting et al., 2021; Chen et al., 2022a,b; Cohen and Panigrahi, 2023; Kevi and Thang, 2023b; Brand et al., 2024). While most works in learning-augmented algorithms treat the advice as a black-box device, there are some recent works that studies how to properly obtain such advices (e.g., Balcan, 2020; Khodak et al., 2022). A comprehensive online archive of recent works in the field of learning-augmented algorithms can be found at Lindermayr and Megow (2022).

Other lines of work exploring alternative forms of additional information given to the algorithm studies the stochastic setting, where the assumption is that the input instance is drawn from an underlying distribution known to the algorithm. Problems studied under this model include stochastic matching (Lucic et al., 2018), graph optimization (Azar et al., 2022), and others (Mahdian et al., 2012; Mitzenmacher, 2018). In comparison, the advice model studied by the learning-augmented algorithms community at large considers only explicit advices about the future inputs of the instance. While these advices may take forms that admits a distributional interpretation, there is no assumption made on the input distribution itself.

The primal-dual method. Introduced in the seminal work of Goemans and Williamson (1995), the primal-dual method is a powerful algorithmic technique used to solve a variety of problems in the field of approximation algorithms. The primal-dual method has been applied to individual problems such as online set cover (Alon et al., 2009), network optimization (Alon et al., 2006), ski rental (Karlin et al., 1994), and generalized into a unifying framework for online LP-based problems by Buchbinder and Naor (2009a). We refer the readers to Buchbinder and Naor (2009b) for a survey on the topic of primal-dual methods for online algorithms.

In the field of learning-augmented algorithms, Bamas, Maggiori, and Svensson (2020b) initiated the study of primal-dual learning-augmented algorithms, and inspired follow-up work to extend the framework to more generalized models. Anand et al. (2022) considered PDLA algorithms on online covering LPs with multiple predictions; Grigorescu et al. (2022a) generalized the PDLA framework to general online covering problems with fractional constraints and advices, as well as online semidefinite programs with ideas from Elad et al. (2016).

Towards generalizing the framework of Buchbinder and Naor (2009a) and Grigorescu et al. (2022a) to non-linear objectives, a concurrent line of work by Thang and Dürr (2021) and Kevi and Thang (2023a) studies online packing and covering problems using configuration programs and multilinear extensions of monotone objective functions. We remark that their model admits more generality in allowing non-convex or non-concave objective functions, but also incurs a potential loss in performance due to the generalized setting, and thus are incomparable to ours. Their advice is also integral, while the form of advice we consider allows for fractional predictions.

Some prior works have studied problems that allow a convex covering (e.g., Bamas et al., 2020a; Cohen and Panigrahi, 2023; Golrezaei et al., 2023; Balkanski et al., 2024; Lechowicz et al., 2024) and concave packing (e.g., Im et al., 2021; Dütting et al., 2021; Jin and Ma, 2022; Kevi and Thang, 2023b) program formulation in the context of learning-augmented algorithms. Our learning-augmented frameworks unify and generalize these settings and also apply to many other problems.

B REMAINING PROOF FOR ONLINE CONVEX COVERING

In this appendix, we complete Section 3 by providing a full proof of Theorem 3.2, as well as present a variant of Algorithm 2 that we use in Appendix D to obtain better performance bounds.

Recall that our goal is to solve the following in an online fashion:

$$\min f(x) \text{ over } x \in \mathbb{R}^n_{\geq 0} \text{ subject to } Ax \geq \mathbf{1}. \tag{2}$$

where $f : \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a convex, monotone, differentiable function with $f(\mathbf{0}) = 0$. We assume that ∇f is monotone, and there exists some $p := \sup_{x \geq \mathbf{0}} \frac{\langle x, \nabla f(x) \rangle}{f(x)}$.

Our choice of the dual packing program to our convex covering program is the Fenchel dual program, formulated as:

$$\max \sum_{i=1}^{m} y_i - f^*(\mu) \text{ over } y \in \mathbb{R}^m_{\geq 0} \text{ subject to } y^T A \leq \mu^T \tag{4}$$

where $f^* : \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ is the Fenchel dual of f, defined as $f^*(\mu) = \sup_z \{\mu^T z - f(z)\}$.

Algorithm 2 PDLA algorithm for online convex covering problems in round t

Input: $x_1, \cdots, x_n$, $y_1, \cdots, y_{t-1}$, μ: current solution, A_{tj} for $1 \leq j \leq n$: current coefficients, (x', λ): advice and confidence parameter.
Output: Updated x and (y, μ).

$\mu \leftarrow \delta \nabla f(\bar{x})$.
$y_t \leftarrow 0$.
$\tau \leftarrow$ current time.
if $A_t x' \geq 1$ **then** ▷ if x' is feasible
    $D_j^{(t)} \leftarrow \frac{\lambda}{a_{tj} d} + \frac{(1-\lambda) x'_j \mathbf{1}_{x_j < x'_j}}{A_i x'_c}$.
else
    $D_j^{(t)} \leftarrow \frac{1}{a_{tj} d}$.
while $A_t x < 1$ **do** ▷ while unsatisfied
    Increase τ at rate 1.
    for each $j \in [n]$ s.t. $a_{tj} > 0$ **do**
        Increase x_j at rate

$$\frac{\partial x_j}{\partial \tau} = \frac{a_{tj}}{\nabla_j f(x)} (x_j + D_j^{(t)})$$

    Increase y_t at rate

$$r := \frac{\partial y_t}{\partial \tau} = \frac{\delta}{\log(1 + \frac{2}{\lambda} d^2)}$$

    for each $j \in [n]$ s.t. $\sum_{i=1}^{t} a_{ij} y_i = \mu_j$ **do**
        $m_j^* = \arg\max_{i=1}^{t} \{a_{ij} | y_i > 0\}$.
        Decrease $y_{m_j^*}$ at rate $\frac{a_{tj}}{a_{(m_j^*)j}} \cdot r$.

For completeness' sake, we restate the weak duality theorem between the convex covering program and its Fenchel dual (Fact 3.1) and provide a formal proof:

Fact 3.1 (Buchbinder et al. (2014))**.** *Let x and (y, μ) be any feasible solution to the primal covering program (2) and the dual packing program (4), respectively. Then*

$$f(x) \geq \sum_{i=1}^{m} y_i - f^*(\mu)$$

Proof. Observe that by definition,

$$\sum_{i=1}^{m} y_i = y^T \mathbf{1} \leq y^T A x \leq \mu^T x = (\mu^T x - f(x)) + f(x) \leq f^*(\mu) + f(x).$$

Rearranging, we have $\sum_{i=1}^{m} y_i - f^*(\mu) \leq f(x)$, as desired. □

Finally, we restate our main result, Theorem 3.2:

Theorem 3.2. *For the learning-augmented online convex covering problem with the monotone gradient assumption, there exists an online algorithm that takes a problem instance and an advice x', and generates a solution $\bar{x}$ such that*

$$f(\bar{x}) \leq \min\left\{O\left(\frac{1}{1-\lambda}\right) f(x') + O((p \log d)^p)\mathit{OPT}, O\left(\left(p \log \frac{d}{\lambda}\right)^p\right)\mathit{OPT}\right\}$$

Additionally, if x' is feasible, i.e., $Ax' \geq \mathbf{1}$, then $f(\bar{x}) \leq O\left(\frac{1}{1-\lambda}\right) f(x')$. Here, $d = \max_{i \in [m]} |\{a_{ij} | a_{ij} > 0\}|$ is the row sparsity of the constraint matrix A, and $p := \sup_{x \geq \mathbf{0}} \frac{\langle x, \nabla f(x) \rangle}{f(x)}$.

We prove Theorem 3.2 by proving the two components, $f(\bar{x}) \leq O((p \log d/\lambda)^p)\mathsf{OPT}$ (robustness) and $f(\bar{x}) \leq O(1/(1-\lambda))f(x') + O((p \log d)^p)$ (consistency), separately.

Robustness. First, we prove the robustness of Algorithm 2, that $f(\bar{x}) \leq O((p \log d/\lambda)^p)\mathsf{OPT}$. Specifically, we show the following:

1. $\bar{x}$ is feasible and monotone;
2. $(\bar{y}, \mu)$ is feasible, and μ is monotone;
3. The primal objective P is at most $O((p \log d/\lambda)^p)$ times the dual objective D.

Equipped with these subclaims and weak duality, we have

$$P = f(\bar{x}) \leq O((p \log d/\lambda)^p) D \leq O((p \log d/\lambda)^p)\mathsf{OPT}$$

as desired.

We begin by showing the first two feasibility claims together.

Lemma B.1. *For any $\delta > 0$, the following are maintained.*

- *The algorithm maintains a feasible monotone primal solution.*
- *The algorithm maintains a feasible dual solution with monotone μ_j.*

Proof. The feasibility of the primal solution follows by construction, since we only increment each coordinate of x continuously, at least one of which contributing with non-zero rate, and only stop when the arriving constraint is satisfied.

By design we only set μ once throughout the entire execution. To show that the dual is feasible, observe that at round t and time τ, for any dual constraint $\sum_{i=1}^{t} a_{ij} y_i^{(\tau)} \leq \mu_j$, if the inequality is strict, the constraint is satisfied and we are done. If there is equality, line 11 through 13 of Algorithm 2 will counteract by decrementing $y_{m_j^*}$, resulting in a net growth rate of

$$\frac{\partial}{\partial \tau} \sum_{i=1}^{t} a_{ij} y_i^{(\tau)} = a_{ij} \cdot r - a_{(m_j^*)j} \cdot \frac{a_{tj}}{a_{(m_j^*)j}} \cdot r = 0$$

so that the constraint remains satisfied. □

Towards showing that the ratio between the primal and the dual objectives are bounded, our strategy on a high level is classical to proving the competitiveness of primal-dual algorithms, bounding the ratio between the rates of change of both the primal and the dual objectives during the algorithm's execution. However, the rate of change of the dual objective, specifically $\sum_i y_i$, is very nuanced to analyze, since we both increment y_t on line 11 of Algorithm 2 and decrement some other dual variable on line 14. Towards this, we use a sandwich strategy with the primal variables x_j as a proxy, to bound the rate of decrease of $\sum_i y_i$ on line 14.

We present the following lemma as the first part of our sandwich strategy, a lower bound on the value of x_j at time τ:

Lemma B.2. *For a variable x_j, let $T_j = \{i | a_{ij} > 0\}$ and let S_j be any subset of T_j. Then for any $t \in T_j$ and $\tau_t \leq \tau \leq \tau_{t+1}$,*

$$x_j^{(\tau)} \geq \frac{\lambda}{\max_{i \in S_j}\{a_{ij}\} \cdot d} \cdot \left(\exp\left(\frac{\log(1+\frac{2}{\lambda}d^2)}{\mu_j} \sum_{i \in S_j} a_{ij} y_i^{(\tau)} \right) - 1 \right)$$

where τ_t denotes the value of τ at the arrival of the t-th primal constraint.

Proof. Note that

$$\begin{aligned} \frac{\partial x_j}{\partial y_t} = \frac{\partial x_j}{\partial \tau} \cdot \frac{\partial \tau}{\partial y_t} &= \frac{a_{tj}(x_j + D_j^{(t)})}{\nabla_j f(x)} \cdot \frac{\log(1+\frac{2}{\lambda}d^2)}{\delta} \\ &\geq \log(1+\frac{2}{\lambda}d^2) \frac{a_{tj}(x_j + D_j^{(t)})}{\delta \nabla_j f(\bar{x})} \end{aligned} \tag{5}$$

where the last inequality is due to our assumption that the gradient is monotone, and that $x \leq \bar{x}$. Solving this differential equation, we have

$$\frac{x_j^{(\tau)} + D_j^{(t)}}{x_j^{(\tau_t)} + D_j^{(t)}} \geq \exp\left(\frac{\log(1+\frac{2}{\lambda}d^2)}{\delta \nabla_j f(\bar{x})} \cdot a_{tj} y_t^{(\tau)} \right)$$

From here, we use the following two claims:

Claim B.3. *For all scalars $a \geq b > 0$ and $c_1 \geq c_2 \geq 0$,*

$$\frac{a + c_1}{b + c_1} \leq \frac{a + c_2}{b + c_2}$$

Claim B.4. *For any $j \in [n]$ and $i \in S_j$ the following holds:*

$$D_j^{(i)} = \frac{\lambda}{a_{ij} d} + \frac{(1-\lambda) x'_j \mathbf{1}_{x_j < x'_j}}{A_i x'_c} \geq \frac{\lambda}{\max_{i \in S_j} a_{ij} d}$$

Multiplying over all indices, where for notational convenience we set $\tau_{t+1} = \tau$,

$$\begin{aligned}
\exp\left(\frac{\log(1+\frac{2}{\lambda}d^2)}{\mu_j}\sum_{i\in S_j} a_{ij}y_i^{(\tau)}\right) &= \exp\left(\sum_{i\in S_j}\frac{\log(1+\frac{2}{\lambda}d^2)}{\delta\nabla_j f(\bar{x})}\cdot a_{ij}y_i^{(\tau)}\right) \\
&\le \prod_{i\in S_j}\frac{x_j^{(\tau_{i+1})}+D_j^{(i)}}{x_j^{(\tau_i)}+D_j^{(i)}} \\
&\le \prod_{i\in S_j}\frac{x_j^{(\tau_{i+1})}+\frac{\lambda}{\max_{i\in S_j}a_{ij}d}}{x_j^{(\tau_i)}+\frac{\lambda}{\max_{i\in S_j}a_{ij}d}} && \text{by Claim B.3 and Claim B.4} \\
&\le \prod_{i\in T_j}\frac{x_j^{(\tau_{i+1})}+\frac{\lambda}{\max_{i\in S_j}a_{ij}d}}{x_j^{(\tau_i)}+\frac{\lambda}{\max_{i\in S_j}a_{ij}d}} && \text{since } x_j \text{ is monotone} \\
&= \frac{x_j^{(\tau)}+\frac{\lambda}{\max_{i\in S_j}a_{ij}d}}{\frac{\lambda}{\max_{i\in S_j}a_{ij}d}} && \text{by a telescoping argument over all indices}
\end{aligned}$$

Reorganizing the terms yields the desired bound on $x_j^{(\tau)}$. □

Since the arriving constraint is not satisfied yet during the algorithm, trivially $1/a_{tj}$ is a upper bound on x_j. Equipped with this and the lower bound in Lemma B.2, we move on to bound the ratio between the primal objective P and the dual objective D:

Lemma B.5. *Let* $p := \sup_{x\ge\mathbf{0}}\frac{\langle x,\nabla f(x)\rangle}{f(x)}$. *Then*

$$P = f(\bar{x}) \le (4p\log(1+\frac{2}{\lambda}d^2))^p\left(\sum_{i=1}^m y_i - f^*(\mu)\right) = O((p\log\frac{d}{\lambda})^p)D$$

Proof. We assume that x' is feasible, i.e., $A_t x' \ge 1$. A similar analysis can be obtained for the case of $A_t x' < 1$ by setting $\lambda = 1$.

Consider the update when primal constraint t arrives and τ is the current time. Let $U(\tau)$ denote the set of tight dual constraints at time τ, i.e., for every $j \in U(\tau)$ we have $a_{tj} > 0$ and $\sum_{i=1}^t a_{ij}y_i^{(\tau)} = \delta\nabla_j f(\bar{x})$. $|U(\tau)| \le d$, and define for every j the set $S_j := \{i | a_{ij} > 0, y_i^{(\tau)} > 0\}$. Clearly $\sum_{i\in S_j} a_{ij}y_i^{(\tau)} = \sum_{i=1}^t a_{ij}y_i^{(\tau)} = \delta\nabla_j f(\bar{x})$, and $\sum_j a_{tj}x_j^{(\tau)} < 1$, thus by Lemma B.2, we have

$$\frac{1}{a_{tj}} > x_j^{(\tau)} \ge \frac{\lambda}{\max_{i\in S_j}\{a_{ij}\}\cdot d}\cdot\left(\exp\left(\log(1+\frac{2}{\lambda}d^2)\right)-1\right)$$

Rearranging, we have

$$\frac{a_{tj}}{a_{m_j^* j}} = \frac{a_{tj}}{\max_{i\in S_j}a_{ij}} \le \frac{1}{2d}$$

Thus, we can bound the rate of change of $\sum_{i=1}^t y_i$ at time τ:

$$\frac{\partial(\sum_{i=1}^t y_i)}{\partial\tau} \ge r - \sum_{j\in U(\tau)}\frac{a_{tj}}{a_{m_j^* j}}\cdot r \ge r\left(1-\sum_{j\in U(\tau)}\frac{1}{2d}\right) \ge \frac{1}{2}r$$

where the last inequality follows from $|U(\tau)| \le d$.

On the other hand, when processing constraint t, the rate of increase of the primal is

$$\begin{aligned}\frac{\partial f(x^{(\tau)})}{\partial \tau} &= \sum_j \nabla_j f(x^{(\tau)}) \frac{\partial x_j^{(\tau)}}{\partial \tau} \qquad \text{by the chain rule} \\ &= \sum_{j|a_{tj}>0} \nabla_j f(x^{(\tau)}) \left(\frac{a_{tj}(x_j^{(\tau)} + D_j^{(t)})}{\nabla_j f(x^{(\tau)})} \right) \\ &= \sum_{j|a_{tj}>0} \left(a_{tj} x_j^{(\tau)} + \frac{\lambda}{d} + \frac{(1-\lambda) a_{tj} x'_j \mathbf{1}_{x_j < x'_j}}{A_t x'_c} \right) \\ &\leq 1 + \lambda + (1-\lambda) = 2\end{aligned}$$

Here, the last inequality is due to the constraint being unsatisfied, and the definition of x'_c as the constraint restricted to entries where $x_j < x'_j$ holds.

Thus, bounding the rate of change between the primal and the dual:

$$\frac{\partial(\sum_{i=1}^t y_i^{(\tau)})}{\partial f(x^{(\tau)})} \geq \frac{1}{4} r = \frac{\delta}{4\log(1 + \frac{2}{\lambda} d^2)}$$

Let $\bar{x}$ and $\bar{y}$ be the final primal and dual solutions. We have

$$\sum_{i=1}^m \bar{y}_i \geq \frac{\delta}{4\log(1 + \frac{2}{\lambda} d^2)} \cdot f(\bar{x}).$$

We have thus obtained our bound on the part of the dual objective corresponding to the variables y. To obtain a simple form of the dual objective corresponding to μ, we use the following facts on f, which bounds how quickly f and f^* can grow:

Fact B.6 (Bounded Growth, Azar et al. (2016))**.** *Let f be a convex, monotone, differentiable function with non-decreasing gradient and $f(0) = 0$. Assume additionally that there is some $p \geq 1$ s.t. $x\nabla f(x) \leq p f(x)$ for all $x \in \mathbb{R}^n_{\geq 0}$. Then,*

1. *For all $\delta \geq 1$, for all $x \in \mathbb{R}^n_{\geq 0}$, $f(\delta x) \leq \delta^p f(x)$;*
2. *For all $x \in \mathbb{R}^n_{\geq 0}$, $f^*(\nabla f(x)) = x^T \nabla f(x) - f(x) \leq (p-1) f(x)$;*
3. *If $p > 1$ then for any $0 < \gamma \leq 1$, $x \in \mathbb{R}^n_{\geq 0}$, $f^*(\gamma x) \leq \gamma^{\frac{p}{p-1}} f^*(x)$ and $f^*(\gamma \nabla f(x)) \leq \gamma^{\frac{p}{p-1}} (p-1) f(x)$;*
4. *If $p = 1$ then for any $0 < \gamma \leq 1$, $x \in \mathbb{R}^n_{\geq 0}$, $f^*(\gamma \nabla f(x)) \leq 0$.*

See Azar et al. (2016) for a proof of Fact B.6.

By definition of the dual objective D, we have

$$\begin{aligned}D &= \sum_{i=1}^m \bar{y}_i - f^*(\mu) \\ &\geq \frac{\delta}{4\log(1 + \frac{2}{\lambda} d^2)} \cdot f(\bar{x}) - f^*(\delta \nabla f(\bar{x})) \\ &\geq \left(\frac{\delta}{4\log(1 + \frac{2}{\lambda} d^2)} - \delta^{\frac{p}{p-1}} (p-1) \right) \cdot f(\bar{x})\end{aligned}$$

where the last inequality is from (3) of Fact B.6. With the choice of $\delta = \frac{1}{(4p\log(1+\frac{2}{\lambda}d^2))^{p-1}}$, we have

$$\begin{aligned}\frac{\delta}{4\log(1+\frac{2}{\lambda}d^2)} - \delta^{\frac{p}{p-1}}(p-1) &= \frac{1}{(4\log(1+\frac{2}{\lambda}d^2))^p \cdot p^{p-1}} - \frac{p-1}{(4\log(1+\frac{2}{\lambda}d^2))^p \cdot p^p} \\ &= \frac{1}{(4p\log(1+\frac{2}{\lambda}d^2))^p}\end{aligned}$$

Thus,

$$P = f(\bar{x}) \leq (4p\log(1+\frac{2}{\lambda}d^2))^p D = O((p\log\frac{d}{\lambda})^p)D$$

as desired. □

Consistency. Towards showing the consistency of Algorithm 2, we employ a similar proof strategy as in Bamas et al. (2020b) and Grigorescu et al. (2022a). We partition the growth rate of the primal variables x_j based on whether the variable has exceeded the corresponding entry in the advice x'_j. With this partition, we argue that the part of the growth rate credited to the classical primal-dual increment procedure is at most a certain factor of the part of the growth rate credited to the advice. Specifically, we present and prove the following lemma:

Lemma B.7. *Algorithm 2 is $O(\frac{1}{1-\lambda})$-consistent, i.e., if $\bar{x}$ is the final solution of the algorithm, then $f(\bar{x}) \leq O(\frac{1}{1-\lambda})f(x') + O((p\log d)^p)OPT$.*

Proof. Let τ denote the current time and t denote the current round. For any arriving constraint A_t, in the case of $A_t x' < 1$, we can bound the growth rate similarly to in Lemma B.5 with λ set to 1. Thus we can obtain $f(\bar{x}) \leq O((p\log d)^p)OPT$.

Assuming that x' is feasible, i.e., $A_t x' \geq 1$, notice that

$$\frac{\partial f(x^{(\tau)})}{\partial \tau} = \sum_j \nabla_j f(x^{(\tau)}) \frac{\partial x_j^{(\tau)}}{\partial \tau}$$

We split the rate of growth of the primal objective as follows: let $S_c := \{j | x_j < x'_j\}$ denote the set of indices for which the primal variable has not reached the corresponding advice variable yet, and let $S_u := \{j | x_j \geq x'_j\}$ denote the set of indices for which the primal variable has surpassed the corresponding advice variable. Further define

$$r_c = \sum_{j \in S_c} \nabla_j f(x^{(\tau)}) \frac{\partial x_j^{(\tau)}}{\partial \tau}$$

and

$$r_u = \sum_{j \in S_u} \nabla_j f(x^{(\tau)}) \frac{\partial x_j^{(\tau)}}{\partial \tau}$$

with

$$\frac{\partial f(x^{(\tau)})}{\partial \tau} = r_c + r_u$$

Note that the value of the advice, $f(x')$, can be lower bounded by the value of the solution of the primal-dual algorithm *if it does not increase any variable past the advice.* This is due to the monotonicity of f. As a result, we can exclusively attribute the rate of increase in r_c to the corresponding part in $f(x')$.

Notice that

$$\begin{aligned} r_c &= \sum_{j \in S_c} \nabla_j f(x^{(\tau)}) \frac{\partial x_j^{(\tau)}}{\partial \tau} \\ &= \sum_{j | x_j < x'_j} \left(a_{tj} x_j^{(\tau)} + \frac{\lambda}{d} + \frac{(1-\lambda) a_{tj} x'_j \mathbf{1}_{x_j < x'_j}}{A_t x'_c} \right) \\ &\geq 0 + 0 + (1-\lambda) \end{aligned}$$

since x'_c is defined as the advice restricted to indices in S_c, and

$$\begin{aligned} r_u &= \sum_{j \in S_u} \nabla_j f(x^{(\tau)}) \frac{\partial x_j^{(\tau)}}{\partial \tau} \\ &= \sum_{j | x_j \geq x'_j} \left(a_{tj} x_j^{(\tau)} + \frac{\lambda}{d} + \frac{(1-\lambda) a_{tj} x'_j \mathbf{1}_{x_j < x'_j}}{A_t x'_c} \right) \\ &\leq 1 + \lambda + 0 \end{aligned}$$

since d is the row sparsity, and that the arriving constraint t has not been satisfied yet. Thus we have

$$r_u \leq \frac{1+\lambda}{1-\lambda} r_c$$

and that

$$\frac{\partial f(x^{(\tau)})}{\partial \tau} \leq \left(1 + \frac{1+\lambda}{1-\lambda}\right) r_c = O\left(\frac{1}{1-\lambda}\right) r_c$$

which, taking integrals over τ, implies that $f(\bar{x}) \leq O\left(\frac{1}{1-\lambda}\right) f(x')$, as desired. □

Combining the robustness and consistency of Algorithm 2, we obtain a proof of Theorem 3.2.

B.1 A variant of Algorithm 2 in Buchbinder et al. (2014)

We remark that our Algorithm 2 follows the algorithmic structure and analysis of Azar et al. (2016), which is a merge of Buchbinder et al. (2014) with other works. Azar et al. (2016) is a significant simplification of Buchbinder et al. (2014), suffering a potential penalty in the competitive ratio in exchange for much cleaner expressions and analyses. For completeness, we state our variant of Algorithm 2 and Theorem 3.2 following Buchbinder et al. (2014), and briefly sketch the proof, specifically necessary modifications from the proof of Theorem 3.2. In Appendix D, we use this variant to obtain tighter bounds on various applications of our framework.

On a high level, Algorithm 3 follows exactly the same idea as Algorithm 2, with only two minor differences: On line 1, μ is set to $\nabla f(\delta \bar{x})$ instead of $\delta \nabla f(\bar{x})$[2]; On line 12, the numerator of r is $\min_{\ell=1}^{n} \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\}$ instead of δ. These two changes do not modify the primal update at all, and preserve a vital component in the proof of Theorem 3.2 that bounds the relative rate of change between x_j and y_t with μ_j, more specifically:

$$\begin{aligned} \frac{\partial x_j}{\partial y_i} = \frac{\partial x_j}{\partial \tau} \cdot \frac{\partial \tau}{\partial y_i} &= \frac{a_{ij}(x_j + D_j)}{\nabla_j f(x)} \cdot \frac{\log(1 + \frac{2}{\lambda} d^2)}{\min_{\ell=1}^{n} \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\}} \\ &\geq \log(1 + \frac{2}{\lambda} d^2) \frac{a_{ij}(x_j + D_j)}{\nabla_j f(\delta \bar{x})} \\ &= \log(1 + \frac{2}{\lambda} d^2) \frac{a_{ij}(x_j + D_j)}{\mu_j} \end{aligned}$$

[2]The choice of δ here is different between the two algorithms: they are the solutions to two distinct minimization problems.

Algorithm 3 The Buchbinder et al. (2014) variant of Algorithm 2 in round t

Input: $x_1, \cdots, x_n$, $y_1, \cdots, y_{t-1}$, μ: current solution, A_{tj} for $1 \le j \le n$: current coefficients, (x', λ): advice and confidence parameter.
Output: Updated x and (y, μ).

$\mu \leftarrow \nabla f(\delta \bar{x})$.
$y_t \leftarrow 0$.
$\tau \leftarrow$ current time.
if $A_t x' \ge 1$ **then** ▷ if x' is feasible
    $D_j^{(t)} \leftarrow \frac{\lambda}{a_{tj} d} + \frac{(1-\lambda) x'_j \mathbf{1}_{x_j < x'_j}}{A_i x'_c}$.
else
    $D_j^{(t)} \leftarrow \frac{1}{a_{tj} d}$.
while $A_t x < 1$ **do** ▷ while unsatisfied
    Increase τ at rate 1.
    for each $j \in [n]$ s.t. $a_{tj} > 0$ **do**
        Increase x_j at rate

$$\frac{\partial x_j}{\partial \tau} = \frac{a_{tj}}{\nabla_j f(x)} (x_j + D_j^{(t)})$$

    Increase y_t at rate

$$r := \frac{\partial y_t}{\partial \tau} = \frac{\min_{\ell=1}^{n} \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\}}{\log(1 + \frac{2}{\lambda} d^2)}$$

    for each $j \in [n]$ s.t. $\sum_{i=1}^{t} a_{ij} y_i = \mu_j$ **do**
        $m_j^* = \arg\max_{i=1}^{t} \{a_{ij} | y_i > 0\}$.
        Decrease $y_{m_j^*}$ at rate $\frac{a_{tj}}{a_{(m_j^*)j}} \cdot r$.

where the inequality follows from the monotonicity of $\nabla f(x)$ and

$$\begin{aligned} \min_{\ell=1}^{n} \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\} \cdot \nabla_j f(x) &\le \frac{\nabla_j f(\delta \bar{x})}{\nabla_j f(\bar{x})} \cdot \nabla_j f(x) \\ &\le \frac{\nabla_j f(\delta \bar{x})}{\nabla_j f(x)} \cdot \nabla_j f(x) \\ &= \nabla_j f(\delta \bar{x}) \end{aligned}$$

The choice of μ in Algorithm 3 no longer allows the bounded growth assumptions of Fact B.6 to simplify the $f^*(\mu)$ term in the dual objective. Thus, the performance of Algorithm 3 is stated in the corollary below:

Corollary B.8. *For the learning-augmented online convex covering problem with the monotone gradient assumption, there exists an online algorithm that takes a problem instance and an advice x', and generates a solution $\bar{x}$ such that*

$$f(\bar{x}) \le \min \left\{ O\left(\frac{1}{1-\lambda}\right) f(x') + R(1)\mathit{OPT}, R(\lambda)\mathit{OPT} \right\}$$

where

$$\frac{1}{R(\lambda)} = \max_{\delta > 0} \min_{z} \left(\frac{\min_\ell \left\{ \frac{\nabla_\ell f(\delta z)}{\nabla_\ell f(z)} \right\}}{4 \log(1 + \frac{2}{\lambda} d^2)} - \frac{\delta z^T \nabla f(\delta z) - f(\delta z)}{f(z)} \right). \tag{6}$$

Additionally, if x' is feasible, i.e., $Ax' \ge \mathbf{1}$, then $f(\bar{x}) \le O\left(\frac{1}{1-\lambda}\right) f(x')$. Here, $d = \max_{i \in [m]} |\{a_{ij} | a_{ij} > 0\}|$ is the row sparsity of the constraint matrix A.

While the statement of Corollary B.8 may be intimidating at first glance, its proof is almost identical to that of Theorem 3.2. For completeness' sake, we include the proof here.

B.2 Proof of Corollary B.8

As with the proof of Theorem 3.2, we prove the two clauses of Corollary B.8 separately: that $f(\bar{x}) \leq O(1/(1-\lambda))f(x') + R(1)\mathsf{OPT}$, and that $f(\bar{x}) \leq R(\lambda)\mathsf{OPT}$.

Robustness. We show the following subclaims:

1. $\bar{x}$ is feasible and monotone;
2. $(\bar{y}, \mu)$ is feasible, and μ is monotone;
3. The primal objective P is at most $R(\lambda)$ times the dual objective D.

which, together with weak duality, implies:

$$P = f(\bar{x}) \leq R(\lambda)D \leq R(\lambda)\mathsf{OPT}$$

as desired.

For the two feasibility subclaims, Lemma B.1 holds by algorithm design, regardless of our choice of μ and r, the dual growth rate.

Towards the third subclaim bounding the ratio between the primal and dual objectives, we continue using the sandwich strategy, starting with the lower bound of x_j at time τ, corresponding to Lemma B.2:

Lemma B.9. *In context of Algorithm 3, for a variable x_j, let $T_j = \{i | a_{ij} > 0\}$ and let S_j be any subset of T_j. Then for any $t \in T_j$ and $\tau_t \leq \tau \leq \tau_{t+1}$,*

$$x_j^{(\tau)} \geq \frac{\lambda}{\max_{i \in S_j}\{a_{ij}\} \cdot d} \cdot \left(\exp\left(\frac{\log(1 + \frac{2}{\lambda}d^2)}{\mu_j} \sum_{i \in S_j} a_{ij} y_i^{(\tau)} \right) - 1 \right)$$

where τ_t denotes the value of τ at the arrival of the t-th primal constraint.

Proof. Note that

$$\begin{aligned}
\frac{\partial x_j}{\partial y_i} = \frac{\partial x_j}{\partial \tau} \cdot \frac{\partial \tau}{\partial y_i} &= \frac{a_{ij}(x_j + D_j)}{\nabla_j f(x)} \cdot \frac{\log(1 + \frac{2}{\lambda}d^2)}{\min_{\ell=1}^{n} \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\}} \\
&\geq \log(1 + \frac{2}{\lambda}d^2) \frac{a_{ij}(x_j + D_j)}{\nabla_j f(\delta \bar{x})} \\
&= \log(1 + \frac{2}{\lambda}d^2) \frac{a_{ij}(x_j + D_j)}{\mu_j}
\end{aligned}$$

where the inequality follows from the monotonicity of $\nabla f(x)$ and

$$\begin{aligned}
\min_{\ell=1}^{n} \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\} \cdot \nabla_j f(x) &\leq \frac{\nabla_j f(\delta \bar{x})}{\nabla_j f(\bar{x})} \cdot \nabla_j f(x) \\
&\leq \frac{\nabla_j f(\delta \bar{x})}{\nabla_j f(x)} \cdot \nabla_j f(x) \\
&= \nabla_j f(\delta \bar{x})
\end{aligned}$$

where the last inequality is due to our assumption that the gradient is monotone, and that $x \leq \bar{x}$. Solving this differential equation, we have

$$\frac{x_j^{(\tau)} + D_j^{(t)}}{x_j^{(\tau_t)} + D_j^{(t)}} \geq \exp\left(\frac{\log(1+\frac{2}{\lambda}d^2)}{\mu_j} \cdot a_{tj} y_t^{(\tau)}\right)$$

Multiplying over all indices, where for notational convenience we set $\tau_{t+1} = \tau$,

$$\begin{aligned}
\exp\left(\frac{\log(1+\frac{2}{\lambda}d^2)}{\mu_j}\sum_{i\in S_j} a_{ij} y_i^{(\tau)}\right) &\leq \prod_{i\in S_j} \frac{x_j^{(\tau_{i+1})} + D_j^{(i)}}{x_j^{(\tau_i)} + D_j^{(i)}} \\
&\leq \prod_{i\in S_j} \frac{x_j^{(\tau_{i+1})} + \frac{\lambda}{\max_{i\in S_j} a_{ij} d}}{x_j^{(\tau_i)} + \frac{\lambda}{\max_{i\in S_j} a_{ij} d}} && \text{by Claim B.3 and Claim B.4} \\
&\leq \prod_{i\in T_j} \frac{x_j^{(\tau_{i+1})} + \frac{\lambda}{\max_{i\in S_j} a_{ij} d}}{x_j^{(\tau_i)} + \frac{\lambda}{\max_{i\in S_j} a_{ij} d}} && \text{since } x_j \text{ is monotone} \\
&= \frac{x_j^{(\tau)} + \frac{\lambda}{\max_{i\in S_j} a_{ij} d}}{\frac{\lambda}{\max_{i\in S_j} a_{ij} d}} && \text{by a telescoping argument over all indices}
\end{aligned}$$

Reorganizing the terms yields the desired bound on $x_j^{(\tau)}$. □

With the lower bound of Lemma B.9 and the trivial upper bound of $1/a_{tj}$, we continue to proving the third subclaim bounding the ratio between the primal and dual objectives, analogous to Lemma B.5:

Lemma B.10. *Let* $p := \sup_{x\geq \mathbf{0}} \frac{\langle x, \nabla f(x)\rangle}{f(x)}$. *Then*

$$P = f(\bar{x}) \leq R(\lambda) D$$

where

$$\frac{1}{R(\lambda)} = \max_{\delta>0} \min_{z} \left(\frac{\min_\ell \left\{\frac{\nabla_\ell f(\delta z)}{\nabla_\ell f(z)}\right\}}{4\log(1+\frac{2}{\lambda}d^2)} - \frac{\delta z^T \nabla f(\delta z) - f(\delta z)}{f(z)}\right).$$

Proof. We assume that x' is feasible, i.e., $A_t x' \geq 1$. A similar analysis can be obtained for the case of $A_t x' < 1$ by setting $\lambda = 1$.

Consider the update when primal constraint t arrives and τ is the current time. Let $U(\tau)$ denote the set of tight dual constraints at time τ, i.e., for every $j \in U(\tau)$ we have $a_{tj} > 0$ and $\sum_{i=1}^t a_{ij} y_i^{(\tau)} = \nabla_j f(\delta\bar{x}) = \mu_j$. $|U(\tau)| \leq d$, and define for every j the set $S_j := \{i | a_{ij} > 0, y_i^{(\tau)} > 0\}$. Clearly $\sum_{i\in S_j} a_{ij} y_i^{(\tau)} = \sum_{i=1}^t a_{ij} y_i^{(\tau)} = \nabla_j f(\delta\bar{x}) = \mu_j$, and $\sum_j a_{tj} x_j^{(\tau)} < 1$, thus by Lemma B.2, we have

$$\frac{1}{a_{tj}} > x_j^{(\tau)} \geq \frac{\lambda}{\max_{i\in S_j}\{a_{ij}\}\cdot d} \cdot \left(\exp\left(\log(1+\frac{2}{\lambda}d^2)\right) - 1\right)$$

Rearranging, we have

$$\frac{a_{tj}}{a_{m_j^* j}} = \frac{a_{tj}}{\max_{i\in S_j} a_{ij}} \leq \frac{1}{2d}$$

Thus, we can bound the rate of change of $\sum_{i=1}^t y_i$ at time τ:

$$\frac{\partial(\sum_{i=1}^t y_i)}{\partial\tau} \geq r - \sum_{j\in U(\tau)} \frac{a_{tj}}{a_{m_j^* j}} \cdot r \geq r\left(1 - \sum_{j\in U(\tau)} \frac{1}{2d}\right) \geq \frac{1}{2} r$$

where the last inequality follows from $|U(\tau)| \le d$.

The rate of increase of the primal is unaffected by the changes we made in Algorithm 3. Thus, we can similarly bound the rate of change between the primal and the dual as before:

$$\begin{aligned}\frac{\partial(\sum_{i=1}^{t} y_i^{(\tau)})}{\partial f(x^{(\tau)})} &= \frac{\partial(\sum_{i=1}^{t} y_i^{(\tau)})}{\partial \tau} \cdot \frac{\partial \tau}{\partial f(x^{(\tau)})} \\ &\ge \frac{1}{2}r \cdot \frac{1}{2} = \frac{1}{4}r = \frac{\min_\ell \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\}}{4\log(1+\frac{2}{\lambda}d^2)}\end{aligned}$$

Taking integral over τ, we have

$$\sum_{i=1}^{m} \bar{y}_i \ge \frac{\min_\ell \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\}}{4\log(1+\frac{2}{\lambda}d^2)} \cdot f(\bar{x}).$$

By definition of the dual objective D, we have

$$\begin{aligned}D &= \sum_{i=1}^{m} \bar{y}_i - f^*(\mu) \\ &\ge \frac{\min_\ell \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\}}{4\log(1+\frac{2}{\lambda}d^2)} \cdot f(\bar{x}) - f^*(\nabla f(\delta \bar{x})) \\ &= \left(\frac{\min_\ell \left\{ \frac{\nabla_\ell f(\delta \bar{x})}{\nabla_\ell f(\bar{x})} \right\}}{4\log(1+\frac{2}{\lambda}d^2)} - \frac{\delta \bar{x}^T \nabla f(\delta \bar{x}) - f(\delta \bar{x})}{f(\bar{x})} \right) f(\bar{x}) \\ &\ge \min_z \left(\frac{\min_\ell \left\{ \frac{\nabla_\ell f(\delta z)}{\nabla_\ell f(z)} \right\}}{4\log(1+\frac{2}{\lambda}d^2)} - \frac{\delta z^T \nabla f(\delta z) - f(\delta z)}{f(z)} \right) \cdot P\end{aligned}$$

where the equality on the third line is from (2) of Fact B.6. Our desired result follows from maximizing this ratio and choosing δ to be the maximizer. □

Consistency. We note that Lemma B.7 is unaffected by our changes in Algorithm 3, since the primal growth rate is unaltered, and thus it holds for Algorithm 3 as well.

C PRIMAL-DUAL FRAMEWORK FOR ONLINE COVERING WITH ℓ_q-NORM OBJECTIVES

The online covering problem with convex objectives is a very general and expressive framework that encompasses a variety of problems. However, while our algorithmic framework in Section 3 and Appendix B is general-purpose and can be applied to many such problems, it does require the technical assumption that the gradient of the objective function is monotone, and cannot utilize any inherent structural properties of the problem itself, possibly suffering a loss in generality. Is it possible to specialize our algorithms and analyses for online covering with specific convex objective functions? Can we use properties of these objective functions to obtain improved bounds, or remove the monotone gradient assumption?

In this section, we study a more structured variant of the general online convex covering problem: Online covering problems with ℓ_q-norm objectives. We wish to solve the following problem online:

$$\min \sum_{e=1}^{r} c_e ||x(S_e)||_{q_e} \text{ over } x \ge \mathbf{0} \text{ subject to } Ax \ge 1. \tag{3}$$

where each $S_e \subseteq [n]$ is a subset of indices, $q_e \geq 1$, $c_e \geq 0$, $x(S)$ denote the vector x restricted to indices in S, and $||x(S)||_q$ is the ℓ_q norm, i.e., $\left(\sum_{i \in S} x_i^q\right)^{1/q}$.

Shen and Nagarajan (2020) investigated online covering problems with ℓ_q-norm objectives (3), and presented an $O(\log \kappa d)$-competitive algorithm, where κ is the condition number, defined as the ratio between the largest and the smallest non-zero entry in A, and d is *the maximum between the row sparsity of A and the cardinality of the largest S_e*. Their algorithm is in fact identical to a variant of the algorithm of Buchbinder et al. (2014), which we base our Algorithm 2 on, but they pair the algorithm with a new analysis utilizing the structural properties of (3), removing the assumption in Buchbinder et al. (2014) that the gradient $\nabla f(x)$ is monotone, which does not hold for ℓ_q-norms in general.

The Fenchel-based dual that Shen and Nagarajan (2020) considers is the following:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{m} y_i \\ \text{over} \quad & y \in \mathbb{R}^m_{\geq 0} \\ \text{subject to} \quad & A^T y = \mu \\ & \sum_{e=1}^{r} \mu_e = \mu \\ & ||\mu_e(S_e)||_{p_e} \leq c_e \qquad \forall e \in [r] \\ & \mu_e(\bar{S}_e) = \mathbf{0} \qquad \forall e \in [r] \end{aligned} \tag{6}$$

where p_e satisfies that $\frac{1}{p_e} + \frac{1}{q_e} = 1$ for all $e \in [r]$, i.e., $||\cdot||_{p_e}$ is the dual norm of $||\cdot||_{q_e}$. We remark that (6) is in fact equivalent to (4), with $f(x) = \sum_{e=1}^{r} c_e ||x(S_e)||_{q_e}$. As a result, the weak duality of Fact 3.1 still holds for (3) and (6). We refer the readers to Shen and Nagarajan (2020) (Section 2) for a detailed derivation of this Fenchel dual.

We present a primal-dual learning-augmented algorithm for online covering problems with ℓ_q-norm objectives (3), stated as Algorithm 4, and analyze its performance in Theorem C.1. On a high level, Algorithm 4 is identical to Algorithm 2, only without the step of decrementing the dual variables for tight packing constraints (Line 15 of Algorithm 2). As discussed in Section 3, this modification will violate the exact feasibility of the dual solution, incurring an additive $\log \kappa$ factor loss in the competitiveness of the algorithm. However, it preserves the monotonicity of the dual solution, and is vital for our analysis, which removes the reliance on the assumption on the monotonicity of the gradient $\nabla f(x)$.

Theorem C.1. *For the learning-augmented online covering problem with ℓ_q-norm objectives, there exists an online algorithm that takes a problem instance and an advice x', and generates a solution $\bar{x}$ such that*

$$f(\bar{x}) \leq \min\left\{ O\left(\frac{1}{1-\lambda}\right) f(x') + O(\log \kappa d)\, \mathit{OPT}, O\left(\log \frac{\kappa d}{\lambda}\right) \mathit{OPT} \right\}$$

Additionally, if x' is feasible, i.e., $Ax' \geq \mathbf{1}$, then $f(\bar{x}) \leq O\left(\frac{1}{1-\lambda}\right) f(x')$. Here, $\kappa := a_{\max}/a_{\min}$ is the condition number, $a_{\max} = \max_{i \in [m], j \in [n]}\{a_{ij} | a_{ij} > 0\}$, $a_{\min} = \min_{i \in [m], j \in [n]}\{a_{ij} | a_{ij} > 0\}$, and $d = \max\{\max_{i \in [m]} |\{a_{ij} | a_{ij} > 0\}|, \max_{e \in [r]} |S_e|\}$.

In our analysis, we assume that all of the index sets S_e are disjoint. This allows us to separate the gradient $\nabla f(x)$ into independent terms for each $e \in [r]$. Recall that $f(x) = \sum_{e=1}^{r} c_e ||x(S_e)||_{q_e}$. We argue that this assumption is without loss of generality, and that proving Theorem C.1 with the disjointness assumption suffices to show that Algorithm 4 is consistent and robust even without the disjointness assumption, with the following fact from Shen and Nagarajan (2020):

Fact C.2 (Shen and Nagarajan (2020))**.** *Suppose there is a polynomial-time $O(C)$-consistent $O(R)$-robust algorithm $\mathcal{A}$ for Equation (3) for disjoint sets S_e, then there is a polynomial-time $O(C)$-consistent $O(R)$-robust algorithm for Equation (3) for general instances without the disjointness assumption.*

Algorithm 4 Primal-dual learning-augmented algorithm for online covering with ℓ_q-norm objectives in round t

Input: $x_1, \cdots, x_n$, $y_1, \cdots, y_{t-1}$, μ: current solution, A_{tj} for $1 \le j \le n$: current coefficients, (x', λ): advice and confidence parameter.

Output: Updated x and y_t.

$\tau \leftarrow$ current time.

if $A_t x' \ge 1$ **then** ▷ x' is feasible

  $D_j^{(t)} \leftarrow \frac{\lambda}{a_{tj} d} + \frac{(1-\lambda) x'_j \mathbf{1}_{x_j < x'_j}}{A_t x'_c}$

else

  $D_j^{(t)} \leftarrow \frac{1}{a_{tj} d}$

while $A_t x < 1$ **do**

  for each $j \in [n]$ s.t. $a_{tj} > 0$ **do**

    Increase τ at rate 1.

    Increase x_j at rate

$$\frac{\partial x_j}{\partial \tau} = \frac{a_{tj}}{\nabla_j f(x)} (x_j + D_j^{(t)})$$

  Increase y_t at rate 1.

  $\mu \leftarrow A^T y$.

We outline a proof sketch for Fact C.2 here, and refer the readers to Shen and Nagarajan (2020) for a more detailed proof.

Shen and Nagarajan (2020) proves Fact C.2 via a reduction between a general instance of Equation (3) and one with equivalent optimal value with disjoint sets. For any instance $\mathcal{P}$ with general index sets $\{S_e\}_{e \in [r]}$, we can construct an instance $\mathcal{P}'$ with disjoint sets as follows: for each variable x_j in $\mathcal{P}$, we create variables $x_j^{(1)}, \cdots, x_j^{(r)}$ in $\mathcal{P}'$, where $x_j^{(e)}$ corresponds to the possible occurence of x_j in S_e. Let S'_e consists of the variables $\{x_j^{(e)}\}_{j \in [n]}$, so that $\{S'_e\}_{e \in [r]}$ is a collection of disjoint sets. For any constraint $\sum_j a_{ij} x_j$ in $\mathcal{P}$, we create r^n constraints in $\mathcal{P}'$, each corresponding to a possible combination of the $x_j^{(e)}$ variables, i.e.,

$$\sum_{j=1}^{n} a_{ij} x_j^{(e_j)} \ge 1 \qquad \forall e_1, \cdots, e_n \in [r]$$

It is simple to show that $\mathcal{P}$ and $\mathcal{P}'$ have the same optimal value, and any solution to $\mathcal{P}'$ can be converted to a solution to $\mathcal{P}$ by setting x_j to the minimum value over all copies $\min_e \{x_j^{(e)}\}$, maintaining the same objective value. While the reduction isn't a polynomial one due to the exponential amount of constraints in $\mathcal{P}'$, one can use a separation oracle-based approach as in Alon et al. (2006).

While Shen and Nagarajan (2020) only proved the robustness component of Fact C.2, corresponding to the competitiveness of classical online algorithms without learning-augmentation, we can easily extend their analysis to the consistency as well. Any advice for an instance $\mathcal{P}$ can be mapped to an advice with equivalent value for $\mathcal{P}'$ by duplicating each entry r times.

With this disjointness assumption, made without loss of generality, we proceed to prove the robustness and consistency component of Theorem C.1 separately.

Robustness. We first prove the robustness of Algorithm 4, that $f(\bar{x}) \le O(\log \frac{\kappa d}{\lambda})\mathsf{OPT}$. While Algorithm 4 is similar to Algorithm 2 on a high level, there are some subtle differences in the detailed specifications. We show the following:

- $\bar{x}$ is feasible;
- $(\bar{y}, \mu)$ is $O(\log \frac{\kappa d}{\lambda})$-approximately feasible;

- The primal objective P is at most twice the dual objective D.

Equipped with these subclaims and weak duality, we have

$$P = f(\bar{x}) \leq O(1)D \leq O(\log \frac{\kappa d}{\lambda})\mathsf{OPT}$$

as desired.

The feasibility of the primal variables is straightforward, by definition of the algorithm, as with that of Algorithm 2. We use the following component lemma to bound the ratio between the primal and the dual objective:

Lemma C.3. *Let $\bar{x}$ and $\bar{y}$ be the primal and dual variables at the end of the execution, respectively. The primal objective $f(\bar{x})$ is at most twice the dual objective $\sum_{i=1}^{m} \bar{y}_i$.*

Proof. Consider during round t when the primal constraint $A_t x \geq 1$ arrives. Then,

$$\begin{aligned}\frac{\partial f(x^{(\tau)})}{\partial \tau} &= \sum_{j:a_{tj}>0} \nabla_j f(x^{(\tau)}) \cdot \frac{\partial x_j}{\partial \tau} \\ &= \sum_{j:a_{tj}>0} \left(a_{tj} x_j + \frac{\lambda}{d} + \frac{(1-\lambda) a_{tj} x'_j \mathbf{1}_{x_j < x'_j}}{A_t x'_c} \right) \\ &\leq 1 + \lambda + (1-\lambda) = 2\end{aligned}$$

Since the dual objective increases at rate 1, we conclude as desired. □

The approximate feasibility of the dual variables is the most sophisticated component of the analysis of Algorithm 4. We follow the proof strategy of Shen and Nagarajan (2020). On a high level, in lieu of the assumption that the gradient is monotone, our analysis uses the structural properties of the problem, which the general online convex covering problem lacks, to define a potential function, which we then use to partition the execution of the algorithm into phases, and bound the growth of the dual objective in each phase separately.

Lemma C.4. *The dual solution $\bar{y}, \bar{\mu}$ is $O(\log \frac{\kappa d}{\lambda})$ approximately-feasible, i.e.,*

$$\mu(\cap_e \bar{S}_e) = \mathbf{0}$$

$$||\mu(S_e)||_{p_e} \leq c_e \cdot O(\log \frac{\kappa d}{\lambda}) \qquad \forall e \in [r]$$

Proof. The first feasibility condition $\mu(\cap_e \bar{S}_e) = \mathbf{0}$ follows straightforwardly from the algorithm. Fix any $j \in \cap_e \bar{S}_e$, and consider during round t when the primal constraint $A_t x \geq 1$ arrives. If $a_{tj} = 0$, then incrementing y_t will not cause μ_j to increase; If $a_{tj} > 0$, then $\nabla_j f(x) = 0$ will imply that $\frac{\partial x_j}{\partial \tau} = \infty$, so the algorithm will set x_j to the desired value satisfying the constraint without incrementing y_t. In either case, μ_j is not incremented at all, so that at the end of the algorithm, $\mu(\cap_e \bar{S}_e) = \mathbf{0}$ as desired.

For the second feasibility condition, fix any $e \in [r]$, so that we consider only $f(x) = c_e ||x(S_e)||_{q_e}$. We want to show that $||\mu(S_e)||_{p_e} \leq c_e \cdot O(\log \frac{\kappa d}{\lambda})$. We consider two cases, $q_e = 1$, and the more complicated $q_e > 1$ case.

The case of $q_e = 1$. The part of the objective function corresponding to our chosen value of e reduces to the linear case, in the form of $f(x) = c_e \sum_{j \in S_e} x_j$, and that $\nabla_j f(x) = c_e$. The corresponding $p_e = \infty$, thus it suffices to show that $||\mu(S_e)||_\infty \leq c_e \cdot O(\log \frac{\kappa d}{\lambda})$, or equivalently, $\mu_j \leq c_e \cdot O(\log \frac{\kappa d}{\lambda})$ for all $j \in S_e$.

First, by definition,

$$\frac{\partial x_j}{\partial \mu_j} = \frac{\partial x_j}{\partial \tau} \frac{\partial \tau}{\partial \mu_j} = \frac{a_{tj}(x_j + D_j^{(t)})}{c_e} \cdot \frac{1}{a_{tj}}$$

Notice that the algorithm never increments any x_j past $\frac{1}{a_{\min}}$, since $a_{tj}x_j \geq a_{\min}x_j$ is at most 1 during the algorithm. Thus, we can upper bound the total growth of μ_j over the entire algorithm by

$$\Delta\mu_j \leq \int_0^{\frac{1}{a_{\min}}} \frac{c_e a_{tj}}{a_{tj}(x_j + D_j^{(t)})} dx_j = c_e \log\left(\frac{\frac{a_{tj}}{a_{\min}}}{a_{tj}D_j^{(t)}} + 1\right) \leq c_e \cdot O(\log \frac{\kappa d}{\lambda})$$

as we desire, where the last inequality is due to $\frac{a_{tj}}{a_{\min}} \leq \frac{a_{\max}}{a_{\min}} = \kappa$, and $a_{tj}D_j^{(t)} = \frac{\lambda}{d} + \frac{(1-\lambda)a_{tj}x'_j\mathbf{1}_{x_j<x'_j}}{A_t x'_c} \geq \frac{\lambda}{d}$.

The case of $q_e > 1$. In this case, the objective function corresponding to our chosen value of e is $f(x) = c_e||x(S_e)||_{q_e} = c_e\left(\sum_{k\in S_e} x_k^{q_e}\right)^{1/q_e}$. The corresponding partial gradient is

$$\nabla_j f(x) = c_e \cdot \frac{1}{q_e}\left(\sum_{k\in S_e} x_k^{q_e}\right)^{\frac{1}{q_e}-1} \cdot q_e x_j^{q_e-1} = c_e x_j^{q_e-1}\left(\sum_{k\in S_e} x_k^{q_e}\right)^{\frac{1}{q_e}-1}$$

We use a potential function to bound the total change of μ_j throughout the algorithm. With the potential function, we partition the algorithm's execution into phases, and bound the change in each phase separately, before summing the change up across all phases.

We define the potential function as follows:

$$\Phi = \sum_{j\in S_e} x_j^{q_e}$$

Trivially, Φ is non-decreasing in the primal covering variables x. At the beginning of the algorithm, $\Phi = 0$.

Let phase 0 denote the period of the algorithm when $\Phi \leq \zeta := \left(\frac{\lambda}{a_{\max}d^2}\right)^{q_e}$, and for each $\ell \geq 1$, let phase ℓ denote the period when $\theta^{\ell-1}\zeta \leq \Phi \leq \theta^\ell\zeta$, for some selection of θ that will be determined later. Since x_j will never be incremented past $\frac{1}{a_{\min}}$, we know that $\Phi \leq d\left(\frac{1}{a_{\min}}\right)^{q_e}$ at all times, thus there can be at most $3q_e \log\left(\frac{\kappa d}{\lambda}\right)/\log\theta$ phases.

Note that in Shen and Nagarajan (2020), a similar potential-based strategy is used, but the phases of the algorithm is defined with respect to $\zeta = \left(\frac{1}{a_{\max}d^2}\right)^{q_e}$. The change we make by adding the λ factor to the numerator is necessary to suffer an additive error of only $O(\log\frac{1}{\lambda})$, as opposed to $O(\frac{1}{\lambda})$.

Phase 0. We first bound $\Delta\mu_j$, the change in μ_j, during phase 0. Let α_j denote the value of x_j at the end of phase 0. Then, using a similar strategy to the case of $q_e = 1$, we have

$$\frac{\partial x_j}{\partial \mu_j} = \frac{\partial x_j}{\partial \tau}\frac{\partial \tau}{\partial \mu_j} = \frac{a_{tj}\left(x_j + D_j^{(t)}\right)}{c_e x_j^{q_e-1}\left(\sum_{k\in S_e} x_k^{q_e}\right)^{\frac{1}{q_e}-1}} \cdot \frac{1}{a_{tj}}$$

Rearranging, we obtain

$$\frac{\partial \mu_j}{\partial x_j} = \frac{c_e a_{tj} x_j^{q_e-1}}{\left(\sum_{k\in S_e} x_k^{q_e}\right)^{1-\frac{1}{q_e}} a_{tj}\left(x_j + D_j^{(t)}\right)} \leq \frac{d c_e a_{tj} x_j^{q_e-1}}{\lambda\left(\sum_{k\in S_e} x_k^{q_e}\right)^{1-\frac{1}{q_e}}}$$

where the last inequality holds since $a_{tj}(x_j + D_j^{(t)}) = a_{tj}x_j + \frac{\lambda}{d} + \frac{(1-\lambda)a_{tj}x'_j\mathbf{1}_{x_j<x'_j}}{A_t x'_c} \geq \frac{\lambda}{d}$.

Thus, integrating over the entirety of phase 0, we obtain

$$\begin{aligned}\Delta\mu_j &\le \int_0^{\alpha_j} \frac{dc_e a_{tj} x_j^{q_e-1}}{\lambda\left(\sum_{k\in S_e} x_k^{q_e}\right)^{1-\frac{1}{q_e}}} dx_j \\ &= \frac{dc_e a_{tj}}{\lambda}\int_0^{\alpha_j} \frac{\left(x_j^{q_e}\right)^{1-\frac{1}{q_e}}}{\left(\sum_{k\in S_e} x_k^{q_e}\right)^{1-\frac{1}{q_e}}} dx_j \\ &\le \frac{dc_e a_{tj}}{\lambda}\int_0^{\alpha_j} 1 dx_j = \frac{dc_e a_{tj}}{\lambda}\alpha_j\end{aligned}$$

Since in phase 0, we have $\Phi = \sum_{k\in S_e} x_k^{q_e} \le \left(\frac{\lambda}{a_{\max} d^2}\right)^{q_e}$, it must hold that $a_j \le \frac{\lambda}{a_{\max} d^2}$. Thus

$$\Delta\mu_j \le \frac{dc_e a_{tj}}{\lambda}\alpha_j \le \frac{c_e}{d}$$

Taking the p_e norm over all $j \in S_e$, we have that at the end of phase 0, $||\mu(S_e)||_{p_e} \le ||\mu(S_e)||_1 \le c_e$, since $d \ge \max_e |S_e|$ by definition.

Phase $\ell \ge 1$. We then bound $\Delta\mu_j$ during some phase $\ell \ge 1$. Denote by s_j and t_j the value of x_j at the beginning and the end of the phase, and by Φ_0 and Φ_1 the value of Φ at the beginning and the end, respectively.

Similarly, bounding the rate of change of μ_j with that of x_j yields

$$\frac{\partial\mu_j}{\partial x_j} = \frac{c_e a_{tj} x_j^{q_e-1}}{\left(\sum_{k\in S_e} x_k^{q_e}\right)^{1-\frac{1}{q_e}} a_{tj}\left(x_j + D_j^{(t)}\right)} \le \frac{c_e x_j^{q_e-2}}{\left(\sum_{k\in S_e} x_k^{q_e}\right)^{1-\frac{1}{q_e}}} \tag{7}$$

where the last inequality holds since $a_{tj}(x_j + D_j^{(t)}) = a_{tj}x_j + \frac{\lambda}{d} + \frac{(1-\lambda)a_{tj}x'_j \mathbf{1}_{x_j < x'_j}}{A_t x'_c} \ge a_{tj}x_j$.

Bounding the right-hand side of Equation (7) with $\frac{c_e}{x_j}$ as per our strategy in phase 0, and integrating over the range of x_j yields $\Delta\mu_j \le c_e \log(t_j/s_j)$, which depends on the value of s_j and t_j and is not tight enough. Instead, we use our defined potential Φ, and the fact that $\Phi = \sum_{k\in S_e} x_k^{q_e} \ge \Phi_0$ throughout this phase, to obtain the following bound:

$$\Delta\mu_j \le \int_{s_j}^{t_j} \frac{c_e x_j^{q_e-2}}{\Phi_0^{1-\frac{1}{q_e}}} = \frac{c_e(t_j^{q_e-1} - s_j^{q_e-1})}{(q_e-1)\Phi_0^{1-\frac{1}{q_e}}} \tag{8}$$

Here, our assumption that $q_e > 1$ is required to evaluate the integral.

Towards bounding the p_e norm, we have

$$(\Delta\mu_j)^{p_e} \le \frac{c_e^{p_e}(t_j^{q_e-1} - s_j^{q_e-1})^{p_e}}{(q_e-1)^{p_e}\Phi_0^{p_e(1-\frac{1}{q_e})}}$$

We use two mathematical facts to simplify Equation (8). The first is the fact that $a^{p_e} + b^{p_e} \le (a+b)^{p_e}$ for any $a, b \ge 0$ and $p_e \ge 1$. Setting $a = (t_j^{q_e-1} - s_j^{q_e-1})$ and $b = s_j^{q_e-1}$ yields $(t_j^{q_e-1} - s_j^{q_e-1})^{p_e} \le t_j^{(q_e-1)p_e} - s_j^{(q_e-1)p_e}$.

The second fact we use is that by definition, $\frac{1}{p_e} + \frac{1}{q_e} = 1$. We have both $p_e(1-\frac{1}{q_e}) = 1$, and $(q_e-1)p_e = q_e$.

Thus, we can simplify Equation (8) to obtain

$$(\Delta\mu_j)^{p_e} \le \frac{c_e^{p_e}(t_j^{q_e-1} - s_j^{q_e-1})^{p_e}}{(q_e-1)^{p_e}\Phi_0^{p_e(1-\frac{1}{q_e})}} \le \frac{c_e^{p_e}(t_j^{q_e} - s_j^{q_e})}{(q_e-1)^{p_e}\Phi_0}$$

Finally, summing over all indices $j \in S_e$, we have

$$\begin{aligned}\sum_{j\in S_e}(\Delta\mu_j)^{p_e} &\le \sum_{j\in S_e}\frac{c_e^{p_e}(t_j^{q_e}-s_j^{q_e})}{(q_e-1)^{p_e}\Phi_0}\\ &= \frac{c_e^{p_e}}{(q_e-1)^{p_e}\Phi_0}\left(\sum_{j\in S_e}t_j^{q_e}-\sum_{j\in S_e}s_j^{q_e}\right)\\ &= \frac{c_e^{p_e}}{(q_e-1)^{p_e}\Phi_0}(\Phi_1-\Phi_0)\\ &\le \frac{c_e^{p_e}(\theta-1)}{(q_e-1)^{p_e}}\end{aligned}$$

where the equality on the third line is by definition of Φ and selection of s_j, t_j, Φ_0, Φ_1; and the inequality on the fourth line is by our definition of phases, that $\Phi_0 = \theta^{\ell-1}\zeta$ and $\Phi_1 = \theta^{\ell}\zeta$.

Combining over all phases. Let $\mu^{(\ell)}$ denote the increment of the vector μ during phase ℓ, and let L denote the total amount of phases. Then $\mu = \sum_{\ell=0}^{L}\mu^{(\ell)}$. By the triangle inequality for ℓ_q-norms, we have

$$\begin{aligned}||\mu||_{p_e} &\le \sum_{\ell=0}^{L}||\mu^{(\ell)}||_{p_e}\\ &\le c_e + \left(\frac{c_e^{p_e}(\theta-1)}{(q_e-1)^{p_e}}\right)^{\frac{1}{p_e}}\cdot 3q_e\frac{\log\frac{\kappa d}{\lambda}}{\log\theta}\\ &= c_e\cdot\left(1+\frac{3q_e(\theta-1)^{1/p_e}}{(q_e-1)\log\theta}\cdot\log\frac{\kappa d}{\lambda}\right)\end{aligned}$$

by our individual bounds for each phase, and the fact that $L \le 3q_e\log(\frac{\kappa d}{\lambda})/\log\theta$.

Choosing the value of θ. It remains to choose an appropriate value of θ for any $q_e > 1$, such that $\frac{3q_e(\theta-1)^{1/p_e}}{(q_e-1)\log\theta} \le O(1)$, or equivalently, $||\mu||_{p_e} \le c_e\cdot O(\log\frac{\kappa d}{\lambda})$, as we desire.

For $q_e \ge 2$, we can simply choose $\theta = 2$, which gives us

$$\frac{3q_e(\theta-1)^{1/p_e}}{(q_e-1)\log\theta} \le 6$$

as desired.

For $1 < q_e < 2$, we choose $\theta = 1 + (q_e-1)^{-\varepsilon p_e}$, where $\varepsilon = \frac{1}{-\log(q_e-1)} > 1$. Then

$$\begin{aligned}\frac{3q_e(\theta-1)^{1/p_e}}{(q_e-1)\log\theta} &\le \frac{3q_e(\theta-1)^{1/p_e}}{(q_e-1)\log(q_e-1)^{-\varepsilon p_e}} \qquad \text{since } \theta \ge (q_e-1)^{-\varepsilon p_e}\\ &= \frac{3q_e}{q_e-1}\cdot\frac{(q_e-1)^{-\varepsilon}}{-\varepsilon p_e\log(q_e-1)}\\ &= \frac{3q_e}{q_e-1}\cdot\frac{e}{p_e}\\ &\le 9 \qquad \text{since } p_e(q_e-1)=q_e\end{aligned}$$

as desired.

Thus, for all values of $q_e > 1$, we have that

$$||\mu||_{p_e} \le c_e\cdot\left(1+9\log\frac{\kappa d}{\lambda}\right) = c_e\cdot O(\log\frac{\kappa d}{\lambda})$$

which concludes the proof. □

We remark that the component of our above proof for phase $\ell \geq 1$ is in fact identical to that of Shen and Nagarajan (2020), and that introducing learning-augmentation with the confidence parameter λ does not modify the proof explicitly. However, since we defined ζ as $\left(\frac{\lambda}{a_{\max} d^2}\right)^{q_e}$, our analysis sums over a different amount of total phases, dependent on λ.

Consistency. We remark that Lemma B.7 holds independent of the dual update theme and the monotone gradient assumption. Thus, Lemma B.7 still holds for Algorithm 4.

D APPLICATIONS

D.1 Applications of Concave Packing

In this section, we present several direct applications of general interest of our switching-based algorithmic framework where Theorem 2.1 immediately gives non-trivial results, in some cases even optimal, up to constant factors.

D.1.1 Online Packing Linear Programming

The online packing linear programming (LP) problem is a special case of (1) where the objective $g(y)$ is the linear function $\sum_{i=1}^{m} y_i = \mathbf{1}^T y$.

$$\max \mathbf{1}^T y \text{ over } y \in \mathbb{R}^m_{\geq 0} \text{ subject to } A^T y \leq b. \tag{9}$$

The seminal work of Buchbinder and Naor (2009a) presents the following theorem.

Theorem D.1 (Buchbinder and Naor (2009a))**.** *For any $B > 0$, there exists a B-competitive algorithm for online packing LP (9) such that for each constraint $j \in [n]$ it holds:*

$$\sum_{i=1}^{m} a_{ij} y_i = b_j \cdot O\left(\frac{\log n\kappa_j}{B}\right),$$

where $\kappa_j = a_j(\max)/a_j(\min)$, $a_j(\max) = \max_{i=1}^m\{a_{ij}\}$, and $a_j(\min) = \min_{i=1}^m\{a_{ij} | a_{ij} > 0\}$.

For impossibility results, it is further shown in Buchbinder and Naor (2009a) that for any $B > 0$, there exists a single-constraint instance for the online packing LP problem, such that any B-competitive algorithm must violate the constraint by a factor of $\omega(\log n\kappa/B)$.

The following corollary holds by Theorems 2.1 and D.1. Here, $\kappa := \max_{j=1}^n\{\kappa_j\}$.

Corollary D.2. *There exists a $\frac{1}{1-\lambda}$-consistent, $\frac{B}{\lambda}$-robust, and $O(\log n\kappa/B)$-feasible learning-augmented online algorithm for online packing LP (9) that takes parameter $B > 0$, $\lambda \in [0, 1]$, and an advice which is $O(\log n\kappa/B)$-feasible.*

Corollary D.2 is optimal in consistency, robustness, and feasibility, up to constant factors. We note that the parameter κ is non-decreasing over time. It is possible that Algorithm 1 enters line 6 in earlier rounds and then enters line 4 in later rounds when $\beta = \Theta(\log n\kappa/B)$ increases.

In the following subsections, we present other specific applications for learning-augmented online packing LP.

D.1.2 Online Knapsack

In this section, we apply Algorithm 1 to the classical online *knapsack* problem. In the most basic formulation of this problem, a sequence of m items arrive online, and each item i has value v_i and weight w_i. Our goal is to pack a selected set of items with as much value as possible into a knapsack of capacity C, deciding

irrevocably whether to include each item as soon as it arrives online, without the total weight exceeding the capacity. Formulated as an online packing linear program, we obtain

$$\max \mathbf{1}^T y \text{ over } y \in \mathbb{R}^m_{\geq 0} \text{ subject to } \sum_{i=1}^{m} \frac{w_i y_i}{v_i} \leq C.$$

in which the elements y_i together with it's value v_i and weight w_i arrive online.

As a corollary to Theorem 2.1, it follows that our simple algorithm can be applied to the knapsack problem to obtain asymptotically optimal performance in both consistency and robustness.

Corollary D.3. *Given any α-competitive algorithm $\mathcal{O}$ for the online knapsack problem, Algorithm 1 using $\mathcal{O}$ as a subroutine implies a $\frac{1}{1-\lambda}$-consistent with 1-feasible advice, $\frac{\alpha}{\lambda}$-robust, and $(2-\lambda)$-feasible learning-augmented online algorithm.*

The knapsack problem is regarded as one of the most fundamental packing problems in computer science and operations research along with its many variants. It has been extensively studied in the offline setting (Martello and Toth, 1987) (see Salkin and De Kluyver (1975) for a survey) and the online setting (Marchetti-Spaccamela and Vercellis, 1995; Ma et al., 2019). Recent work in the learning-augmented setting includes Daneshvaramoli et al. (2023); Im et al. (2021); Boyar et al. (2022), which study the online knapsack problems with learning augmentation from many angles, such as frequency predictions, threshold predictions, and unit profit settings.

We point out that our advice model on the values of the arriving dual variables generalizes most, if not all, forms of advice studied in prior literature. For any advice suggesting a course of action, such as taking an arriving item into the knapsack (or doing so with some probability, in the case of randomized algorithms), we can process such advice by setting the corresponding entry in y' to its value (resp. some fraction of its value).

D.1.3 Online Resource Management Benefit

In this section, we apply Algorithm 1 to the online resource management benefit problem (Leonardi and Marchetti-Spaccamela, 1995). For this problem, we have n distinct resources. Each resource $j \in [n]$ can be assigned a maximum capacity c_j. Here, n and each c_j are given in advance. A sequence of m jobs is presented online, one job at a time, where m can be unknown. Job $i \in [m]$ has a benefit of w_i and can be scheduled with r_i different alternatives. We use $a_{ij}^{(k)}$ for $k \in [r_i]$ to denote the amount of resource j necessary to schedule job i with alternative k. Upon the arrival of job i, w_i, r_i, and $a_{ij}^{(k)}$ are revealed, and one must schedule job i to different alternatives irrevocably. This problem has the following LP formulation:

$$\begin{aligned}
\max_{y} \quad & \sum_{i=1}^{m} \sum_{k=1}^{r_i} y_i^{(k)} & \\
\text{subject to} \quad & \sum_{i=1}^{m} \sum_{k=1}^{r_i} \frac{a_{ij}^{(k)} y_i^{(k)}}{w_i} \leq c_j & \forall j \in [n], \\
& \sum_{k=1}^{r_i} \frac{y_i^{(k)}}{w_i} \leq 1 & \forall i \in [m], \\
& y_i^{(k)} \geq 0 & \forall i \in [m], k \in [r_i].
\end{aligned}$$

Here, $y_i^{(k)}$ denotes the fraction of job i assigned to alternative k. Upon the arrival of job $i \in [m]$, the decision $y_i^{(k)}$ is irrevocable.

In Leonardi and Marchetti-Spaccamela (1995), it is assumed that the benefit of each job $w_i \in [1, W]$ with $W \geq 1$ and for every non-zero coefficient $a_{ij}^{(k)}$, the ratio $a_{ij}^{(k)}/c_j \in [1/P, 1]$ with $P \geq 1$. Leonardi and Marchetti-Spaccamela (1995) presents an $O(\log nWP)$-competitive algorithm and shows it is the best possible. As

a corollary, it follows that with advice, we can achieve the best outcome either obtained from an online algorithm or the advice by losing a constant factor, in a black box manner.

Corollary D.4. *Given any α-competitive algorithm $\mathcal{O}$ for the online resource management problem, Algorithm 1 using $\mathcal{O}$ as a subroutine implies a $\frac{1}{1-\lambda}$-consistent with 1-feasible advice, $\frac{\alpha}{\lambda}$-robust, and $(2-\lambda)$-feasible learning-augmented online algorithm.*

D.1.4 Online Throughput Maximization

In the Online Throughput Maximization problem (Buchbinder and Naor, 2006; Awerbuch et al., 1993), we are given a directed or undirected graph $G = (V, E)$. Each edge has a capacity $c : E \to \mathbb{R}_{\geq 0}$. A set of requests $(s_i, t_i) \in V \times V$ arrive online, one at a time. Upon the arrival of request i, the algorithm irrevocably chooses (fractionally) an s_i-t_i path and allocates a total bandwidth of one unit. The objective is to maximize the throughput. The following packing LP describes this problem.

$$\begin{aligned} \max_{y} \quad & \sum_{i} \sum_{P \in \mathcal{P}_i} y_{i,P} \\ \text{subject to} \quad & \sum_{P \in \mathcal{P}_i} y_{i,P} \leq 1 && \forall i, \\ & \sum_{i} \sum_{P \in \mathcal{P}_i : e \in P} y_{i,P} \leq c(e) && \forall e \in E, \\ & y_{i,P} \leq 0 && \forall i \quad \forall P \in \mathcal{P}_i, \end{aligned}$$

where $\mathcal{P}_i$ denotes the set of all the s_i-t_i paths. Here, $y_{i,P}$ indicates the amount of flow on the path P. The first set of constraints captures that each request requires a unit of bandwidth. The second set of constraints captures that the load on each edge does not exceed the edge capacity.

An $O(1)$-competitive $O(\log |V|)$-feasible online algorithm is presented in Buchbinder and Naor (2006), which together with our result implies the following.

Corollary D.5. *There exists a $\frac{1}{1-\lambda}$-consistent with $O(\log |V|)$-feasible advice, $O(\frac{1}{\lambda})$-robust, and $O(\log |V|)$-feasible learning-augmented algorithm for the online throughput maximization problem.*

D.1.5 Online Network Utility Maximization

In the online network utility maximization (ONUM) problem (Cao et al., 2022), we are given a directed or undirected graph $G = (V, E)$. Each edge has a capacity of one unit. A set of requests $(s_i, t_i) \in V \times V$ arrive online one at a time. Each request i is associated with a budget $b_i \in \mathbb{R}_{>0}$, a monotone concave utility function $g_i : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with $g(0) = 0$, and a specified s_i-t_i path P_i. Upon the arrival of request i, one must decide irrevocably its allocation rate without exceeding the budget b_i. The goal is to approximately maximize the total utility under the online allocation requirement. This problem has the following packing formulation.

$$\begin{aligned} \max_{y} \quad & \sum_{i} g(y_i) \\ \text{subject to} \quad & \sum_{i : e \in P_i} y_i \leq 1 \quad \forall e \in E, \\ & y_i \in [0, b_i] && \forall i. \end{aligned}$$

In Cao et al. (2022), a sufficient condition of having a roughly linear (in $|E|$) competitive algorithm for ONUM is presented. Together with our results, this implies the following corollary.

Corollary D.6. *Given any α-competitive algorithm $\mathcal{O}$ for ONUM, Algorithm 1 using $\mathcal{O}$ as a subroutine implies a $\frac{1}{1-\lambda}$-consistent with 1-feasible advice, $\frac{\alpha}{\lambda}$-robust, and $(2-\lambda)$-feasible learning-augmented online algorithm.*

D.1.6 Online Optimization under Inventory Constraints

The online optimization with inventory constraints (OOIC) problem is introduced in Lin et al. (2019). In this problem, a decision maker sells inventory across an interval of T discrete time slots to maximize the total revenue. At time $t \in [T]$, the decision maker observes the revenue function $g_t : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ and makes an irrevocable decision on the quantity y_t. Upon choosing y_t, the decision maker receives revenue $g_t(y_t)$. The goal is to approximately maximize the total revenue, while respective the inventory constraint $\sum_{t \in T} y_t \leq \Delta$ with a given parameter $\Delta > 0$. It is further assumed that for all $t \in [T]$, g_t has the following nice properties: (1) g_t is concave, increasing, and differentiable over $[0, \Delta]$, (2) $g_t(0) = 0$, and (3) $g_t'(0) > 0$ and $g'(0) \in [m, M]$ for some $M \geq m > 0$. OOIC has the following packing formulation:

$$\begin{aligned} \max_{y} \quad & \sum_{t \in [T]} g_t(y_t) \\ \text{subject to} \quad & \sum_{t \in [T]} y_t \leq \Delta, \\ & y_t \geq 0 \qquad \forall t \in [T]. \end{aligned}$$

Lin et al. (2019) presented a tight π-competitive algorithm for OOIC, where $\pi \in [\log(M/m)+1, \eta(\log(M/m)+1)]$ and $\eta := \sup_{g, x \in [0,\Delta]} \{g'(0)x/g(x)\}$. Together with our results, this implies the following corollary.

Corollary D.7. *Given any α-competitive algorithm $\mathcal{O}$ for OOIC, Algorithm 1 using $\mathcal{O}$ as a subroutine implies a $\frac{1}{1-\lambda}$-consistent with 1-feasible advice, $\frac{\alpha}{\lambda}$-robust, and $(2-\lambda)$-feasible learning-augmented online algorithm.*

D.2 Application of Convex Covering: Online Mixed Covering and Packing

In this section, we present an application of our PDLA framework for online convex covering problems to online mixed covering and packing. We remark that our application can be employed to design learning-augmented online algorithms for a vast list of problems that can be formulated as mixed covering and packing problems, including online capacity-constrained facility location, online capacitated multicast, and different variants of online set cover problems. We refer the reader to Buchbinder et al. (2014) for a more detailed description of the applications to these problems.

In the mixed covering and packing problem, we have k linear objectives. Let $x \in \mathbb{R}^n_{\geq 0}$ be the decision variables. The vector of the linear objectives is captured by Bx for a matrix $B \in \mathbb{R}^{k \times n}_{\geq 0}$. The goal is to minimize the ℓ_q-norm of the linear objectives subject to the covering constraint $Ax \geq 0$ where $q \geq 1$ and $A \in \mathbb{R}^{m \times n}_{\geq 0}$. This problem can be formulated as

$$\text{minimize } ||Bx||_q \text{ over } x \in \mathbb{R}^n_{\geq 0} \text{ subject to } Ax \geq \mathbf{1}. \tag{10}$$

In the online problem, the matrix B is given offline, and the rows of A arrive one at a time where m is unknown. The goal is to update x in a non-decreasing manner to satisfy each arriving constraint and approximately minimize the objective. In the learning-augmented problem, we are also given the advice x'. Let OPT denote the optimal objective value of (10). Corollary B.8 implies the following.

Corollary D.8. *For the learning-augmented online mixed covering and packing problem (10), there exists an online algorithm that takes a problem instance and an advice x', and generates a solution $\bar{x}$ such that*

$$||B\bar{x}||_q \leq \min \left\{ O\left(\frac{1}{1-\lambda}\right) ||Bx'||_q + O(q \log d)\mathsf{OPT}, O\left(q \log \frac{d}{\lambda}\right) \mathsf{OPT} \right\}$$

Additionally, if x' is feasible, i.e., $Ax' \geq \mathbf{1}$, then $||B\bar{x}||_q \leq O\left(\frac{1}{1-\lambda}\right) ||Bx'||_q$. Here, $d = \max_{i \in [m]} |\{a_{ij} | a_{ij} > 0\}|$ is the row sparsity of the constraint matrix A.

Proof. Let $f(x) = ||Bx||_q^q$.[3] We consider the optimization problem

$$\min f(x) \text{ over } x \in \mathbb{R}^n_{\geq 0} \text{ subject to } Ax \geq \mathbf{1} \tag{11}$$

and then take the q-th root of the objective.

Let b_{ij} denote the i-th row j-th entry of B and B_i denote the i-th row of B, we have that

$$f(z) = \sum_{i=1}^{k} (B_i z)^q \,, f(\delta z) = \delta^q \sum_{i=1}^{k} (B_i z)^q \,,$$

$$\nabla_j f(z) = q \left(\sum_{i=1}^{k} b_{ij} \, (B_i z)^{q-1} \right),$$

and

$$\nabla_j f(\delta z) = q\delta^{q-1} \left(\sum_{i=1}^{k} b_{ij} \, (B_i z)^{q-1} \right).$$

This implies that $\nabla_j f(\delta z)/\nabla_j f(z) = \delta^{q-1}$ for all $j \in [n]$, $f(\delta z) = \delta^q f(z)$, and thus

$$z^T \nabla f(\delta z) = q\delta^{q-1} \sum_{i=1}^{k} \sum_{j=1}^{n} b_{ij} z_j \, (B_i z)^{q-1} = q\delta^{q-1} \sum_{i=1}^{k} (B_i z)^q = q\delta^{q-1} f(z)$$

for all $z \in \mathbb{R}_{\geq 0}$. We use Algorithm 3 with (6) and have that

$$\begin{aligned}
\frac{1}{R(\lambda)} &= \max_{\delta > 0} \min_{z} \left(\frac{\min_\ell \left\{ \frac{\nabla_\ell f(\delta z)}{\nabla_\ell f(z)} \right\}}{4 \log(1 + \frac{2}{\lambda} d^2)} - \frac{\delta z^T \nabla f(\delta z) - f(\delta z)}{f(z)} \right) \\
&= \max_{\delta > 0} \left(\frac{\delta^{q-1}}{4 \log(1 + \frac{2}{\lambda} d^2)} - q\delta^q + \delta^q \right) \\
&= \frac{1}{\left(4q \log(1 + \frac{2}{\lambda} d^2)\right)^q}
\end{aligned}$$

when $\delta = 1/(q \log(1 + 2d^2/\lambda))$. Let OPT^q be the optimal objective value for (11). We have that

$$||B\bar{x}||_q^q \leq \min \left\{ O\left(\frac{1}{1-\lambda}\right) ||Bx'||_q^q + O((q \log d)^q) \mathsf{OPT}^q, O\left(\left(q \log \frac{d}{\lambda} \right)^q \right) \mathsf{OPT}^q \right\}$$

which implies

$$\begin{aligned}
O\left(\frac{1}{1-\lambda}\right) ||Bx'||_q^q + O(q \log d) \mathsf{OPT}^q &\leq O\left(\left(\frac{1}{1-\lambda} \right)^q \right) ||Bx'||_q^q + O(q \log d) \mathsf{OPT}^q \\
&\leq \left(O\left(\frac{1}{1-\lambda}\right) ||Bx'||_q + O(q \log d) \mathsf{OPT} \right)^q,
\end{aligned}$$

thus,

$$||B\bar{x}||_q \leq \min \left\{ O\left(\frac{1}{1-\lambda}\right) ||Bx'||_q + O(q \log d) \mathsf{OPT}, O\left(q \log \frac{d}{\lambda} \right) \mathsf{OPT} \right\}$$

as required. □

[3] One might ask if we can use $||Bx||_q$ as the objective. The gradient of $||Bx||_q$ is not monotone, but we need the assumption that the gradient is monotone in (5) for the analysis to hold.

D.3 Applications of ℓ_q-Norm Covering: Online Buy-at-Bulk Network Design

In this section, we present an application of our PDLA framework for online covering problems with ℓ_q-norm objectives to the online buy-at-bulk network design problem.

In the buy-at-bulk network design problem, we are given a directed or undirected n-vertex graph $G = (V, E)$. Each edge $e \in E$ is associated with a monotone subadditive cost function $g_e : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with $g_e(0) = 0$. We are also given a set of k terminal pairs $\{(s_i, t_i) \mid i \in [k]\} \subseteq V \times V$. The goal is to find a collection of s_i-t_i paths P_i for each $i \in [k]$ such that the overall cost $\sum_{e \in E} g_e(load_e)$ is minimized, where $load_e$ is the number of paths using e. By paying an approximation factor of two, the objective can be written in terms of the *upfront cost* c_e and *pay-per-use cost* ℓ_e, i.e., $\sum_{e \in \cup P_i} c_e + \sum_{e \in E} \ell_e \cdot load_e$. In the online setting, the graph and the cost functions are given in advance. The terminal pairs arrive one at a time. The goal is to irrevocably assign an s_i-t_i path upon the arrival of pair i while approximately minimizing the overall cost. In the learning-augmented setting, upon the arrival of pair i, we are also given a s_i-t_i path as advice.

The seminal framework for online buy-at-bulk in Chakrabarty et al. (2018) gave a modular online algorithm with competitive ratio $O(\alpha\beta\gamma \log^5 n)$. The ratio was later improved by Shen and Nagarajan (2020) to $O(\alpha\beta\gamma \log^3 n)$. Here,

- α denotes the *junction tree* approximation ratio. A junction tree rooted at a vertex $r \in V$ is a union of s_i-t_i paths where each path passes through r. A junction tree solution is a collection of junction trees rooted at different vertices, such that each terminal pair (s_i, t_i) is connected by using one of the junction trees with a specified root $r \in V$. The cost of a junction tree solution is not shared across junction trees with different roots. The value α is the worst-case ratio between the cost of an optimal junction tree solution and the optimum.
- β is the integrality gap of the natural LP relaxation for single-source buy-at-bulk instances.
- γ is the competitive ratio of an online algorithm for single-source buy-at-bulk instances.

We now describe the convex optimization formulation used in Shen and Nagarajan (2020). Let $T = \{s_i, t_i \mid i \in [k]\}$ denote the set of terminal vertices. Let $x_{r,u,e}$ denote the *flow* on edge $e \in E$ connecting u and r in a junction tree rooted at $r \in V$. For any $S \subseteq E$, $r \in V$, and $u \in T$, we define $x_{r,u}(S) := \sum_{e \in S} x_{r,u,e}$. This notion is useful when S forms a u-r cut while considering a junction tree rooted at r. The optimization problem is as follows.

$$
\begin{aligned}
\min_{x} \quad & \sum_{r \in V} \sum_{e \in E} c_e \max_{u \in T} \{x_{r,u,e}\} + \sum_{r \in V} \sum_{e \in E} \ell_e \sum_{u \in T} x_{r,u,e} \\
\text{s.t.} \quad & \sum_{r \in R_s} x_{r,s_i}(S_r) + \sum_{r \in R_t} x_{r,t_i}(T_r) \geq 1 \\
& \qquad \forall i \in [k], \quad \forall (R_s, R_t) \text{ partition of } V, \\
& \qquad \forall S_r : s_i\text{-}r \text{ cut}, \forall r \in R_s, \\
& \qquad \forall T_r : t_i\text{-}r \text{ cut}, \forall r \in R_t, \\
& x_{r,u,e} \geq 0 \qquad\qquad \forall e \in E, \quad \forall u \in T, \quad \forall r \in V.
\end{aligned}
\tag{12}
$$

Consider a feasible 0-1 solution f. In the objective, for each junction tree rooted at $r \in V$, the upfront cost paid for using edge $e \in E$ is $c_e \max_{u \in T} \{x_{r,u,e}\}$ since $x_{r,u,e}$ would indicate if e is used in the junction tree connecting r and u; the pay-per-use cost is $\ell_e \sum_{u \in T} x_{r,u,e}$ since $\sum_{u \in T} x_{r,u,e}$ counts the number of paths using e. The first constraint says that s_i and t_i must be fractionally connected to the roots of different junction trees with a total flow value of at least one.

An equivalent LP formulation for (12) can be solved online by paying a factor of $O(\log^3 n)$. The improvement in Shen and Nagarajan (2020) is done as follows. The term $\max_{u \in T} \{x_{r,u,e}\}$ can be written as the ℓ_∞-norm of x. Note that $\ell_{\log n}$-norm is a constant approximation for ℓ_∞-norm. The objective can be reformulated as the sum of $\ell_{\log n}$-norm and ℓ_1-norm and the constraint is 0-1. This is in the form of (3) with $\kappa = 1$, which implies an $O(\log n)$ competitive algorithm as presented in Shen and Nagarajan (2020).

Now we consider the learning-augmented setting. Recall that upon the arrival of pair i, we are also given an s_i-t_i path as advice. Consider the solution constructed by taking the union of the s_j-t_j paths for $j \in [i]$. We convert the advice solution into a junction tree solution by paying a factor of α using previous frameworks (Antonakopoulos, 2010; Chekuri et al., 2010). This introduces advice for the 0-1 flow variables termed as x'. Let ADV denote the cost of the advice. We have that $f(x') \leq \alpha\mathsf{ADV}$. Using Theorem C.1 and the rounding scheme for (12) in Chakrabarty et al. (2018) which pays a factor of $O(\log^2 n)$, we have the following.

Corollary D.9. *For the learning-augmented online buy-at-bulk problem, there exists an online algorithm that takes a problem instance and advice consisting of s_i-t_i paths and generates an online solution with cost at most*

$$\min\left\{O\left(\frac{\alpha}{1-\lambda}\right)\mathit{ADV}, O\left(\alpha\beta\gamma\log^2 n\log\frac{n}{\lambda}\right)\mathit{OPT}\right\}.$$

We note that using a minimum cut algorithm as a separation oracle, the learning-augmented algorithm runs in polynomial time. Here we state the high-level idea that slightly modifies Algorithm 4. In the line 6 while loop, whenever $A_t x < 1/2$, we increment x until $A_t x \geq 1$. The while loop is not entered if $A_t x \geq 1/2$. This guarantees that the growth of x is sufficiently large so that the number of iterations entering the line 6 while loop is $O(E)$. We return $2x$ as the final solution. Since the objective function f is convex and $f(\mathbf{0}) = 0$, the solution is feasible and we pay a constant factor for the objective.