Online Smoothed Demand Management

ADAM LECHOWICZ, University of Massachusetts Amherst, USA
NICOLAS CHRISTIANSON, Stanford University, USA
MOHAMMAD HAJIESMAILI, University of Massachusetts Amherst, USA
ADAM WIERMAN, California Institute of Technology, USA
PRASHANT SHENOY, University of Massachusetts Amherst, USA

We introduce and study a class of online problems called online smoothed demand management (OSDM), motivated by paradigm shifts in grid integration and energy storage for large energy consumers such as data centers. In OSDM, an operator makes two decisions at each time step: an amount of energy to be purchased, and an amount of energy to be delivered (i.e., used for computation). The difference between these decisions charges (or discharges) the operator's energy storage (e.g., a battery). Two types of demand arrive online: base demand, which must be covered at the current time, and flexible demand, which can be satisfied at any time steps before a demand-specific deadline Δ_t. The operator's goal is to minimize a cost (subject to the constraints above) that combines a cost of purchasing energy, a cost for delivering energy (if applicable), and smoothness penalties on the purchasing and delivery rates to discourage fluctuations and encourage "grid healthy" decisions. OSDM generalizes several problems in the online algorithms literature while being the first to fully model applications of interest. We propose a competitive algorithm for OSDM called PAAD (partitioned accounting & aggregated decisions) and show it achieves the optimal competitive ratio. To overcome the pessimism typical of worst-case analysis, we also propose a novel learning framework for OSDM that provides guarantees on the worst-case competitive ratio (i.e., to provide robustness against nonstationarity) while allowing end-to-end differentiable learning of the best algorithm on historical instances of the problem. We evaluate our algorithms in a case study of a grid-integrated data center with battery storage, showing that PAAD effectively solves the problem and end-to-end learning achieves substantial performance improvements compared to PAAD. [1]

1 Introduction

The proliferation of renewable energy sources and grid-scale energy storage, the growth of power-hungry AI data centers, and the implementation of dynamic pricing schemes like time-of-use tariffs are fundamentally reshaping the landscape of how energy and compute infrastructures interact. These changes are prompting a paradigm shift in how systems interact with power grids, shifting from a model of passive consumption to one of *active, intelligent demand management*. The core challenge in demand management from an energy perspective is to better align the consumption of energy (e.g., from the grid) with external signals such as time-of-use prices, while meeting certain performance (e.g,. service-level agreement (SLA)) objectives on the demand side. Concurrently, there is a need to consider the "smoothness" of these decisions: recent reports suggest that data centers' sudden large fluctuations in energy use are impacting power quality in U.S. grids [69], and sudden changes in load are known to cause cascading failures and even blackouts on the grid, such as the 2025 Iberian Peninsula blackout [72]. This prompts an additional objective of *smooth decisions* (e.g., gradual ramping up and down).

Many techniques have been developed for this class of problems, with solution designs that are usually tailored to specific applications such as energy storage management [55] and HVAC control [41], among others. These techniques often rely on machine learning somewhere in the

[1] A short abstract of this work appeared at the ACM SIGMETRICS 2026 conference.

Authors' Contact Information: Adam Lechowicz, University of Massachusetts Amherst, USA, alechowicz@cs.umass.edu; Nicolas Christianson, Stanford University, USA, christianson@stanford.edu; Mohammad Hajiesmaili, University of Massachusetts Amherst, USA, hajiesmaili@cs.umass.edu; Adam Wierman, California Institute of Technology, USA, adamw@caltech.edu; Prashant Shenoy, University of Massachusetts Amherst, USA, shenoy@cs.umass.edu.

pipeline, e.g., to provide predictions for a model predictive controller [66, 78], inform a day-ahead provisioning problem [90], or to directly make decisions (e.g., using a reinforcement learning agent) [6]. While these problem-specific solutions show promise, they are seldom deployed in practice. One reason is that environments such as electricity markets are inherently nonstationary, making it difficult to develop heuristics that deal well with distribution shift [47] or come with rigorous performance guarantees. Another reason is that these approaches are typically developed with a single end application in mind, making generalization to other settings difficult.

On the other side of the spectrum, there is a rich body of work in online algorithms [16, 24, 46, 88, 89] on related problems that are sufficiently abstract to model many different applications, and have been studied under frameworks such as competitive analysis, which provides rigorous performance guarantees. Despite this, however, there remains a gap between the existing models studied in the online algorithms literature and the complexity of real-world demand management problems—as of yet, no models exist that fully capture the moving pieces necessary to study the type of problem that arises in, e.g., a grid-integrated data center with local energy storage.

Informed by these challenges, we introduce and study a new class of online optimization problems called **online smoothed demand management (OSDM)**. OSDM models the challenge faced by an operator who satisfies a stream of incoming demand by purchasing energy from a market with fluctuating prices (e.g., an electricity grid). We model several features motivated by practice, namely: (i) a mix of inflexible (base) and deferrable (flexible) demands, each with their own unique SLA constraints; (ii) opportunities to arbitrage using local energy storage (e.g., a battery), which may come at a cost; and (iii) a few smoothing cost models for "unsmooth" behavior, penalizing large changes in both energy purchasing (ramping) and demand-side delivery. These costs capture some desired behavior for any algorithmic scheme from both the energy provider (e.g., grid) perspective, as well as the consumer's perspective. In studying OSDM, we answer the following question:

Is it possible to design online algorithms for OSDM *that manage the challenges of flexibility, uncertain demand, and penalties for unsmooth decisions while providing meaningful competitive guarantees?*

Competitive (i.e., worst-case) guarantees are often overly pessimistic in practice, since real-world problem instances are not adversarial. In recent years, the area of *learning-augmented algorithms* [42, 53, 61] has expanded the online algorithms literature with techniques that can take additional information as input (e.g., predictions about the instance) to improve average-case performance while retaining worst-case guarantees for robustness against nonstationarities. While this paradigm has successfully been applied to problems that share similarities with OSDM [46, 81], traditional learning-augmented designs make strong assumptions about the availability and learnability of *suitable predictions*, which face challenges in OSDM due to the multiple kinds and nonstationary sources of uncertainty (e.g., demand arrivals, deadlines, and market prices). Thus, we also ask:

Can we design data-driven algorithms for OSDM *that directly learn from data (e.g., historical instances) to significantly improve performance while maintaining worst-case guarantees?*

1.1 Contributions

We obtain positive answers to both of the above questions, making the following contributions: In § 2, we formally introduce the OSDM problem, which captures several realistic features of demand management problems that arise in practice. To the best of our knowledge, OSDM is the first online optimization problem that simultaneously captures a mix of flexible and inflexible demands, inventory and/or energy storage dynamics, separate costs for procurement and delivery, and penalties that encourage "smooth" behavior.

In § 4, we provide the first competitive algorithm for OSDM, which we call PAAD (Partitioned Accounting & Aggregated Decisions, see Algorithm 2). PAAD introduces a driver abstraction that

splits OSDM into subproblems that can be solved independently and aggregated to form a globally competitive solution (Theorem 4.3). PAAD reflects, to the best of our knowledge, the most complex extension of the online pseudo-cost minimization algorithm design paradigm [82]; we employ multiple kinds and instances of *threshold functions* to decompose and accommodate the significant heterogeneity of the OSDM setting. PAAD achieves a worst-case competitive ratio that is sublinear in $p_{\max}/p_{\min}$ (the ratio of the procurement price bounds) and linear in other parameters of the problem setting such as the switching and delivery cost magnitudes—in § 5, we give lower bounds that imply PAAD achieves the best-possible competitive ratio for OSDM under our assumptions.

To improve average-case performance and reduce the pessimism of our approach, in § 6 we present a learning framework for OSDM called PALD (Partitioned Accounting & Learned Decisions). Compared to existing work on learning-augmented algorithms, the OSDM setting brings unique challenges that make previously-considered techniques for integrating predictions impractical or ineffective. To address this challenge, PALD proposes a novel end-to-end differentiable learning framework that directly predicts the best algorithm parameters (e.g., threshold functions) that minimize PALD's cost on historical instances, while ensuring that the learned algorithm satisfies a *robustness certificate* (see Theorem 6.3) that guarantees a bounded worst-case competitive ratio under adversarial instances, serving as a safeguard against nonstationarity. This framework will likely be of independent interest as a more flexible and expressive methodology to learn high-performing and robust algorithms across problem domains.

Finally, in § 7, we present a case study of OSDM in a simulation of a grid-integrated data center with co-located battery storage, using real electricity price data and production Alibaba traces [3]. We show that PAAD solves the OSDM problem effectively in practice, and that PALD's end-to-end learning capability significantly improves performance compared to PAAD using data-driven insights.

1.2 Related Work

Our study of OSDM is informed by and builds upon several lines of related work in online algorithms, learning-augmented algorithms, and relevant application areas. We briefly summarize these below.

Applications of Demand Management and Related Ideas. OSDM-like problems arise across multiple domains—below, we review some heuristic and control-theoretic approaches that have been proposed to solve these. Given forecasts (e.g., prices), a common approach is to solve an offline "day-ahead" optimization problem, as applied in battery arbitrage [26, 63, 94], microgrids [79, 84], HVAC [90, 92], and sustainable computing [49, 74]. In online settings, a frequently-applied technique is model predictive control (MPC), which uses forecasts to solve a finite-horizon optimization problem at each time step. MPC has been applied to battery management [10, 14, 66, 77], building systems [1, 11, 57, 78], microgrids [54, 65], and data centers [45, 59, 86]. Select works consider approaches that do not require predictions such as Lyapunov-based control [30, 80, 83, 85, 91], ADMM for distributed storage [75], feedback-based demand response [55], and stochastic optimization [8, 67]. These approaches are generally developed with a single application in mind, limiting their generalizability, and do not typically come with theoretical guarantees on performance, particularly in cases where predictions or stochastic models are used.

Related Online Algorithms Problems. In the literature there are a few online problems that share structure with OSDM. Of these, perhaps the closest related problem is online linear inventory management (OLIM) [89], which models the problem faced by an operator (e.g., a data center) with an inventory (e.g., battery) that must meet a demand at each time step, while minimizing their procurement cost subject to time-varying prices. Compared to OLIM, OSDM additionally models a delivery cost to serve demand (i.e., separate from procurement), smoothing penalties, and a mixture of base (inelastic) and flexible demand. A recent model called online conversion with switching costs

(OCS) [46] has modeled the problem faced by an operator procuring energy (e.g., an EV charger) that must meet a single flexible demand before a deadline (e.g., charging an EV by the end of the day), paying a switching cost when decisions change. OSDM generalizes this by including inflexible (i.e., base) demand constraints, delivery costs, and multiple flexible units with heterogeneous deadlines. Other formulations address energy storage without time-varying procurement prices [39, 62] or microgrid scheduling with variable prices but no storage [32, 52, 95].

Each component dynamic of OSDM has been studied in isolation: procurement and flexible demand via online knapsack and search [24, 28, 97], per-step demand via online matching [35, 58], and smoothness penalities via metrical task systems [12, 27] However, the combination of these in OSDM introduces new technical challenges in the design of robust and efficient algorithms that preclude the direct application of existing results (see, e.g., § 3). Online resource allocation problems [7, 22, 36, 38, 40, 87, 96] are also related, though they usually assume a fixed (non-replenishable) resource. Some variants consider inventories [16, 18, 33, 34, 44, 88], but typically treat either demand or inventory evolution as exogenous, whereas in OSDM both are decisions.

2 Problem Formulation

In this section, we introduce the online smoothed demand management (OSDM) problem, provide motivating examples as context for our modeling, and state assumptions we place on the problem.

2.1 Online Smoothed Demand Management (OSDM)

Problem Statement. Consider an operator who makes decisions over a discrete time horizon $t \in [T]$. At each time step t, several inputs are revealed online. The first of these is a *market price* for energy, denoted by $p_t \geq 0$. Furthermore, two types of demand arrive: a base (inflexible) demand $b_t \geq 0$ that must be satisfied immediately, and a flexible demand $f_t \geq 0$ that can be deferred until a specified deadline $\Delta_t > t$. Further, the operator can control the state of a local energy storage unit (e.g., a battery) with a maximum capacity of $S \geq 0$. We denote the state of charge at time step t as s_t, where the initial storage state is empty (i.e., $s_0 = 0$).[2]

At each time step t, the operator must make two decisions without knowledge of the future. First, they must decide how much energy to purchase, denoted by $x_t \geq 0$. Second, they must decide how much energy to deliver to satisfy demand, denoted by $z_t \geq 0$. The delivery decision can use a combination of newly purchased energy and energy drawn from storage. The operator's goal is to satisfy the above constraints while minimizing a cost consisting of four components: the cost of purchasing energy, penalties for an unsmooth purchasing rate (e.g., a switching or tracking cost), a cost for delivering energy (which may depend on the storage state and the market price), and a penalty for an unsmooth delivery rate (we consider a switching cost). The offline (i.e., full time horizon) version of OSDM is formalized as follows:

$$\text{[OSDM]} \quad \min_{\{x_t, z_t\}_{t\in[T]}} \sum_{t=1}^{T} \Big[\underbrace{p_t x_t}_{\text{purchase cost}} + \underbrace{\mathcal{D}(z_t, s_{t-1}, p_t)}_{\text{delivery cost}} \Big] + \sum_{t=1}^{T+1} \Big[\underbrace{\mathcal{S}(x_t, x_{t-1}) + \delta |z_t - z_{t-1}|}_{\text{smoothing penalty}} \Big], \quad (1)$$

$$\text{s.t.} \quad s_t = s_{t-1} + x_t - z_t, \text{ with } 0 \leq s_t \leq S,$$

$$b_t \leq z_t \leq b_t + \sum_{\tau=1}^{t} f_\tau - \sum_{\tau:\Delta_\tau < t} f_\tau,$$

$$\sum_{t=\tau_1}^{\tau_2} z_t - b_t \geq \sum_{t: t\geq\tau_1 \text{ and } \Delta_t \leq \tau_2} f_t \qquad \forall (\tau_1, \tau_2) : \tau_1 \leq \tau_2 \text{ and } \tau_1, \tau_2 \in [T]. \quad (2)$$

In the objective (1), $\delta \geq 0$ is a known coefficient for the delivery switching penalty, $\mathcal{S}(\cdot)$ is a function that implements the desired smoothness penalty on purchasing decisions, and $\mathcal{D}(\cdot)$ is a

[2]The assumption that the initial storage state is $s_0 = 0$ is without loss of generality—our results hold when $s_0 > 0$.

function that captures the delivery cost. For both of the latter terms, we consider several models inspired by real-world applications—see § 2.3 and § 2.4 for a discussion of each, respectively. We assume decisions outside of the time horizon are zero: $x_0 = z_0 = 0$ and $x_{T+1} = z_{T+1} = 0$.

As outlined above, the problem is subject to several constraints for all $t \in [T]$. In order, they are as follows: First, the storage state is governed by the inventory dynamics $s_t = s_{t-1} + x_t - z_t$, and the state must remain within physical limits, i.e., $0 \le s_t \le S$. OSDM models a "discharged" storage state as $s_t = 0$, but this can capture the more realistic case where $s_t = 0$ corresponds to a minimum charge level for the storage. Next, the amount of energy purchased, i.e., x_t, must be sufficient to cover the desired delivery decision, potentially drawing from storage, but cannot overfill the storage. These constraints are captured by $z_t - s_{t-1} \le x_t \le z_t + (S - s_{t-1})$. The delivery decision at time t (z_t) must *cover* the base demand and is allowed to satisfy any flexible demand that has arrived but is not past due—this is captured by $b_t \le z_t \le b_t + \sum_{\tau=1}^{t} f_\tau - \sum_{\tau:\Delta_\tau < t} f_\tau$. Finally, the cumulative delivery for any subinterval of $[T]$ must satisfy base demand and any flexible demands whose lifetime lies within this interval: this is captured by $\sum_{t=\tau_1}^{\tau_2} z_t - b_t \ge \sum_{t: t \ge \tau_1 \text{ and } \Delta_t \le \tau_2} f_t \ \forall (\tau_1, \tau_2) : \tau_1 \le \tau_2$ and $\tau_1, \tau_2 \in [T]$. We summarize the key pieces of OSDM in Fig. 1. We remark that if flexible demands are not considered, the two decision variables in the OSDM formulation can be simplified to a single decision variable (i.e., $\{x_t - z_t\}_{t \in [T]}$)—this simplified version of OSDM is of independent interest, and we refer to it as "OSDM with only base demand" throughout the paper (see e.g., Corollary 5.2 for tighter results in this regime).

While (1) captures the offline version of the problem, our goal is to design an online algorithm that chooses $\{x_t, z_t\}$ at each time step t *without knowledge of the future online inputs* of $\{p_\tau, b_\tau, f_\tau, \Delta_\tau\}_{\tau > t}$. To do so, we follow the literature on online algorithms and competitive analysis [12, 56] and evaluate performance via the *competitive ratio* [12, 56], defined as follows:

Definition 2.1 (Competitive ratio). Let $\texttt{OPT}(\mathcal{I})$ *denote the optimal offline cost for an arbitrary* OSDM *input instance* $\mathcal{I} \in \Omega$ *(where* Ω *is the set of all feasible inputs), and let* $\texttt{ALG}(\mathcal{I})$ *denote the cost of an online algorithm* ALG *over the same instance.* ALG *is said to be* α*-competitive if for all* $\mathcal{I}$*,* $\texttt{ALG}(\mathcal{I}) \le \alpha \texttt{OPT}(\mathcal{I}) + C$ *holds, where* $\alpha \ge 1$ *is the competitive ratio, and* $C \ge 0$ *is an additive constant.*

Note that a *smaller* competitive ratio implies that the online algorithm is guaranteed to be *closer* to the offline optimal solution. The competitive ratio is a *worst-case* performance metric for online algorithms; as such, bounds on this metric ensure robustness against nonstationarity in the underlying environment, which is crucial for the applications motivating OSDM. For instance, electricity prices are known to be volatile and non-stationary [73], and demand profiles may be non-stationary in certain applications. However, algorithms optimized for competitive guarantees are often overly conservative in practice—to address this limitation in OSDM, we build on the learning-augmented algorithms literature [42, 53, 61] to propose data-driven algorithms that can leverage historical data to significantly improve average-case performance while still maintaining worst-case competitive guarantees against non-stationary inputs. We discuss this further in § 6.

2.2 Motivating Applications

We present several *demand management* applications that can be modeled by OSDM, motivated by emerging *grid integration* problems. Owing to its generality, OSDM applies broadly across energy and operations domains. We focus here on the high-level mapping from problems to their OSDM instantiations, deferring a longer discussion to Appendix B.

Grid-integrated Data Center with Storage. Consider a grid-connected data center with local energy storage. At each time step t, the operator decides how much electricity to purchase (x_t) from the grid at price p_t, and how much energy to use to meet demand (z_t); the difference is stored in or drawn from the battery with state s_t. Base demands b_t represent e.g., interactive workloads

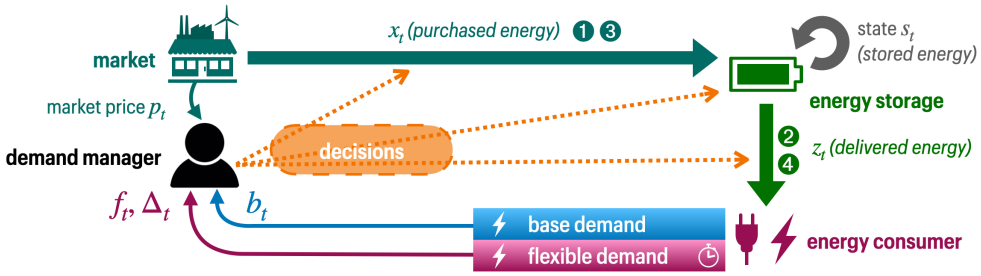


Fig. 1. Diagram of the OSDM problem at a single (discrete) time step t. The demand manager specifies a purchasing decision x_t, a delivery decision z_t, and (implicitly) a storage state s_t. As a function of these decisions, they pay ❶ a purchasing cost $p_t x_t$, ❸ a smoothness penalty $\mathcal{S}(x_t, x_{t-1})$ that discourages fluctuations in the purchasing rate, ❷ a delivery cost of serving demand $\mathcal{D}(z_t, s_{t-1}, p_t)$, and ❹ a switching penalty $\delta|z_t - z_{t-1}|$ that discourages fluctuations in the delivery rate.

that must be satisfied immediately, while flexible demands f_t capture delay-tolerant batch jobs with deadlines $\Delta_t > t$. The operator minimizes their cost of electricity subject to a smoothing penalty ($\mathcal{S}(\cdot)$) that penalizes them for large fluctuations, as sudden shifts in load cause instability and even blackouts on the power grid [43]. Additional penalties model the cost of discharging the battery ($\mathcal{D}(\cdot)$) to account for degradation, promote keeping the battery sufficiently charged to accommodate the risk of an outage, and smooth the rate of internal consumption ($\delta|z_t - z_{t-1}|$) to reduce wear-and-tear on components. Unlike prior demand-management formulations [89], OSDM jointly captures both flexible and inflexible workloads and the need for smooth grid- and demand-side decisions. This application is the focus of our case study in § 7.

Thermal Energy Demand Management. An operator manages a thermal energy system (for example, district heating) with a local storage tank [68]. At each time step t, the operator decides how much thermal energy to purchase (x_t) at price p_t and how much to deliver to meet demand (z_t); the difference is stored in or drawn from the tank with state s_t. Base demands b_t represent immediate thermal needs, while flexible demands f_t can be deferred until a deadline $\Delta_t > t$. The operator minimizes the total cost to procure energy $p_t x_t$ and deliver energy (e.g., using heat exchangers) $\mathcal{D}(\cdot)$, all subject to a smoothing penalty that models power grid stability needs (e.g., if electricity is used to generate thermal energy), since demand spikes can cause grid failure [64]. An optional penalty on the delivery side $\delta|z_t - z_{t-1}|$ models wear-and-tear on thermal components.

Just-in-Time Manufacturing with Material Inventory. A factory converts raw materials into finished products using on-site inventory. At each step t, it decides how much material to purchase (x_t) at price p_t and how much product to produce/deliver (z_t), maintaining warehouse state s_t. Base orders b_t must be filled immediately, while flexible orders f_t can be deferred until deadline $\Delta_t > t$. The objective is to minimize a purchase cost and a fixed per-unit processing cost $p_t x_t + \mathcal{D}(\cdot)$, while smoothing the purchasing rate ($\mathcal{S}(\cdot)$) and production rate ($\delta|z_t - z_{t-1}|$).

Flow Battery Storage Management. A large electricity consumer operates a grid-connected flow battery to shift energy usage [13]. At each time t, the operator purchases electricity (x_t) at price p_t and delivers electricity to meet demand (z_t), with battery charge state s_t. The goal is to minimize the total cost $p_t x_t$ subject to a smoothing penalty informed by grid considerations as above, and accounting for system losses and auxiliary energy needs through $\mathcal{D}(\cdot)$.

2.3 Modeling Smoothness Penalty

In OSDM, the smoothness penalty $\mathcal{S}(\cdot)$ in Equation 1 captures an objective to make "smooth" purchasing decisions over time. This is often an important goal (see, e.g., the applications in § 2.2); for instance, industrial consumers like data centers can have electricity consumption patterns with large fluctuations that destabilize the grid and reduce power quality [69]. Motivated by these considerations, we consider two natural models for the smoothness penalty $\mathcal{S}(\cdot)$. First, we consider a *switching cost* model (with corresponding problem referred to as OSDM-S, hereinafter), where the penalty charges the operator proportionally to the *change* in purchasing decisions:

Definition 2.2 (Switching cost for purchasing decisions). In (1)*, let* $\mathcal{S}(x_t, x_{t-1}) \coloneqq \gamma|x_t - x_{t-1}|$*, where* $\gamma \geq 0$ *is a known coefficient.*

Second, we consider a *tracking cost* model (OSDM-T, hereinafter), where the goal is not just to smooth the purchasing rate, but to follow an externally-provided target signal $\{a_t\}_{t\in[T]}$. This is motivated by recent interest in programs where grid operators provide *load shaping* signals to large electricity consumers to encourage consumption patterns that aid grid stability [5, 37]. This target signal may be known in advance (e.g., day-ahead) or revealed online (e.g., real-time), and the tracking cost penalizes the operator based on the deviation from this target signal.

Definition 2.3 (Tracking cost for purchasing decisions). In (1)*, let* $\mathcal{S}(x_t, x_{t-1}) \coloneqq \eta|x_t - a_t|$*, where* $\eta \geq 0$ *is a tracking cost coefficient known a priori, and* $a_t \geq 0$ *is a target signal revealed at time* t*.*

We remark that the OSDM objective in Equation 1 models the smoothness penalty for the delivery cost (i.e., decisions $\{z_t\}_{t\in[T]}$) as a fixed switching cost (with coefficient δ). In principle, this penalty could also be modeled as a tracking cost, and we believe that analogous results are attainable in such a model. We focus on the case where specifically the purchasing decisions are subject to a tracking cost for brevity—in a motivating application such as electricity purchasing, the grid operator provides suitable signals (e.g., reserve capacity signals [19]) to realize this model (on the purchasing side) in practice.

2.4 Modeling Delivery Cost

The delivery cost $\mathcal{D}(\cdot)$ in Equation 1 represents the additional expense incurred when delivering energy to the consumer. While often negligible ($\mathcal{D}(\cdot) = 0$), it can vary significantly across applications. In a thermal energy system, for instance, efficiency improves when the reservoir (i.e., storage) is fuller or when the temperature differential is small, implying a delivery cost that *decreases* with the storage state and depends on energy price due to pumping or heat-exchange losses [76]. In a just-in-time manufacturing context, $\mathcal{D}(\cdot)$ may reflect a fixed processing cost independent of storage or price. On the other hand, in a flow battery storage system, using stored energy requires some additional input energy to drive pumps and circulate liquid [13], yielding a cost that *increases* with discharge volume and depends on the contemporaneous energy price.

To capture this diversity of cases across multiple applications, in what follows, we define a class of monotone and price-dependent delivery costs. We believe this form of delivery cost to be a particularly challenging case that, if analyzed, offers corollary insights for other delivery cost models. It is defined as follows:

Definition 2.4 (Monotone affine price-dependent delivery cost). Let $c \geq 0$ *and* $\varepsilon \geq 0$ *be delivery cost coefficients known a priori. We say that the delivery cost* $\mathcal{D}(\cdot)$ *in* (1) *is a* ***monotone affine price-dependent delivery cost*** *if it satisfies one of the following:*

(a) (Increasing case) Let $\mathcal{D}(z_t, s_{t-1}, p_t) \coloneqq (\frac{cs_{t-1}}{S} + \varepsilon)p_t z_t$ *— this delivery cost increases linearly from* $\varepsilon p_t z_t$ *to* $(c + \varepsilon)p_t z_t$ *as the storage state increases from* $s_{t-1} = 0$ *to* $s_{t-1} = S$*.*

(b) (Decreasing case) Let $\mathcal{D}(z_t, s_{t-1}, p_t) \coloneqq (c - \frac{cs_{t-1}}{S} + \varepsilon)p_t z_t$ *— this delivery cost decreases linearly from* $(c+\varepsilon)p_t z_t$ *to* $\varepsilon p_t z_t$ *as the storage state increases from* $s_{t-1} = 0$ *to* $s_{t-1} = S$.

Several alternative delivery cost models are "covered by" Def. 2.4—for instance, setting $c = 0$ recovers a price-dependent delivery cost that is *constant* as a function of the storage state. Similarly, a reasonable alternative model might be a delivery cost that *is* state-dependent but doesn't depend on the price p_t—such a model is captured as a simpler sub-case of Def. 2.4, and our algorithm (see § 4) would directly capture any such simpler cases. In the rest of the paper, we primarily give results for the OSDM-S and OSDM-T problems using the *decreasing* monotone delivery cost case defined in Def. 2.4(b), because it is the most challenging case to analyze. Theorem 5.3 shows that given the same values of c and ε, the best achievable competitive ratios for the increasing case in Def. 2.4(a) imply it is a strictly easier problem than the decreasing case. Thus, our results primarily focus on this "hard case," holding for the general monotone definition in Def. 2.4 and any simpler subcases of it. We note that setting the delivery cost according to Def. 2.4(b) with $c > 0$ results in a non-convex (bilinear) objective in (1) due to the dependence on s_{t-1}.[3]

2.5 Assumptions and Notation

In this section, we summarize our main assumptions and additional notation for OSDM. First, we assume that the market prices are bounded: $p_t \in [p_{\min}, p_{\max}]\ \forall t \in [T]$, where $0 < p_{\min} \leq p_{\max}$. $p_{\min}$ and $p_{\max}$ are known to the operator a priori. This is a standard assumption in the literature on online search and related problems such as online knapsack [24, 44, 97]. It is known that without such an assumption, no online algorithm can achieve a bounded competitive ratio [24, 51].

We also assume that it is always feasible to satisfy demand as it becomes due, i.e., if necessary, it is always possible to set $x_t = b_t + \sum_{\tau:\Delta_\tau = t} f_\tau$ and $z_t = b_t + \sum_{\tau:\Delta_\tau = t} f_\tau$ for any $t \in [T]$. This is a natural assumption that ensures the problem is well-posed—without such an assumption, an online algorithm can be backed into a corner where feasibility becomes impossible. While we do not consider *rate constraints* in our setting of OSDM (i.e., constraints on the amount of energy that can be purchased or delivered at any time), this gives an implicit condition relevant to such constraints—see discussion of future work in § 8 for more on this point.

We assume that the switching cost coefficients are not "too large" relative to the prices. Formally, γ and δ satisfy $\gamma + \delta \leq \frac{p_{\max} - p_{\min}}{2}$. Note that if $\gamma + \delta > \frac{p_{\max} - p_{\min}}{2}$, then competitive decision-making is simple: an algorithm should choose decisions that minimize the total switching cost (i.e., by spreading decisions evenly over time). Thus, we focus on the "interesting regime" where an online algorithm must balance the purchasing and switching costs.

Using the notation of Def. 2.4, we assume the delivery cost coefficients $c \geq 0$ and $\varepsilon \geq 0$ satisfy $0 \leq c + \varepsilon \leq 1$. This ensures that the delivery cost is not "too large" relative to the purchasing cost, and is realistic in the target applications of OSDM, where the delivery cost (e.g., the added cost to serve demand using stored energy) is a fraction of the original cost to procure said energy.

In OSDM-T, we make two additional assumptions related to the target signal and the tracking cost. First, we assume that the cumulative target is at most the demand, i.e., $\sum_{t=1}^{T} a_t \leq \sum_{t=1}^{T} (b_t + f_t)$. This ensures that $\sum_{t \in [T]} a_t$ is reasonable relative to the total demand over the horizon. Without this assumption, there are unrealistic OSDM instances with, e.g., a small amount of demand and a consistently large target signal that force any algorithm to pay a large penalty for failing to have enough demand to meet the target. Finally, we assume that the tracking cost coefficient η satisfies $\eta \leq \frac{p_{\max} - p_{\min}}{2}$—this ensures that the tracking cost coefficient is not "too large" relative to the range of market prices. If this assumption is violated (i.e., if $\eta > \frac{p_{\max} - p_{\min}}{2}$), then it is always preferable

[3]The online problem is unaffected by non-convexity because the state s_{t-1} is a fixed value at time t (not a decision variable).

(from a cumulative purchasing cost perspective) to set $x_t = a_t$ wherever possible. Thus, we focus on the "interesting regime" where an online algorithm balances the tracking cost and the purchasing cost, allowing for slight deviations from the target signal when they are advantageous.

Throughout the paper, we use the shorthand notation $\Phi(w, z)$ and $\Psi(w, z)$ to denote the *definite integrals* of functions $\int_w^z \phi(u)\, du$ and $\int_w^z \psi(u)\, du$, respectively, for any $0 \le w \le z$. We also use the shorthand $P \coloneqq (1 + c + \varepsilon)p_{\max}$, $\kappa \coloneqq \gamma + \delta$, and $\omega \coloneqq \frac{(1+c+\varepsilon)}{(1+\varepsilon)}$.

3 Warmup and Preliminary Results

In this section, we start by discussing similarities between OSDM and related problems in the literature. We review one such problem and an existing algorithm for it, showing that an extension of this algorithm to OSDM fails to achieve a good competitive ratio, even in a simplified setting—this warmup result illustrates the complexity of the problem and motivates observations that we use in the design of our main algorithm, PAAD, described in § 4.

Relation to Online Conversion with Switching Costs. A much-simplified version of the OSDM problem defined in § 2 has been studied under the name of *online conversion with switching costs* (OCS) [46]. This problem is a special case of OSDM-S where there is a single unit of flexible demand (whose size is known a priori) that must be satisfied by the end of the time horizon (i.e., $f_0 = 1$ and $\Delta_0 = T$). OCS also does not consider base demand (i.e., $b_t = 0$ for all t), there is no storage (i.e., $S = 0$), and there is no delivery cost (i.e., $c = \varepsilon = \delta = 0$). The goal is to minimize the total cost of purchasing an asset subject to time-varying prices, while also minimizing a switching cost incurred from changing the purchase amount over time. For the OCS problem, an algorithmic framework called "ramp-on, ramp-off" (RORO) achieves the best possible competitive ratio [46]. The key idea underlying RORO is to maintain a *single* threshold function and completion state (i.e., "how much demand has been satisfied") that guides the algorithm's decision-making; this threshold is designed to capture the marginal trade-off between purchasing at the current price and waiting for better prices that may not arrive (see Appendix C for more details on the algorithm and setting).

A key challenge preventing the direct application of RORO to the OSDM setting is uncertainty in how much demand will arrive over time, since RORO is designed to handle an a priori fixed total demand. A natural approach to deal with such uncertainty is a *doubling strategy* that progressively increases the total demand that RORO is specified to satisfy. We describe this extension below.

Doubling Strategy for RORO. To apply the RORO framework in the setting of OSDM, our key idea is to maintain a guess $\hat{d}$ of the total flexible demand that will arrive over time, and create instances of RORO as if there is a single unit of flexible demand of size $\hat{d}$ that must be satisfied by the end of the total time horizon T (known a priori). Initially, we set $\hat{d} = 2^0 = 1$ and $j = 0$. Whenever a new flexible demand f_t arrives, we first attempt to "assign it" to the most-recently initialized instance of RORO by checking whether adding f_t to its total assigned demand would violate the guessed bound $\hat{d}$. If not, then we assign f_t to this current RORO instance and allow it to continue; otherwise, we double the guess $\hat{d} \leftarrow 2^{j+1}$, increment j, and create a new instance of RORO that is responsible for satisfying all flexible demands arriving from time t onward (including f_t). Prior instances of RORO operate until all of the flexible demands assigned to them are satisfied (or until their deadlines have passed), and the global purchasing decision is the sum of the decisions of all active instances of RORO. We formalize this doubling extension of RORO in the Appendix (see Algorithm 4). To analyze this extension of RORO, we restrict to a subset of OSDM-S instances with only flexible demand (i.e., $b_t = 0$ for all t), no delivery cost (i.e., $c = \varepsilon = \delta = 0$), no storage ($S = 0$), and we assume that the maximum demand unit is upper-bounded by some σ (i.e., $\max_{t\in[T]} f_t \le \sigma$), where $\sigma < 1$. Then, we can characterize the competitive ratio of the doubling extension of RORO as follows:

Theorem 3.1. *For the subset of* OSDM-S *instances described above, the "doubling extension" of* RORO *is ζ-competitive, where α_{RORO} is the optimal competitive ratio for* OCS *(see Equation 9) and ζ is at least:*

$$\zeta \geq \min\left\{\frac{\alpha_{\text{RORO}}}{1-\sigma} + \frac{\alpha_{\text{RORO}}}{1-\sigma}\cdot\frac{\left(p_{\max}-2\gamma-\frac{p_{\max}}{\alpha_{\text{RORO}}}\right)\exp\left(\frac{\sigma}{\alpha_{\text{RORO}}}\right)-\frac{\sigma p_{\max}(1-\sigma)}{\alpha_{\text{RORO}}}}{p_{\min}}, \frac{p_{\max}+2\gamma}{p_{\min}}\right\}.$$

We give the full proof of Theorem 3.1 in Appendix D. This result shows that the competitive ratio of the doubling RORO algorithm can be *arbitrarily bad*—in particular, the first entry in the min expression above grows to infinity as σ approaches one, implying that the competitive ratio of the doubling RORO algorithm is not better than the trivial $O\left(\frac{p_{\max}}{p_{\min}}\right)$ bound achievable by any feasible OSDM algorithm. Intuitively, a simple extension of RORO cannot account for the demand that arrives in a coarse-grained manner: it fails to resolve the "gap" between the demand the algorithm anticipates and the actual demand that arrives, and this gap leads to its suboptimality. This limitation motivates the modular and adaptive design of our main algorithm PAAD, which we present next.

4 A Unified Competitive Algorithm

In this section, we present a unified algorithm called **P**artitioned **A**ccounting & **A**ggregated **D**ecisions (PAAD) to solve OSDM. Motivated by the result in § 3 that demonstrates the necessity of *accounting* for OSDM, PAAD introduces abstractions called *drivers* that perform a double duty of accounting for distinct demand units while making decisions that can be aggregated into global purchasing and delivery actions. We formally introduce the PAAD algorithm in § 4.1 and analyze its competitive ratio for OSDM-S and OSDM-T in § 4.2.

4.1 The PAAD Algorithm

The core idea of PAAD is to handle each newly-arrived unit of demand with an abstract *driver*, which produces decisions which are aggregated into a globally feasible purchasing and delivery action at each time. We start by detailing the driver types used in PAAD before describing the algorithm itself.

Drivers. PAAD uses two types of drivers to manage the purchasing and delivery decisions: (i) *base demand* drivers (denoted by set $\mathcal{B}$) that manage units of base demand, and (ii) *flexible demand* drivers (denoted by set $\mathcal{F}$) that manage units of flexible demand. Each driver is assigned a *size* based on the demand it is responsible for, and makes local decisions based on its own state.

At initialization ($t = 0$), PAAD creates an initial *manager* driver that charges the storage when the market price is sufficiently low—this driver is instantiated as a base demand driver with index 0 ($\mathcal{B} \leftarrow 0$) and size $d^{(0)} \coloneqq S$ (the storage capacity). At each time step $t \in [T]$, upon the arrival of p_t, b_t, and f_t (with deadline Δ_t), drivers are created as follows. If $b_t > 0$, then a new base demand driver with even index $2t$ is created and added to the set of base drivers $\mathcal{B}$, with size $d^{(2t)} \coloneqq b_t$. If $f_t > 0$, then a new flexible demand driver with odd index $2t+1$ is created and added to the set of flexible drivers $\mathcal{F}$, with size $d^{(2t+1)} \coloneqq f_t$ and deadline slack Δ_t.

Each base demand driver $i \in \mathcal{B}$ makes local decisions governed by a *purchasing threshold function* $\phi_b^{(i)}(w) : w \in [0, d^{(i)}]$, and updates a state variable $w_b^{(i)}$ that indicates how much has been purchased so far. $\phi_b^{(i)}$ is non-increasing in w, and captures the marginal trade-off between making progress towards satisfying the driver's total demand and waiting for (potentially) better prices.

Definition 4.1 (Base demand threshold function). Given a base driver with size denoted by d, the threshold function ϕ_b for OSDM-S *is defined as follows, where α is the competitive ratio (see Thm. 4.3):*

$$\phi_b(w) = p_{\max}+2\gamma+p_{\min}c+\left(\frac{p_{\max}(1+c+\varepsilon)+2\kappa}{\alpha}-\left(p_{\max}(1+\varepsilon)+p_{\min}c+\frac{2\kappa}{T}\right)\right)\exp\left(\frac{w}{\alpha d}\right) : w \in [0, d],$$

Each flexible demand driver $l \in \mathcal{F}$ maintains a *purchasing threshold function* $\phi_f^{(l)}(w) : w \in [0, d^{(i)}]$ and a *delivery threshold function* $\psi^{(l)}(v) : v \in [0, d^{(i)}]$ (defined below), with state variables $w_f^{(l)}$ and $v_f^{(l)}$ denoting the amount of their demand already purchased and delivered, respectively.

Definition 4.2 (Flexible demand threshold functions). Given a flexible driver with size denoted by d, the purchasing threshold function ϕ_f for OSDM-S *is defined as follows, where $\omega = \frac{(1+c+\varepsilon)}{(1+\varepsilon)}$ and $\alpha' = \alpha/\omega$:*

$$\phi_f(w) = p_{\max} + p_{\min}c + 2\gamma + \left(\frac{p_{\max} + 2\gamma}{\alpha'} - \left(p_{\max} + p_{\min}c + \frac{2\gamma}{T}\omega\right)\right)\exp\left(\frac{w}{\alpha' d}\right) : w \in [0, d],$$

and the delivery threshold function ψ for OSDM-S *is defined as:*

$$\psi(v) = p_{\max}(c+\varepsilon) + 2\delta + \left(\frac{p_{\max}(c+\varepsilon) + 2\delta}{\alpha'} - \left(p_{\max}(c+\varepsilon) + \frac{2\delta}{T}\omega\right)\right)\exp\left(\frac{v}{\alpha' d}\right) : v \in [0, d].$$

Algorithm. We now describe the PAAD algorithm itself. The main pseudocode is provided in Algorithm 2, and the pseudo-cost minimization subroutine for individual drivers is detailed in Algorithm 1. The key idea guiding PAAD's behavior is to have each driver make local decisions based on its own state and threshold functions, while also carefully accounting for switching costs across multiple drivers, allowing the algorithm to amortize global switching costs. All local decisions are *aggregated* into a single global decision: $x_t \leftarrow \sum_{i \in \mathcal{B} \cup \mathcal{F}} x_t^{(i)}, z_t \leftarrow b_t + \sum_{i \in \mathcal{F}} z_t^{(i)}$. PAAD ensures feasibility by checking for binding constraints (i.e., flexible demands nearing their deadlines, Line 4 in Algorithm 1) and ensuring that sufficient energy is purchased to meet the delivery decision (Line 21). PAAD amortizes global switching costs by communicating *pseudo-decisions* to each driver, which are adjusted based on the global purchasing decision from the previous time step. This allows each driver to recognize when increasing its purchase amount can help to amortize the total switching cost across multiple drivers, reducing the pessimism inherent in a purely local approach.

For each driver $i \in \mathcal{B} \cup \mathcal{F}$, this is formalized as the definition of $\hat{x}_{t-1}^{(i)}$ on Line 2 in Algorithm 1, where $q(x_{t-1})$ is the difference between the sum of all driver decisions and the global decision at time $t-1$. Intuitively, $q(x_{t-1})$ captures any "extra" purchasing from the previous time step that was not attributable to any currently active driver (e.g., due to base demand or inactive drivers). Note that Line 3 (Algorithm 1) includes a $\gamma|x|$ term—this term accounts for the worst-case future cost of "switching off" whenever $x > 0$ (e.g., when the sequence ends at time T, decisions $x_{T+1} = z_{T+1} = 0$). By considering the pseudo-decision $\hat{x}_{t-1}^{(i)}$ instead of the actual previous decision $x_{t-1}^{(i)}$ when making the next decision, PAAD amortizes this "future cost" across multiple drivers, resulting in smoother global decisions. The same is applied to the delivery decisions of flexible drivers. Note that PAAD may generate as many as $\Omega(T)$ drivers on a given problem instance. The necessity of these many drivers is supported by our warmup result in § 3, which shows that a granular, per-demand accounting is likely necessary to achieve the best possible competitive ratio when future demands are unknown. However, PAAD's driver refresh (Line 23) ensures that, in practice, there will likely be far fewer than

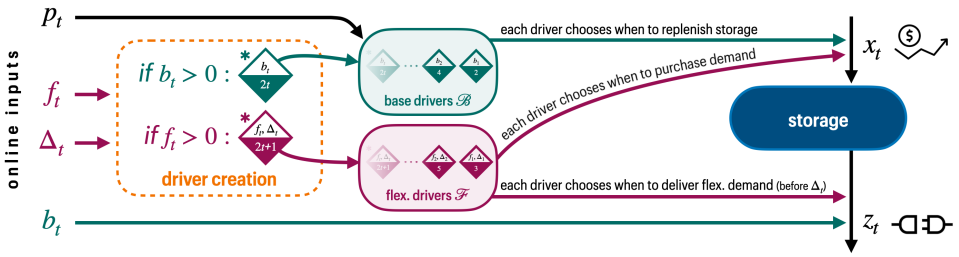


Fig. 2. Diagram of the intuition behind the PAAD algorithm. Each driver accounts for a single unit of base or flexible demand, making local decisions based on its own state and role that are aggregated into a single global decision. Purchasing decisions x_t govern how much to purchase at the current market price, and delivery decisions z_t dictate how much demand is satisfied at the current time (subject to feasibility constraints).

$\Omega(T)$ drivers active at any given time. In particular, a driver is only active for a finite period—e.g., until flexible drivers' deadlines pass, or until base drivers complete their purchasing. We discuss this point further and give runtime measurements in Appendix A.1.

Algorithm 1 Pseudo-cost Minimization Subroutine for PAAD Drivers (GetDecision)

input: driver sets $\mathcal{B}$ and $\mathcal{F}$, driver index i, excess $q(x_{t-1})$, price p_t, switching coeff. γ, threshold Φ
define *pseudo-decision* $\hat{x}_{t-1}^{(i)} \leftarrow x_{t-1}^{(i)} + \frac{q(x_{t-1})\cdot d^{(i)}}{\sum_{i\in\mathcal{B}\cup\mathcal{F}} d^{(i)}}$
solve $x_t^{(i)} \leftarrow \operatorname{argmin}_{x\in[0,d^{(i)}-w_{t-1}^{(i)}]} p_t x + \gamma|x-\hat{x}_{t-1}^{(i)}| + \gamma|x| - \Phi^{(i)}(w_{t-1}^{(i)}, w_{t-1}^{(i)}+x)$
if $i\in\mathcal{F}$ **and** $\Delta^{(i)} = t$ **then** $x_t^{(i)} \leftarrow d^{(i)} - w_{t-1}^{(i)}$ ▷ flexible demand must be satisfied *now* (at the deadline)
output: driver's decision for time step t: $x_t^{(i)}$

Algorithm 2 Partitioned **A**ccounting & **A**ggregated **D**ecisions (PAAD) Algorithm for OSDM-S

input: Storage capacity S, switching cost coefficients γ, δ, delivery cost $\mathcal{D}(\cdot)$, threshold functions Φ_b, Φ_f, Ψ
initialize: Storage state $s_0 \leftarrow 0$, decisions $x_0 \leftarrow 0$, $z_0 \leftarrow 0$, driver sets $\mathcal{B} \leftarrow \emptyset, \mathcal{F} \leftarrow \emptyset$
for $t = 1$ to T: **observe** p_t, b_t, f_t, Δ_t, **do**
  if $s_{t-1} = 0$ or $b_t > S$ **then** ▷ refresh base drivers and storage manager driver
    Create new storage manager driver with index 0 and size $d^{(0)} = S$; $\mathcal{B} \leftarrow \{0\}$
  if $0 < b_t < S$ **then**
    Create base driver with index $2t$ and size $d^{(2t)} = b_t$; $\mathcal{B} \leftarrow \mathcal{B}\cup\{2t\}$
  if $0 < f_t$ **then**
    Create flexible driver with index $2t{+}1$, size $d^{(2t+1)} = f_t$ and deadline $\Delta^{(2t+1)} = \Delta_t$; $\mathcal{F} \leftarrow \mathcal{F}\cup\{2t{+}1\}$
  compute $q(x_{t-1}) \leftarrow x_{t-1} - \sum_{i\in\mathcal{B}\cup\mathcal{F}} x_{t-1}^{(i)}$, $\quad q(z_{t-1}) \leftarrow z_{t-1} - \sum_{i\in\mathcal{F}} z_{t-1}^{(i)}$
    ▷ captures "extra" purchase/delivery due to prior demand or inactive drivers
  for each active flexible driver $i\in\mathcal{F}$ **do**
    $z_t^{(i)} \leftarrow \texttt{GetDecision}(\mathcal{B},\ \mathcal{F},\ i,\ q(z_{t-1}),\ \mathcal{D}(z_t, s_{t-1}, p_t),\ \delta,\ \Psi^{(i)})$
    update delivery state: $v_t^{(i)} = v_{t-1}^{(i)} + z_t^{(i)}$
  aggregate delivery decisions $z_t \leftarrow b_t + \sum_{i\in\mathcal{F}} z_t^{(i)}$ and **compute** buying $\texttt{cap}_t \leftarrow z_t + (S - s_{t-1})$
  for each active driver $i\in\mathcal{B}\cup\mathcal{F}$ **while** $\texttt{cap}_t > 0$ **do**
    $x_t^{(i)} \leftarrow \texttt{GetDecision}(\mathcal{B},\ \mathcal{F},\ i,\ q(x_{t-1}),\ p_t,\ \gamma,\ \Phi^{(i)})$
    $x_t^{(i)} \leftarrow \min\{x_t^{(i)}, \texttt{cap}_t\}$ ▷ note $\Phi^{(i)}$ indicates Φ_b or Φ_f depending on driver type
    update buying $\texttt{cap}_t \leftarrow \texttt{cap}_t - x_t^{(i)}$ and purchasing state $w_t^{(i)} = w_{t-1}^{(i)} + x_t^{(i)}$
  aggregate global decisions: $x_t \leftarrow \sum_{i\in\mathcal{B}\cup\mathcal{F}} x_t^{(i)}$
  $x_t \leftarrow \max\{x_t, z_t - s_{t-1}\}$ ▷ ensure sufficient energy to deliver
  update storage state: $s_t = s_{t-1} + x_t - z_t$
  refresh drivers: remove any driver i with $w_t^{(i)} = d^{(i)}$ or $\Delta^{(i)} = t$ from $\mathcal{B}$ or $\mathcal{F}$, respectively
output: Purchase decisions $\{x_t\}_{t=1}^T$, delivery decisions $\{z_t\}_{t=1}^T$

4.2 Competitive Analysis

We now analyze the PAAD algorithm and show that it achieves the best possible competitive ratio for OSDM. We start by stating our main result.

Theorem 4.3. PAAD *is α-competitive for* OSDM-S*, where α is given by:*

$$\alpha = \frac{\omega}{\left[W\left(-\frac{(P-(1+\varepsilon)p_{\min})\exp\left(\frac{-\omega 2\kappa/T - p_{\min}c - P}{P+2\kappa}\right)}{P+2\kappa}\right) + \frac{P+p_{\min}c-\omega 2\kappa/T}{P+2\kappa}\right]}, \tag{3}$$

where $W(\cdot)$ is the Lambert W function [20], $P = (1+c+\varepsilon)p_{\max}$, $\kappa = \gamma + \delta$ and $\omega = \frac{(1+c+\varepsilon)}{(1+\varepsilon)}$.

This expression for α is complex, but a leading order approximation yields asymptotics that are sublinear in the ratio $p_{\max}/p_{\min}$ (specifically, $\alpha \approx O\left(\sqrt{p_{\max}/p_{\min}}\right)$), and linear in both the switching cost coefficients γ, δ and the delivery cost coefficient c. We plot α as a function of these parameters in Appendix E, and prove that it is the optimal competitive ratio in § 5.

We remark on several regimes of interest for α where OSDM-S essentially "recovers" the optimal competitive ratio of other online problems. First, in a case with zero delivery cost (and zero delivery switching cost), letting $T \to \infty$, $\omega = 1$, $\kappa = \beta$, $c = \epsilon = 0$, and $P = U - 2\beta$,[4] α exactly reduces to the optimal competitive ratio for online conversion with switching costs, where β is the switching coefficient [46]. Furthermore, letting $T \to \infty$, $\omega \to 1$, $\kappa = 0$, $c = \epsilon = 0$, and $P \to U$, α reduces to the optimal competitive ratio for the minimization version of one-way trading [51].

Corollary 4.4. PAAD *is α_{T}-competitive for* OSDM-T*, for α_{T} given by:*

$$\alpha_{\mathsf{T}} = \frac{\omega}{\left[W\left(-\frac{(P-(1+\varepsilon)p_{\min})\exp\left(\frac{-\omega 2\delta/T - p_{\min}c - P}{P+2(\eta+\delta)}\right)}{P+2(\eta+\delta)}\right) + \frac{P+p_{\min}c-\omega 2\delta/T}{P+2(\eta+\delta)}\right]}, \tag{4}$$

This result uses a redefinition of the thresholds and pseudo-cost minimization problem (see Def. E.6).

4.3 Proof Overview

We sketch the proof of Theorem 4.3, relegating its full proof and that of Corollary 4.4 to Appendix E.

Proof Sketch of Theorem 4.3. To show this result, we prove lemmas to characterize the cost of OPT, the cost of PAAD, and the relationship between the two. First, note that PAAD's solution is feasible (i.e., it satisfies all OSDM constraints, see Lemma E.1)—this follows by PAAD's definition.

For an arbitrary OSDM instance, we then carefully lower bound the cost of OPT—compared to prior applications of the pseudo-cost minimization framework, the key technical challenge is that it is not sufficient to derive a single global lower bound on OPT's cost (e.g., as in OCS [46]), since there are many units of demand with differing constraints. The key idea of our analysis in Lemma E.3 is to partition the instance into *active* and *inactive* periods based on PAAD's base driver state. This enables us to derive a lower bound on OPT's cost that is a combination of the "local" best prices during each active period and the best price during all inactive periods, which is itself lower bounded. Next, we derive an upper bound on PAAD's total cost by leveraging the definition of the pseudo-cost minimization to "charge" each driver's decisions to their threshold functions (see Lemma E.4), while accounting for any worst-case purchasing, switching, and delivery costs that arise due to binding constraints for a particular unit of demand. To relate the cost of PAAD to that of OPT, we prove two lemmas that give a relation between each threshold function and its integral:

Lemma 4.5. *By the definition of the threshold function $\phi_b(\cdot)$, the following relation always holds:* $\Phi_b(0,w) + (1-w)(p_{\max}+2\gamma) + p_{\max}(c+\varepsilon) + 2\delta - cwp_{\min} = \alpha\left[\phi_b(w) - 2\gamma + \varepsilon p_{\max} + \frac{2\kappa}{T}\right] \ \forall w \in [0,1]$.

[4] In contrast to OSDM, the original OCS paper [46] assumed that any necessary compulsory trading at the end of the time horizon could be accomplished without incurring additional switching cost, captured by the -2β in this term.

Lemma 4.6. *By the definitions of the threshold functions $\phi_f(\cdot)$ and $\psi_f(\cdot)$, the following relation always holds:* $\Phi_f(0,w) + (1-w)(p_{\max} + 2\gamma) - cwp_{\min} + \Psi_f(0,v) + (1-v)(p_{\max}(c+\varepsilon) + 2\delta) = \alpha'\left[\phi_f(w) + \psi_f(v) - 2\kappa + \frac{2\kappa\omega}{T}\right] \forall w \in [0,1], v \in [0,w]$.

The rest follows by contradiction—we show that if Lemma E.3 and E.4 hold, then if the threshold functions adhere to Lemmas 4.5 and 4.6, $\frac{\texttt{PAAD}(\mathcal{I}) - p_{\max}\hat{s}}{\texttt{OPT}(\mathcal{I})}$ must be at most α, completing the proof. □

5 Fundamental Limits

In this section, given the results in § 4, we ask whether any algorithm can achieve a better competitive bound for OSDM than PAAD. In Theorems 5.1 and 5.4 below, we answer this question in the negative, showing that PAAD's competitive ratio is the best achievable among all deterministic online algorithms for OSDM-S and OSDM-T, respectively. Along the way, we show results to contextualize the relative "hardness" of OSDM in subcases of interest, such as the case of only base demand (see Corollary 5.2) or the case of price-dependent increasing delivery cost (recall Def. 2.4).

Lower Bound for OSDM-S. We start by considering the switching cost case of OSDM-S. Theorem 4.3 shows that PAAD is α-competitive (defined in (3)) for OSDM-S. We now prove that α is optimal by constructing a difficult set of OSDM-S instances on which no deterministic algorithm can achieve a better competitive ratio; this *lower bounds* the achievable ratio of any deterministic online algorithm for OSDM-S (see Def. 2.1). We give a sketch of the result, deferring the full proof to Appendix F.

Theorem 5.1. *There exists a set of* OSDM-S *instances for which no deterministic online algorithm* ALG *can achieve a competitive ratio better than α (for α defined in* (3)*).*

Proof Sketch of Theorem 5.1. In Def. F.1, we define a set of *x-decreasing instances* for OSDM-S (denoted by $\{\mathcal{I}_x\}_{x\in[p_{\min},p_{\max}]}$) where x is the "best" (i.e., lowest) price in the instance. These instances consist of a single unit of flexible demand at time $t=1$ with a deadline of $\Delta_1 = T$. An adversarial price sequence presents market prices that generally decrease over time, interrupted by "spikes" to the highest price $p_{\max}$ that force an online algorithm ALG to pay a high switching cost. This captures a trade-off between being eager or reluctant to accept a certain price x—if ALG purchases too much at prices $> x$, it may miss better prices later on, but if it waits to purchase at prices $< x$, it may be forced to purchase and deliver at a high price $p_{\max}$ (e.g., if the sequence ends after x).

Under these special instances, the cost of any deterministic online algorithm ALG can be described by two (arbitrary) purchasing and delivery functions $h(x), z(x) : [p_{\min}, p_{\max}] \to [0,1]$ that describe the fraction of the demand purchased and delivered at prices $\geq x$. Since OPT is easy to describe on these instances (i.e., $\texttt{OPT}(\mathcal{I}_x) = x + \frac{2\gamma}{T} + \varepsilon x + \frac{2\delta}{T}$), we show that the purchasing and delivery functions of an $\alpha^\star$-competitive ALG must satisfy a necessary condition that relates $h(x), z(x)$ to $\alpha^\star$ for all $x \in [p_{\min}, p_{\max}]$. By applying Grönwall's Inequality [60, Theorem 1, p.356], this gives the following condition that $\alpha^\star$ must satisfy at optimality: $1 = \frac{\alpha^\star}{\omega} \ln\left[\frac{P-(1+\varepsilon)p_{\min}}{P - \frac{\omega[P+2\kappa]}{\alpha^\star} + \frac{\omega 2\kappa}{T} + p_{\min}c}\right]$. Solving this transcendental for $\alpha^\star$ yields the result, completing the proof. □

In the above result, we have shown that PAAD achieves the best possible competitive ratio for OSDM-S. The proof of Theorem 5.1 relies on a class of OSDM instances that consist purely of flexible demand. In the following corollary, we use a similar class of "hard instances" to argue that the competitive ratio *improves* if we focus on the case of OSDM-S instances with *only base demand*—recall from § 2.1 that this case corresponds to a simplification of OSDM wherein the two decision variables of OSDM can be simplified to a single decision variable (i.e., $\{x_t - z_t\}_{t\in[T]}$).

Corollary 5.2. *There exists a set of* OSDM-S *instances with only base demand for which no deterministic online algorithm* ALG *can achieve a competitive ratio better than* α_{B}*, given by:*

$$\alpha_{\mathsf{B}} \geq \left[W\left(-\frac{\left((1+c)p_{\max} - p_{\min}\right) \exp\left(-\frac{P+2\delta+\frac{2\gamma}{T}}{P+2\kappa} \right)}{P+2\kappa} \right) + \frac{P+2\delta+\frac{2\gamma}{T}}{P+2\kappa} \right]^{-1}, \tag{5}$$

where $P = (1+c+\varepsilon)p_{\max}$*. Given fixed* $p_{\min}, p_{\max}, c, \varepsilon, \gamma$*, and* δ*, we have* $\alpha_{\mathsf{B}} < \alpha$ *(for* α *defined in* (3)*).*

We relegate the full proof of Corollary 5.2 to Appendix F.2. The main insight of Corollary 5.2 is that the additional complexity of flexible demands makes the general OSDM problem (i.e., allowing both flexible and base demand) strictly harder to solve in the online setting.

We remark that Theorem 5.1 and Corollary 5.2 do not rule out the possibility of randomization improving the achievable competitive ratio. While several works [24, 51, 97] have shown that randomization does not improve the competitive ratio for continuous or fractional online allocation problems such as one-way trading and online knapsack, the smoothness terms in the objective of OSDM (and related problems such as OCS) may change this property.

Delivery Cost Dynamics. In the above results and throughout the paper, we focus on the case of a price-dependent delivery cost that is *decreasing* as a function of the storage state. Amongst the class of delivery costs defined in Def. 2.4, we claimed this is the most challenging case to consider in § 2. We now prove this claim by showing that OSDM with a price-dependent *increasing* delivery cost is strictly easier. As a corollary, any simpler special cases of Def. 2.4 (e.g., a price-dependent delivery cost that is constant in the storage state) are also easier in a competitive sense.

Theorem 5.3. *Amongst the class of monotone delivery costs defined in Def. 2.4, the best achievable competitive ratios for* OSDM *with an increasing delivery cost are strictly better than those for* OSDM *with a decreasing delivery cost (i.e., the cases considered in Theorem 5.1, 5.4 and Corollary 5.2).*

We relegate the full proof of Theorem 5.3 to Appendix F.3, which relies on lower bound "hard instance" constructions similar to those used in Theorem 5.1 and Corollary 5.2. The main insight is that our general competitive results focus on the most challenging setting of a delivery cost that is price-dependent and decreasing as a function of the storage state—these results directly extend to the other "simpler" cases of delivery cost models.

Lower Bound for OSDM-T. Following the results for OSDM-S, we now consider the tracking cost case of OSDM-T. Recall that Corollary 4.4 shows that PAAD is α_{T}-competitive (defined in (4)) for OSDM-T. We now prove this is optimal by constructing a difficult set of instances and tracking target sequences on which no algorithm can do better than α_{T}-competitive, thus showing that PAAD achieves the best possible competitive ratio for OSDM-T. We defer the proof to Appendix F.4.

Theorem 5.4. *There exists a set of* OSDM-T *instances for which no deterministic online algorithm* ALG *can achieve a competitive ratio better than* α_{T} *(for* α_{T} *defined in* (4)*).*

6 Learning Data-Driven Decisions

In § 2.1, we mentioned that algorithms designed using competitive analysis are often overly pessimistic, and can sometimes perform poorly compared to simple heuristics used in practice. However, these heuristics—as well as data-driven, machine-learned algorithms—can suffer a lack of robustness to nonstationarity or distribution shift. In this section, we propose a novel paradigm that directly learns "the best OSDM algorithm" lying within a defined set of algorithms that satisfy a given (desired) worst-case performance guarantee, improving average-case performance while retaining robustness. To accomplish this, we show that the pseudo-cost design of PAAD provides a mechanism by which one can theoretically describe the general set of "ρ-robust algorithms" (for any $\rho > \alpha$, see

Def. 6.1). To realize this learning paradigm in practice, we show a *end-to-end learning methodology* (see § 6.2) that leverages recent techniques from the literature on learning-to-optimize (L2O) [15], such as differentiable optimization layers. We begin by reviewing some relevant preliminaries.

Preliminaries & Challenges. In the literature on learning-augmented online algorithms, the competitive ratio (Def. 2.1) is replaced with two metrics: *consistency* and *robustness* [42, 53, 61]. If ALG is a learning-augmented online algorithm provided with some advice ADV (e.g., a prediction about the instance), then ALG is said to be *c-consistent* if it is c-competitive when ADV is *perfectly accurate*, and ρ-*robust* if it is ρ-competitive, regardless of how bad (e.g., adversarial) ADV is.

There are two common models for advice. The first is *decision advice*, where ADV predicts the optimal decisions at each time step—this type of advice is frequently used for problems with multiple time steps and switching costs, such as metrical task systems [4] and the OCS problem discussed in § 3 [46] The second is *input prediction advice*, where ADV predicts a characteristic of the problem instance, such as the number of skiing days in a ski rental problem [42]. This type of advice has been used for related problems such as online knapsack [21] and one-way trading [81], which is itself closely related to OCS. Both models face limitations in the context of OSDM. Obtaining decision advice directly from an ML model can be difficult due to the generalization challenges posed by complex problem structure—existing methods instead predict input features (e.g., a price forecast in OCS), solve the corresponding "certainty equivalent" offline problem, and use that solution as advice [46]. In OSDM, however, joint uncertainty in prices and demand arrivals/deadlines makes this approach brittle to prediction error and impractical in practice. On the other hand, point predictions about instances (e.g., estimates of the best price) are straightforward to obtain, but are too simplistic to provide useful insights for OSDM. For instance, even given a perfect prediction of the best (i.e., lowest) price, it is not possible to determine optimal purchasing and delivery schedules subject to the unknown demand arrivals/deadlines and switching/tracking costs.

On top of these limitations to existing prediction models, there is a gap between theory and practice where metrics are concerned. In particular, while robustness is useful as a worst-case "guardrail" against, e.g., significantly out-of-distribution inputs, the *consistency* metric is less meaningful—it is only defined for perfect predictions that are unrealistic in practice, and it is still a worst-case metric (i.e., it is defined as the competitive ratio given accurate predictions). While recent work has considered "smooth degradation" of algorithm performance to small prediction errors in online search problems [9, 25], it is not clear that these methods can be extended to the substantially more complex setting of OSDM. In practice, the actual objective is for an algorithm to perform well *on average*: if an algorithm performs very well in practice but has bad theoretical consistency, that is preferable to an algorithm that has good consistency in theory but poor real-world performance. This motivates us to propose a model that circumvents the challenges above while still providing the key benefits of "both worlds"—improved performance in practice with worst-case guarantees.

6.1 The PALD Framework

To tackle the above challenges, rather than defining a new prediction model and designing an algorithm around it, we propose a new learning framework (Partitioned Accounting & Learned Decisions (PALD)) that directly "learns the best algorithm" from within a general characterized set of algorithms, where each algorithm in the set satisfies a certain desired robustness (i.e., each algorithm guarantees a certain competitive ratio).

Motivation. Under the pseudo-cost minimization framework, a key step in bounding the competitive ratio of PAAD (Thm. 4.3) uses characteristics of the threshold functions (ϕ_b for base drivers, ϕ_f and ψ_f for flexible drivers) that govern the algorithm's decisions. In particular, the definition of the pseudo-cost minimization problem allows us to relate PAAD's cost to a lower bound on OPT via

these threshold functions. Then, for a given *target robustness* $\rho > \alpha$ (where α is the competitive ratio of PAAD for OSDM-S), we can exactly define a *feasible set* of threshold functions, based on the necessary relations for the competitiveness proof, that guarantee ρ-robustness for PALD. A similar characterization of "robust and consistent thresholds" was derived by Benomar et al. [9] for one-max search to analytically investigate smoothness in algorithms with predictions. In PALD however, our goal is to directly *learn* the best thresholds within a feasible set that *only* considers robustness, empirically optimizing performance on historical instances while retaining a bounded competitive ratio. Given learned thresholds $\hat{\phi}_b, \hat{\phi}_f$ and $\hat{\psi}_f$, PALD's pseudocode is exactly the same as that of PAAD (see Algorithm 2) but with $\hat{\phi}_b, \hat{\phi}_f$ and $\hat{\psi}_f$ in place of ϕ_b, ϕ_f and ψ_f (analytical thresholds).

Robustness Certificate. In what follows, we define feasible sets of *robust threshold functions* that guarantee ρ-robustness. The PALD framework accommodates any arbitrary learning mechanism as long as one can parameterize threshold functions and ensure they lie within these feasible sets, providing a robustness certificate that we use to prove PALD's worst-case guarantees. We focus on the setting of OSDM-S in the main body, deferring definitions, theorems, and proofs for OSDM-T to Appendix G.2. We start with the definition of the feasible set for base demand drivers:

Definition 6.1 (Robust threshold set for base drivers in OSDM-S*). Given a target robustness $\rho > \alpha$ (α as defined in (3)), a learned threshold function $\hat{\phi}_b$ must lie in the following feasible set:*

$$\mathcal{R}_b(\rho) \coloneqq \Big\{\phi_b : [0,1] \to [p_{\min}, p_{\max}] \;\Big|\; \phi_b \text{ is monotone non-increasing, } \phi_b(1) \le p_{\min} + 2\gamma, \text{ and } \forall w \in [0,1]: \\ \Phi_b(0,w) + (1-w)(p_{\max}+2\gamma) + p_{\max}(c+\varepsilon) + 2\delta - cwp_{\min} \le \rho\left[\phi_b(w) - 2\gamma + \varepsilon p_{\max} + 2\kappa/T\right]\Big\}.$$

If a learned threshold function $\hat{\phi}_b$ lies in the set $\mathcal{R}_b(\rho)$, we have the necessary conditions to prove that PALD is ρ-robust (with respect to base demand drivers). The first two conditions (monotonicity and the inequality at $w = 1$) ensure that $\hat{\phi}_b$ "sweeps" the range of possible prices to provide a necessary range of lower bounds on OPT. The third condition is the key inequality that relates PALD's cost to that of the offline optimal via the pseudo-cost paradigm—on the left-hand-side, the integral and additive terms correspond to an upper bound on PALD's total cost, while the right-hand-side corresponds to a lower bound on OPT's cost scaled by the target robustness ρ. The same logic extends, with some additional complexity, to the case of flexible demand drivers:

Definition 6.2 (Robust threshold set for flexible drivers). Given a target robustness $\rho > \alpha$, learned threshold functions $\hat{\phi}_f$ and $\hat{\psi}_f$ for flexible demand drivers must lie in the following joint feasible set:

$$\mathcal{R}_f(\rho) \coloneqq \Big\{(\phi_f, \psi_f) : [0,1]^2 \to [p_{\min}, p_{\max}]^2 \;\Big|\; \phi_f, \psi_f \text{ are monotone non-increasing, } \phi_f(1) \le p_{\min} + 2\gamma, \\ \psi_f(1) \le p_{\min}(c+\varepsilon) + 2\delta, \text{ and } \forall w \in [0,1], v \in [0,w]: \Phi_f(0,w) + (1-w)(p_{\max}+2\gamma) - cwp_{\min} + \\ \Psi_f(0,v) + (1-v)(p_{\max}(c+\varepsilon)+2\delta) \le \rho\left[1/\omega\left(\phi_f(w) + \psi_f(v) - 2\kappa\right) + 2\kappa/T\right]\Big\}.$$

Under Def. 6.1 and Def. 6.2, we have the following robustness certificate for PALD when given any learned thresholds $\hat{\phi}_b, \hat{\phi}_f$ and $\hat{\psi}_f$ that lie in the respective feasible sets:

Theorem 6.3. *Given learned threshold functions that lie in the feasible sets $\hat{\phi}_b \in \mathcal{R}_b(\rho)$ and $(\hat{\phi}_f, \hat{\psi}_f) \in \mathcal{R}_f(\rho)$ for some $\rho > \alpha$ (α defined in (3)), the* PALD *algorithm is ρ-robust for* OSDM-S.

Proof Sketch of Theorem 6.3. (full proof in Appendix G.1) Since PALD is identical to PAAD save for the learned threshold functions, we inherit feasibility from Lemma E.1. First, we lower bound the cost of OPT; as in the proof of Thm. 4.3, we partition the time horizon into active and inactive periods. Since the partitioning relies on the value of the base driver threshold function and PALD's is learned, we must use the definition of $\mathcal{R}_b$ to prove that OPT's cost during inactive periods can be bounded in the same manner. Next, we upper bound the cost of PALD by "charging" each driver's decisions to their corresponding threshold functions and accounting for any worst-case

purchasing, switching, and delivery costs that arise due to constraints. We show that the same charging argument from Thm. 4.3 still holds with learned thresholds. Finally, we use the definitions of $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$ to relate the cost of PALD to that of OPT. Using a similar contradiction argument as in the proof of Thm. 4.3, the last inequalities in the definition of the feasible sets are exactly those needed to show that the ratio $\frac{\mathsf{PALD}(\mathcal{I})-p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})}$ must be at most ρ, completing the proof. □

Remark. PALD is similar to certain existing approaches, but distinct in important ways. For instance, Zeynali et al. [93] propose learning the best exponent for an exponential threshold function in the online knapsack problem—our approach is distinguished by its expressiveness in allowing any arbitrary threshold function while retaining worst-case guarantees. Our concept of "learning a threshold" is structurally similar to the notion of learning dual functions in online convex optimization [50]—PALD is distinguished by the "robustness certificate," which additionally guarantees a certain (adversarial) competitive ratio. Li et al. [48] learn an ML model to solve online convex optimization that is calibrated (although not guaranteed) to be robust with respect to a baseline—in contrast, PALD allows us to specify the worst-case level performance we are willing to incur, and to learn an algorithm that performs well subject to that constraint with certainty.

6.2 Learning Methodology

In this section, we describe the methodology we use in our case study (see § 7) to instantiate a "proof-of-concept" of the PALD framework in practice. We parameterize each threshold function as a piecewise-affine function using a "knot" representation at a fixed grid of $K > 1$ points $\{0, 1/K-1, 2/K-1, \ldots, 1\}$. We learn a parameter vector $\mathbf{y} \in \mathbb{R}_+^K$ where each element defines the function's value at a knot (between knots, the function is interpolated linearly). We denote by $\mathbf{y}_b, \mathbf{y}_f, \mathbf{y}_\psi$ the parameter vectors for $\hat{\phi}_b, \hat{\phi}_f, \hat{\psi}_f$, respectively. This makes the robust sets tractable:

Lemma 6.4. *Letting $\hat{\phi}_b, \hat{\phi}_f, \hat{\psi}_f$ be piecewise-affine functions parameterized by $\mathbf{y}_b, \mathbf{y}_f, \mathbf{y}_\psi$ as outlined above, the feasible sets defined in Def. 6.1 and 6.2 are convex sets in $\mathbf{y}_b$ and $(\mathbf{y}_f, \mathbf{y}_\psi)$, respectively.*

The full proof is in Appendix G.3—the key idea is that the necessary robustness certificates for PALD reduce to box constraints, linear inequalities, and affine halfspaces, whose intersection is convex.

Recall that the analytical thresholds ϕ_b, ϕ_f and ψ_f (Defs. 4.1 & 4.2) are continuous and monotone non-increasing functions—by definition of the PAAD algorithm, these lie in the feasible sets $\mathcal{R}_b(\alpha)$ and $\mathcal{R}_f(\alpha)$, respectively. For ρ increasing away from α, the feasible sets $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$ become strictly larger, since the robustness conditions become easier to satisfy. Since piecewise-affine functions are known to be dense in continuous functions [17, 71], the analytical thresholds can be approximated arbitrarily well using piecewise-affine functions with sufficiently many knots. Thus, for a given robustness $\rho > \alpha$, the feasible sets $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$ are non-empty since they at least contain the analytical thresholds. Lemma 6.4 implies that it is efficient to *project* any learned piecewise-affine function into the feasible sets—this makes robustness tractable to enforce during learning, and unlocks several learning techniques which we discuss below.[5]

Directly Learning Thresholds. We implement the PALD algorithm by using differentiable convex optimization layers (e.g., CVXPYLayers [2]) to solve the pseudo-cost minimization problem at each time step. This allows backpropagation through the entire PALD procedure to minimize empirical competitive ratio on historical instances. Given a data set of historical instances $\mathcal{D} = \{\mathcal{I}_1, \ldots, \mathcal{I}_N\}$, we can use this differentiable implementation of PALD to find a single set of parameters $\mathbf{y}_b, \mathbf{y}_f, \mathbf{y}_\psi$ that minimizes the empirical competitive ratio on $\mathcal{D}$, using projected gradient descent (PGD) to

[5]In principle, PALD could use substantially different learning techniques than what we consider here—our main contribution is the notion of directly finding "the best algorithm" amongst a set of algorithms satisfying a given worst-case guarantee.

ensure that, during training, the learned thresholds lie in the robust sets $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$ for a target $\rho > \alpha$. We refer to this as PALD-S ("S" for simple) in our case study.

End-to-End Contextual Learning. While directly learning a single set of thresholds improves average-case performance, it cannot leverage instance-specific *context* that is extremely useful in practice. For instance, our case study in § 7 uses locational marginal price (LMP) data for different grids; LMPs are known to correlate with factors such as the time of day, the weather, and the season [73]. Using a learning method that is aware of such context can enable better performance in practice by learning thresholds that are tailored to likely operating conditions.

To realize this context-aware approach, we employ a neural network $F_\Theta(\mathbf{x}) \to \mathbf{y}_b, \mathbf{y}_f, \mathbf{y}_\psi$ (with parameters Θ) that takes as input a context vector $\mathbf{x}$ about a given driver and outputs predicted vectors $\mathbf{y}_b, \mathbf{y}_f, \mathbf{y}_\psi$ that define the threshold functions. The final layer of this neural network is a differentiable projection layer that projects the output parameters $\mathbf{y}_b, \mathbf{y}_f, \mathbf{y}_\psi$ into the feasible sets $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$ to ensure target robustness $\rho > \alpha$. We construct an unsupervised learning pipeline that simulates the differentiable PALD algorithm on a historical data set $\mathcal{D}$. For each instance, each new driver in PALD makes a forward pass through the neural network F_Θ to obtain its context-aware thresholds $\hat{\phi}_b, \hat{\phi}_f$, and $\hat{\psi}_f$. After observing the resulting performance, we backpropagate through the entire pipeline to learn a better mapping between context and thresholds. We refer to this technique as PALD-C in our case study, and give a diagram of it in Appendix A.3.

7 Case Study: A Grid-integrated Data Center with Local Energy Storage

We conclude with a case study application of OSDM and our algorithms for the motivating application of demand management in a grid-integrated data center with local energy storage.

Experimental Setup. We simulate a data center with co-located energy storage that purchases electricity from a real-time market and serves a mix of base and flexible demand (e.g., interactive and batch jobs, respectively).

▹ *Electricity Price Data.* We use a year of locational marginal price (LMP) data at a 15-minute granularity in four U.S. grid regions: California ISO (CAISO), Electric Reliability Council of Texas (ERCOT), PJM Interconnection (PJM), and ISO New England (ISONE) [29]. For PALD-Contextual, we also use day-ahead LMP forecasts at an hourly granularity as contextual features. We truncate any negative or small prices to \$1 per MWh to avoid a divide-by-zero in the competitive ratio, and cap prices at the 99.9^{th} percentile to remove outliers. We summarize key statistics in Table 1.

Table 1. Statistics for LMP traces. All traces span the time period 01/01/2024-12/31/2024 at a 15-minute granularity, and LMP data is reported in \$ (USD) per megawatt-hour (MWh).

Trace	Min.	Max.	Mean	Std. Dev.	Coeff. of Var. [29]
CAISO	1.00	523.85	45.63	34.91	0.765
ISONE	1.00	336.27	39.57	31.44	0.794
PJM	1.00	325.93	31.10	26.95	0.867
ERCOT	1.00	877.48	25.02	41.30	1.651

▹ *Demand Data.* To model the data center's demand, we use Alibaba's 2018 production trace [3], which provides seven days of task start and end times at a 1-second granularity. In our experiments, time steps are fixed to 15 minutes (matching the price data), so we *aggregate* the demand: we partition the trace into 15-minute buckets and compute a *weighted number of tasks* for each bucket based on the number of jobs and their active duration (a task that runs for the entire bucket contributes 1, while one that runs for only 5 minutes contributes 1/3). We plot this quantity for the full trace in the Appendix, in Fig. 6.

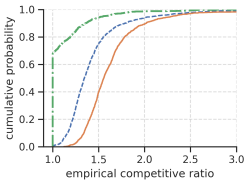
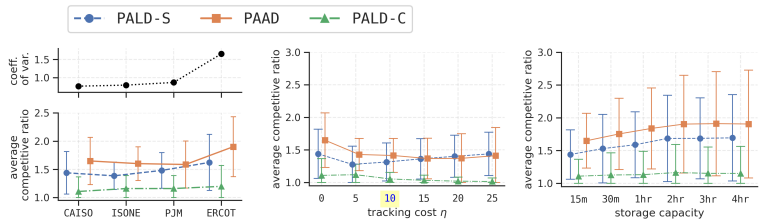


Fig. 3. CDFs of ECR for all algorithms in default CAISO experiments.



(a) Grid regions (b) Tracking cost η (c) Storage size

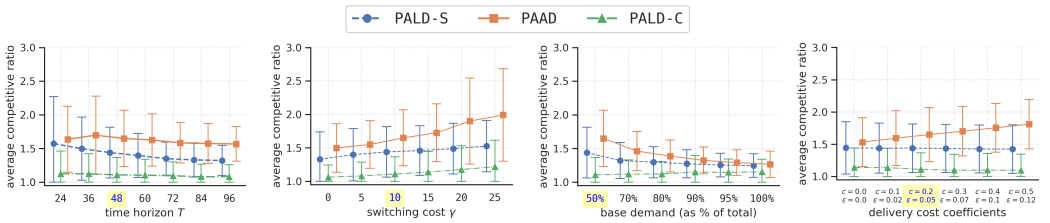
Fig. 4. Average ECR for all algorithms in varying regions, with a tracking cost, and varying storage size. In (a), top subfigure is coeff. of var. for each region.

Based on this, we compute a scaled demand as $^{\text{weighted no. of tasks}}/_{\text{scale factor}}$, where a scale factor is chosen based on the size of the energy storage. In most of our experiments, we set this factor to reflect a battery capacity that can meet the daily peak demand for a single (15-minute) time step on $^2/_7$ days in the trace. In some experiments, we vary this scale factor to study the effect of storage capacity. Since the trace does not specify task types, we also set a parameter $\texttt{prop_base} \in [0.5, 1]$ to probabilistically classify tasks as base or flexible: with probability $\texttt{prop_base}$, a task is "base", and otherwise it is "flexible." We typically set $\texttt{prop_base} = 0.5$, which yields instances that consist of half base and half flexible demand on average.

We simulate 1,200 instances of OSDM for each parameter configuration. In addition to the data parameters above, we isolate the effect of parameters including the delivery cost coefficients c, ε, the switching cost coefficients γ, δ, time horizon T, and the tracking cost coefficient η (for OSDM-T). Unless otherwise specified, we set $T = 48$ (i.e., a 12-hour horizon), $\gamma = 10$, $\delta = 5$, and $\eta = 0$ (i.e., no tracking cost). In experiments with a tracking cost (i.e., with non-zero η), we set a tracking target sequence $\{a_t\}_{t\in[T]}$ that evenly distributes the total demand over the time horizon (i.e., $a_t \approx {}^D/_T$, where D is the total demand), while randomly choosing between two to four time steps to set $a_t = 0$, simulating load-shedding events. We implement a price-dependent decreasing delivery cost as a soft penalty to encourage the decision maker to keep the battery charged, setting $c = 0.2$ and $\varepsilon = 0.05$ unless otherwise specified. These parameters quantify the risk tolerance of the data center as it relates to keeping the battery charged for, e.g., backup power.

▹ *Algorithms.* We implement and test three algorithms, denoted as PAAD, PALD-S, and PALD-C. We solve for the offline optimal solution using GurobiPy [31] to compute the empirical competitive ratio (ECR). PAAD is the competitive algorithm described in § 4, while PALD-S and PALD-C are instantiations of the learning framework described in § 6—we typically set a robustness factor of $\rho = 5\alpha$ (where α is the standard competitive ratio) to ensure robustness while allowing for some benefit from learning, and set $K = 10$. We implement the PALD algorithm using CVXPYLayers [2, 23] to optimize downstream task performance via backpropagation. PALD-S directly learns one set of threshold functions to minimize overall ECR—to capture seasonality in the traces, we train a set of PALD-S thresholds for each month in a given region's LMP data (i.e., 12 sets of thresholds total), using 100 random training instances for each month. In contrast, PALD-C is a neural network-based approach that uses context features to predict threshold functions for each driver—we train it on 100 random training instances using all 12 months in a given region's LMP data, using the time, month, tracking target (if $\eta > 0$), and statistics about day-ahead LMP forecasts (i.e., min, max, avg., std. dev.), as features. We defer more implementation details of PALD-S & PALD-C to Appendix A.3.

Experimental Results. We highlight key experiments in the main body, referring to Appendix A.2 for an investigation of algorithm runtimes and the effect of parameters δ and ρ. A summary is given in Fig. 3, which plots a cumulative distribution function (CDF) of the ECR for all tested algorithms in the CAISO region, with the default parameter settings. Our competitive algorithm



(a) Time horizon T (b) Switching cost γ (c) Demand mix (d) Delivery cost (c and ε)

Fig. 5. Average ECR for all algorithms in CAISO with varying parameters as specified.

PAAD is 1.65-competitive on average and $\leq$ 2.3-competitive in 95% of cases. By learning a single set of thresholds, PALD-S improves on PAAD by 12.7% on average and 8.7% at the 95$^{\text{th}}$ percentile. Finally, by leveraging contextual features about instances, PALD-C significantly improves on PAAD by 32.8% on average and 31.61% at the 95$^{\text{th}}$ percentile. Remarkably, PALD-C's performance is nearly optimal (i.e., ECR $\approx$ 1) in roughly 1/2 of instances, highlighting the value of a context-aware approach. We now consider the impact of grid region, tracking cost, and storage capacity on performance, training PALD once for each region, for the tracking cost $\eta = 10$, and for each capacity.

▹ *Grid Regions.* To capture the effect of different electricity price distributions and characteristics, Fig. 4(a) shows the results of the main experiment with the same parameters in three other regions, namely ERCOT, PJM, and ISONE. Both PALD-S and PALD-C are retrained on a test of training instances generated from each region. We expect to see some variance in performance across these regions, since the global price bounds ($p_{\min}$ and $p_{\max}$) are not homogeneous—interestingly, we find that the performance of algorithms roughly corresponds to the variability of prices in each region, with ERCOT (highest coefficient of variation) showing the most challenging case.

▹ *Tracking Case.* In Fig. 4(b), we show algorithm performance in a setting with a tracking cost (i.e., instead of a switching cost, so $\gamma = 0$), varying η between 0 and 25. Both PALD-S and PALD-C are retrained with $\eta = 10$. We find that the performance of PAAD and PALD-C slightly improves as η grows—PALD-S continues to improve on PAAD in the small η regime, but for large (out-of-distribution) $\eta > 15$, PALD-S's performance degrades because the single set of thresholds PALD-S learns becomes overly conservative (i.e., waits too long to purchase) when η grows.

▹ *Storage Capacities.* Recall that the default setting is where the storage is sized to meet the peak demand on two of the seven days in the Alibaba trace for a single 15-minute time slot—in Fig. 4(c), we plot the effect of larger storage sizes that meet up to 4 hours of the same peak demand on two of the seven days (i.e., a 16× increase in storage size). As the storage size increases and OPT gains flexibility for shifting, the performance of all algorithms slightly degrades, although PALD-C consistently outperforms the other algorithms across all storage cases.

Next, we plot the effect of parameters that are *out-of-distribution* for PALD. Throughout the plots, we *highlight* the instance that is in-distribution (i.e., parameters match training set).

▹ *Time Horizons.* Fig. 5(a) shows the effect of varying the time horizon T between 24 and 96 in CAISO. We find that the performance of all algorithms improves slightly with longer horizons, which we attribute to the increased flexibility in using storage over longer time periods. Notably, PALD-C consistently outperforms the other algorithms across all horizon lengths.

▹ *Switching Costs.* Fig. 5(b) shows the effect of varying the switching cost coefficient γ (with fixed $\delta = 5$) between 0 and 25 in CAISO. As predicted by the theoretical results, the performance of all algorithms degrades slightly as the switching cost coefficient γ increases—in this respect, PALD-S and PALD-C are less sensitive to increases in switching costs than PAAD.

▹ *Varying Demands.* We estimate the effect of varying demand distribution on the performance of all algorithms in the CAISO region. Recall that the default setting splits demand evenly between base and flexible demand (prop_base = 0.5)—in Fig. 5(c), we plot the effect of varying prop_base

between 0.5 and 1.0 in the CAISO region. We find that the performance of PAAD and PALD-S generally improves as the proportion of base demand increases (deviating further from the training instances), while PALD-C's performance slightly degrades, although it still outperforms the other algorithms.
▷ *Delivery Costs.* In Fig. 5(d), we show the effect of varying the delivery cost parameters c and ε (see Def. 2.4) in the CAISO region. As c and ε increase, the delivery cost has a more significant effect on the resulting cost. As predicted by the theoretical results, PAAD's performance degrades slightly as a function of c and ε, while the performance of PALD-S and PALD-C remains relatively constant.

8 Conclusion

Motivated by paradigm shifts promoting active and intelligent demand management models (e.g., demand response), we introduce and study online smoothed demand management (OSDM)—the first online problem to jointly capture key practical features such as mixed inflexible and flexible demand, local energy storage, and costs penalizing unsmooth behavior. We present PAAD, an algorithm achieving the optimal competitive ratio for OSDM, and PALD, a novel differentiable framework that learns the best algorithm from data while preserving worst-case guarantees. We evaluate both in a case study of a grid-integrated data center with co-located storage.

Several directions remain for future work. One natural extension is to incorporate *rate constraints*, where purchase and/or delivery rates are capped per time step as in some related models [46, 89]. While the monotone state- and price-dependent delivery cost we consider (Def. 2.4) is sufficiently general to capture some motivating applications, the delivery cost could be generalized—for instance, one might consider a cost that depends on a second adversarial price sequence (separate from p_t). More broadly, while our PALD learning framework is developed for OSDM, its approach of learning "the best algorithm" out of a set of algorithms that satisfy a given worst-case competitive ratio may generalize to other online problems, motivating further study of which problem classes and algorithm parameterizations admit such a framework.

References

[1] Abdul Afram and Farrokh Janabi-Sharifi. 2014. Theory and applications of HVAC control systems–A review of model predictive control (MPC). *Building and Environment* 72 (2014), 343–355.

[2] Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and Zico Kolter. 2019. Differentiable Convex Optimization Layers. In *Advances in Neural Information Processing Systems*.

[3] Alibaba. 2018. Cluster data collected from production clusters in Alibaba for cluster management research. https://github.com/alibaba/clusterdata/tree/master/cluster-trace-v2018

[4] Antonios Antoniadis, Christian Coester, Marek Elias, Adam Polak, and Bertrand Simon. 2020. Online Metric Algorithms with Untrusted Predictions. In *Proceedings of the 37th International Conference on Machine Learning*. PMLR, 345–355.

[5] Arsam Aryandoust and Johan Lilliestam. 2017. The potential and usefulness of demand response to provide electricity system services. *Applied Energy* 204 (Oct. 2017), 749–766. doi:10.1016/j.apenergy.2017.07.034

[6] Donald Azuatalam, Wee-Lih Lee, Frits De Nijs, and Ariel Liebman. 2020. Reinforcement learning for whole-building HVAC control and demand response. *Energy and AI* 2 (2020), 100020.

[7] Santiago Balseiro, Christian Kroer, and Rachitesh Kumar. 2023. Online Resource Allocation under Horizon Uncertainty. arXiv:2206.13606 [cs.DS] https://arxiv.org/abs/2206.13606

[8] Amrit S. Bedi, P.V. Aditya P., Md. Waseem Ahmad, S. Swapnil, Ketan Rajawat, and Sandeep Anand. 2018. Online algorithms for storage utilization under real-time pricing in smart grid. *International Journal of Electrical Power & Energy Systems* 101 (Oct. 2018), 50–59. doi:10.1016/j.ijepes.2018.02.034

[9] Ziyad Benomar, Lorenzo Croissant, Vianney Perchet, and Spyros Angelopoulos. 2025. Pareto-Optimality, Smoothness, and Stochasticity in Learning-Augmented One-Max-Search. arXiv:2502.05720 [cs.DS] https://arxiv.org/abs/2502.05720

[10] Michael Blonsky, Killian McKenna, Tyrone Vincent, and Adarsh Nagarajan. 2020. Time-of-use and Demand Charge Battery Controller Using Stochastic Model Predictive Control. In *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids*. 1–6. doi:10.1109/SmartGridComm47815.2020.9302943

[11] David Blum, Zhe Wang, Chris Weyandt, Donghun Kim, Michael Wetter, Tianzhen Hong, and Mary Ann Piette. 2022. Field demonstration and implementation analysis of model predictive control in an office HVAC system. *Applied Energy* 318, 119104 (2022), 10–1016.

[12] Allan Borodin, Nathan Linial, and Michael E. Saks. 1992. An Optimal On-Line Algorithm for Metrical Task System. *J. ACM* 39, 4 (Oct 1992), 745–763. doi:10.1145/146585.146588

[13] Paul Breeze. 2019. *Power System Energy Storage Technologies*. Elsevier, 219–249. https://doi.org/10.1016/B978-0-08-102631-1.00010-9

[14] Mohamad Fares El Hajj Chehade, Young ho Cho, Sandeep Chinchali, and Hao Zhu. 2024. Should we use model-free or model-based control? A case study of battery management systems. arXiv:2407.15313 [eess.SY] https://arxiv.org/abs/2407.15313

[15] Tianlong Chen, Xiaohan Chen, Wuyang Chen, Howard Heaton, Jialin Liu, Zhangyang Wang, and Wotao Yin. 2021. Learning to Optimize: A Primer and A Benchmark. arXiv:2103.12828 [math.OC] https://arxiv.org/abs/2103.12828

[16] Xingxing Chen, Jacob Feldman, Seung Hwan Jung, and Panos Kouvelis. 2022. Approximation schemes for the joint inventory selection and online resource allocation problem. *Production and Operations Management* 31, 8 (Aug. 2022), 3143–3159. doi:10.1111/poms.13742

[17] E.W. Cheney. 1998. *Introduction to Approximation Theory*. AMS Chelsea Pub.

[18] Wang Chi Cheung, Will Ma, David Simchi-Levi, and Xinshang Wang. 2021. Inventory Balancing with Online Learning. arXiv:1810.05640 [cs.AI] https://arxiv.org/abs/1810.05640

[19] Philip Colangelo, Ayse K. Coskun, Jack Megrue, Ciaran Roberts, Shayan Sengupta, Varun Sivaram, Ethan Tiao, Aroon Vijaykar, Chris Williams, Daniel C. Wilson, Brandon Records, Zack MacFarland, Daniel Dreiling, Nathan Morey, Anuja Ratnayake, and Baskar Vairamohan. 2025. AI data centres as grid-interactive assets. *Nature Energy* (Dec. 2025). doi:10.1038/s41560-025-01927-1

[20] Robert M Corless, Gaston H Gonnet, David EG Hare, David J Jeffrey, and Donald E Knuth. 1996. On the Lambert W function. *Advances in Computational Mathematics* 5 (1996), 329–359.

[21] Mohammadreza Daneshvaramoli, Helia Karisani, Adam Lechowicz, Bo Sun, Cameron N Musco, and Mohammad Hajiesmaili. 2025. Near-Optimal Consistency-Robustness Trade-Offs for Learning-Augmented Online Knapsack Problems. In *Forty-second International Conference on Machine Learning*. arXiv:2406.18752

[22] Nikhil R. Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A. Wilkens. 2019. Near Optimal Online Algorithms and Fast Approximation Algorithms for Resource Allocation Problems. *J. ACM* 66, 1, Article 7 (Jan. 2019), 41 pages. doi:10.1145/3284177

[23] Steven Diamond and Stephen Boyd. 2016. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research* 17, 83 (2016), 1–5.

[24] Ran El-Yaniv, Amos Fiat, Richard M. Karp, and Gordon Turpin. 2001. Optimal Search and One-Way Trading Online Algorithms. *Algorithmica* 30, 1 (May 2001), 101–139. doi:10.1007/s00453-001-0003-0

[25] Alex Elenter, Spyros Angelopoulos, Christoph Dürr, and Yanni Lefki. 2024. Overcoming Brittleness in Pareto-Optimal Learning Augmented Algorithms. In *Advances in Neural Information Processing Systems*, Vol. 37. 9329–9357. https://proceedings.neurips.cc/paper_files/paper/2024/file/11c6625b0481a7d5625831369f6b7c82-Paper-Conference.pdf

[26] Jamal Faraji, Ahmadreza Abazari, Masoud Babaei, SM Muyeen, and Mohamed Benbouzid. 2020. Day-ahead optimization of prosumer considering battery depreciation and weather prediction for renewable energy sources. *Applied Sciences* 10, 8 (2020), 2774.

[27] Joel Friedman and Nathan Linial. 1993. On convex body chasing. *Discrete & Computational Geometry* 9, 3 (Mar 1993), 293–321. doi:10.1007/bf02189324

[28] Stanley P.Y. Fung. 2021. Online two-way trading: Randomization and advice. *Theoretical Computer Science* 856 (Feb. 2021), 41–50. doi:10.1016/j.tcs.2020.12.016

[29] Grid Status. 2025. Grid Status. https://www.gridstatus.io

[30] Yuanxiong Guo and Yuguang Fang. 2013. Electricity Cost Saving Strategy in Data Centers by Using Energy Storage. *IEEE Transactions on Parallel and Distributed Systems* 24, 6 (June 2013), 1149–1160. doi:10.1109/tpds.2012.201

[31] Gurobi Optimization, LLC. 2024. Gurobi Optimizer Reference Manual. https://www.gurobi.com

[32] Mohammad H. Hajiesmaili, Chi-Kin Chau, Minghua Chen, and Longbu Huang. 2016. Online microgrid energy generation scheduling revisited: the benefits of randomization and interval prediction. In *Proceedings of the Seventh International Conference on Future Energy Systems* (Waterloo, Ontario, Canada) *(e-Energy '16)*. Association for Computing Machinery, New York, NY, USA, Article 1, 11 pages. doi:10.1145/2934328.2934329

[33] Xiaozhou He, Jie Xiang, Jin Xiao, T.C. E. Cheng, and Yuhang Tian. 2022. An online algorithm for the inventory retrieval problem with an uncertain selling duration, uncertain prices, and price-dependent demands. *Computers & Operations Research* 148 (Dec. 2022), 105991. doi:10.1016/j.cor.2022.105991

[34] Massil Hihat, Stéphane Gaïffas, Guillaume Garrigos, and Simon Bussy. 2023. Online Inventory Problems: Beyond the i.i.d. Setting with Online Convex Optimization. In *Advances in Neural Information Processing Systems*, Vol. 36. 20421–20440. https://proceedings.neurips.cc/paper_files/paper/2023/file/41128e5b3a7622da5b17588757599077-Paper-Conference.pdf

[35] Zhiyi Huang, Zhihao Gavin Tang, and David Wajc. 2024. Online Matching: A Brief Survey. *SIGecom Exch.* 22, 1 (Oct. 2024), 135–158. doi:10.1145/3699824.3699837

[36] Patrick Jaillet and Xin Lu. 2011. Online resource allocation problems. https://web.mit.edu/jaillet/www/general/online-resource-pj-xl-1-2011.pdf

[37] Tao Jiang, Yang Cao, Liang Yu, and Zhiqiang Wang. 2014. Load Shaping Strategy Based on Energy Storage and Dynamic Pricing in Smart Grid. *IEEE Transactions on Smart Grid* 5, 6 (Nov. 2014), 2868–2876. doi:10.1109/tsg.2014.2320261

[38] Lei Jiao, Antonia Maria Tulino, Jaime Llorca, Yue Jin, and Alessandra Sala. 2017. Smoothed Online Resource Allocation in Multi-Tier Distributed Cloud Networks. *IEEE/ACM Transactions on Networking* 25, 4 (2017), 2556–2570. doi:10.1109/TNET.2017.2707142

[39] Anthony Kim, Vahid Liaghat, Junjie Qin, and Amin Saberi. 2016. Online Energy Storage Management: an Algorithmic Approach. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPICS.APPROX-RANDOM.2016.12

[40] Max Klimm, Daniel Schmand, and Andreas Tönnis. 2019. *The Online Best Reply Algorithm for Resource Allocation Problems.* Springer International Publishing, 200–215. doi:10.1007/978-3-030-30473-7_14

[41] Xiao Kou, Yan Du, Fangxing Li, Hector Pulgar-Painemal, Helia Zandi, Jin Dong, and Mohammed M Olama. 2021. Model-based and data-driven HVAC control strategies for residential demand response. *IEEE Open Access Journal of Power and Energy* 8 (2021), 186–197.

[42] Ravi Kumar, Manish Purohit, and Zoya Svitkina. 2018. Improving Online Algorithms via ML Predictions. In *Advances in Neural Information Processing Systems*, Vol. 31. arXiv:2407.17712

[43] Kyung-Bin Kwon, Sayak Mukherjee, and Veronica Adetola. 2025. Operational Risks in Grid Integration of Large Data Center Loads: Characteristics, Stability Assessments, and Sensitivity Studies. arXiv:2510.05437 [eess.SY] https://arxiv.org/abs/2510.05437

[44] Kim S. Larsen and Sanne Wøhlk. 2010. Competitive analysis of the online inventory problem. *European Journal of Operational Research* 207, 2 (Dec. 2010), 685–696. doi:10.1016/j.ejor.2010.05.019

[45] Nevena Lazic, Craig Boutilier, Tyler Lu, Eehern Wong, Binz Roy, MK Ryu, and Greg Imwalle. 2018. Data center cooling using model-predictive control. *Advances in Neural Information Processing Systems* 31 (2018).

[46] Adam Lechowicz, Nicolas Christianson, Bo Sun, Noman Bashir, Mohammad Hajiesmaili, Adam Wierman, and Prashant Shenoy. 2024. Online Conversion with Switching Costs: Robust and Learning-augmented Algorithms. In *Proceedings of the 2024 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems* (Venice, Italy) *(SIGMETRICS / Performance '24)*. Association for Computing Machinery, New York, NY, USA.

[47] Todd Levin, Audun Botterud, W Neal Mann, Jonghwan Kwon, and Zhi Zhou. 2022. Extreme weather and electricity markets: Key lessons from the February 2021 Texas crisis. *Joule* 6, 1 (2022), 1–7.

[48] Pengfei Li, Jianyi Yang, and Shaolei Ren. 2022. Expert-Calibrated Learning for Online Optimization with Switching Costs. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 6, 2, Article 28 (June 2022), 35 pages. doi:10.1145/3530894

[49] Yunbo Li, Anne-Cecile Orgerie, and Jean-Marc Menaud. 2017. Balancing the Use of Batteries and Opportunistic Scheduling Policies for Maximizing Renewable Energy Consumption in a Cloud Data Center. In *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*. IEEE, 408–415. doi:10.1109/pdp.2017.24

[50] Alfonso Lobos, Paul Grigas, and Zheng Wen. 2021. Joint Online Learning and Decision-making via Dual Mirror Descent. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 7080–7089. https://proceedings.mlr.press/v139/lobos21a.html

[51] Julian Lorenz, Konstantinos Panagiotou, and Angelika Steger. 2008. Optimal Algorithms for k-Search with Application in Option Pricing. *Algorithmica* 55, 2 (Aug 2008), 311–328. doi:10.1007/s00453-008-9217-8

[52] Lian Lu, Jinlong Tu, Chi-Kin Chau, Minghua Chen, and Xiaojun Lin. 2013. Online energy generation scheduling for microgrids with intermittent energy sources and co-generation. *SIGMETRICS Perform. Eval. Rev.* 41, 1 (June 2013), 53–66. doi:10.1145/2494232.2465551

[53] Thodoris Lykouris and Sergei Vassilvtiskii. 2018. Competitive Caching with Machine Learned Advice. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 3296–3305. https://proceedings.mlr.press/v80/lykouris18a.html

[54] Rubén López-Rodríguez, Adriana Aguilera-González, Ionel Vechiu, and Seddik Bacha. 2021. Day-Ahead MPC Energy Management System for an Island Wind/Storage Hybrid Power Plant. *Energies* 14, 4 (Feb. 2021), 1066. doi:10.3390/en14041066

[55] A. Mamun, I. Narayanan, D. Wang, A. Sivasubramaniam, and H.K. Fathy. 2016. Multi-objective optimization of demand response in a datacenter with lithium-ion battery storage. *Journal of Energy Storage* 7 (Aug. 2016), 258–269. doi:10.1016/j.est.2016.08.002

[56] Mark Manasse, Lyle McGeoch, and Daniel Sleator. 1988. Competitive Algorithms for On-Line Problems. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing* (Chicago, Illinois, USA) *(STOC '88)*. Association for

Computing Machinery, New York, NY, USA, 322–333. doi:10.1145/62212.62243

[57] D. Mariano-Hernández, L. Hernández-Callejo, A. Zorita-Lamadrid, O. Duque-Pérez, and F. Santos García. 2021. A review of strategies for building energy management system: Model predictive control, demand side management, optimization, and fault detect & diagnosis. *Journal of Building Engineering* 33 (Jan. 2021), 101692. doi:10.1016/j.jobe.2020.101692

[58] Aranyak Mehta, Amin Saberi, Umesh Vazirani, and Vijay Vazirani. 2007. Adwords and generalized online matching. *Journal of the ACM (JACM)* 54, 5 (2007), 22–es.

[59] SeyedMorteza Mirhoseininejad, Ghada Badawy, and Douglas G Down. 2021. A data-driven, multi-setpoint model predictive thermal control system for data centers. *Journal of Network and Systems Management* 29, 1 (2021), 7.

[60] Dragoslav S. Mitrinovic, Josip E. Pečarić, and A. M. Fink. 1991. *Inequalities Involving Functions and Their Integrals and Derivatives*. Vol. 53. Springer Science & Business Media.

[61] Michael Mitzenmacher and Sergei Vassilvitskii. 2022. Algorithms with predictions. *Commun. ACM* 65, 7 (June 2022), 33–35. doi:10.1145/3528087

[62] Yanfang Mo, Qiulin Lin, Minghua Chen, and Si-Zhao Joe Qin. 2021. Optimal Online Algorithms for Peak-Demand Reduction Maximization with Energy Storage. In *Proceedings of the Twelfth ACM International Conference on Future Energy Systems (e-Energy '21)*. ACM, 73–83. doi:10.1145/3447555.3464857

[63] Hamed Mohsenian-Rad. 2016. Optimal Bidding, Scheduling, and Deployment of Battery Systems in California Day-Ahead Energy Market. *IEEE Transactions on Power Systems* 31, 1 (2016), 442–453. doi:10.1109/TPWRS.2015.2394355

[64] Attracta Mooney, Alice Hancock, and Amy Kazmin. 2025. Europe's electricity system tested by heatwaves as air-conditioning demand soars. *Financial Times* (3 8 2025). https://www.ft.com/content/23b3dc59-b40f-48e2-ad93-e301de7ac5f2

[65] Juan Moreno-Castro, Victor Samuel Ocaña Guevara, Lesyani Teresa León Viltre, Yandi Gallego Landera, Oscar Cuaresma Zevallos, and Miguel Aybar-Mejía. 2023. Microgrid Management Strategies for Economic Dispatch of Electricity Using Model Predictive Control Techniques: A Review. *Energies* 16, 16 (Aug. 2023), 5935. doi:10.3390/en16165935

[66] Thomas Morstyn, Branislav Hredzak, Ricardo P Aguilera, and Vassilios G Agelidis. 2017. Model predictive control for distributed microgrid battery energy storage systems. *IEEE Transactions on Control Systems Technology* 26, 3 (2017), 1107–1114.

[67] Neda Nasiriani, George Kesidis, and Di Wang. 2017. Optimal Peak Shaving Using Batteries at Datacenters: Characterizing the Risks and Benefits. In *2017 IEEE 25th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE, 164–174. doi:10.1109/mascots.2017.27

[68] National Renewable Energy Laboratory. 2024. Thermal Energy Storage. https://www.nrel.gov/buildings/storage

[69] Leonardo Nicoletti, Naureen Malik, and Andre Tartar. 2024. AI Needs So Much Power, It's Making Yours Worse. *Bloomberg* (26 dec 2024). https://www.bloomberg.com/graphics/2024-ai-power-home-appliances/

[70] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. arXiv:1912.01703 [cs.LG] https://arxiv.org/abs/1912.01703

[71] Allan Pinkus. 2005. Density in Approximation Theory. arXiv:math/0501328 [math.CA] https://arxiv.org/abs/math/0501328

[72] Corina Pons and Andrei Khalip. 2025. Spain's grid operator warns of new voltage swings, urges measures to avoid blackout. *Reuters* (8 10 2025). https://www.reuters.com/business/energy/spains-grid-operator-warns-new-tension-swings-urges-measures-avoid-blackout-2025-10-08/

[73] Ziyu Qu, Li He, Xinxin Ge, Fei Wang, Fei Xu, and Jinling Lu. 2024. A Two-Stage Forecasting Approach for Day-Ahead Electricity Price Based on Improved Wavelet Neural Network With ELM Initialization. *IEEE Transactions on Industry Applications* 60, 3 (May 2024), 5061–5073. doi:10.1109/tia.2024.3365456

[74] Ana Radovanovic, Ross Koningstein, Ian Schneider, Bokan Chen, Alexandre Duarte, Binz Roy, Diyue Xiao, Maya Haridasan, Patrick Hung, Nick Care, et al. 2022. Carbon-Aware Computing for Datacenters. *IEEE Transactions on Power Systems* (2022).

[75] Rémy Rigo-Mariani and Vincent Debucsshere. 2022. An ADMM-based Coordination Strategy for the Control of Distributed Storage at the Household Level -Impact of the End-User Settings. In *ELECTRIMACS 2022 (ELECTRIMACS 2022)*. Nancy, France. https://hal.science/hal-03675572

[76] Oliver Ruhnau, Lion Hirth, and Aaron Praktiknjo. 2019. Time series of heat demand and heat pump efficiency for energy system modeling. *Scientific Data* 6, 1 (Oct. 2019). doi:10.1038/s41597-019-0199-y

[77] Manuel Sage, Joshua Campbell, and Yaoyao Fiona Zhao. 2024. Enhancing Battery Storage Energy Arbitrage With Deep Reinforcement Learning and Time-Series Forecasting. In *ASME 2024 18th International Conference on Energy Sustainability (ES2024)*. American Society of Mechanical Engineers. doi:10.1115/es2024-130538

[78] Gianluca Serale, Massimo Fiorentini, Alfonso Capozzoli, Daniele Bernardini, and Alberto Bemporad. 2018. Model predictive control (MPC) for enhancing building and HVAC system energy efficiency: Problem formulation, applications and opportunities. *Energies* 11, 3 (2018), 631.

[79] Vanderlei Aparecido Silva, Alexandre Rasi Aoki, and Germano Lambert-Torres. 2020. Optimal Day-Ahead Scheduling of Microgrids with Battery Energy Storage System. *Energies* 13, 19 (Oct. 2020), 5188. doi:10.3390/en13195188

[80] Eleni Stai, Cong Wang, and Jean-Yves Le Boudec. 2021. Online Battery Storage Management via Lyapunov Optimization in Active Distribution Grids. *IEEE Transactions on Control Systems Technology* 29, 2 (March 2021), 672–690. doi:10.1109/tcst.2020.2975475

[81] Bo Sun, Russell Lee, Mohammad Hajiesmaili, Adam Wierman, and Danny Tsang. 2021. Pareto-Optimal Learning-Augmented Algorithms for Online Conversion Problems. In *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (Eds.), Vol. 34.

[82] Bo Sun, Ali Zeynali, Tongxin Li, Mohammad Hajiesmaili, Adam Wierman, and Danny H.K. Tsang. 2021. Competitive Algorithms for the Online Multiple Knapsack Problem with Application to Electric Vehicle Charging. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 4, 3, Article 51 (June 2021), 32 pages.

[83] Jun Sun, Shibo Chen, Pengcheng You, Qinmin Yang, and Zaiyue Yang. 2023. Battery-Assisted Online Operation of Distributed Data Centers With Uncertain Workload and Electricity Prices. *IEEE Transactions on Cloud Computing* 11, 2 (April 2023), 1303–1314. doi:10.1109/tcc.2021.3132174

[84] Prajina Tandukar, Labi Bajracharya, Timothy M. Hansen, Robert Fourney, Ujjwol Tamrakar, and Reinaldo Tonkoski. 2018. Real-time Operation of a Data Center as Virtual Power Plant Considering Battery Lifetime. In *2018 International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM)*. IEEE, 81–86. doi:10.1109/speedam.2018.8445345

[85] Rahul Urgaonkar, Bhuvan Urgaonkar, Michael J. Neely, and Anand Sivasubramaniam. 2011. Optimal power cost management using stored energy in data centers. In *Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems* (San Jose, California, USA) *(SIGMETRICS '11)*. Association for Computing Machinery, New York, NY, USA, 221–232. doi:10.1145/1993744.1993766

[86] Kaifeng Wang, Lin Ye, Shihui Yang, Zhanfeng Deng, Jieying Song, Zhuo Li, and Yongning Zhao. 2023. A hierarchical dispatch strategy of hybrid energy storage system in internet data center with model predictive control. *Applied Energy* 331 (2023), 120414.

[87] Lin Wang, Lei Jiao, Jun Li, and Max Muhlhauser. 2017. Online Resource Allocation for Arbitrary User Mobility in Distributed Edge Clouds. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE. doi:10.1109/icdcs.2017.30

[88] Jianyi Yang, Pengfei Li, Mohammad Jaminur Islam, and Shaolei Ren. 2024. Online Allocation with Replenishable Budgets: Worst Case and Beyond. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 8, 1, Article 4 (Feb. 2024), 34 pages. doi:10.1145/3639030

[89] Lin Yang, Mohammad H. Hajiesmaili, Ramesh Sitaraman, Adam Wierman, Enrique Mallada, and Wing S. Wong. 2020. Online Linear Optimization with Inventory Management Constraints. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 4, 1, Article 16 (may 2020), 29 pages. doi:10.1145/3379482

[90] Yijie Yang, Jian Shi, Dan Wang, Chenye Wu, and Zhu Han. 2025. Carbon-Aware Scheduling of Thermostatically Controlled Loads: A Bilevel DRCC Approach. *IEEE Transactions on Smart Grid* 16, 2 (2025), 1233–1247.

[91] Liang Yu, Tao Jiang, Yang Cao, and Qi Qi. 2015. Joint Workload and Battery Scheduling with Heterogeneous Service Delay Guarantees for Data Center Energy Cost Minimization. *IEEE Transactions on Parallel and Distributed Systems* 26, 7 (July 2015), 1937–1947. doi:10.1109/tpds.2014.2329491

[92] Xianyi Zeng, Yuanliang Zhang, Bin Guo, Liyu Huang, and Chuangzhi Li. 2023. Optimal Day-Ahead Dispatch of Air-Conditioning Load under Dynamic Carbon Emission Factors. In *2023 5th Asia Energy and Electrical Engineering Symposium (AEEES)*. 1–6. doi:10.1109/AEEES56888.2023.10114285

[93] Ali Zeynali, Bo Sun, Mohammad Hajiesmaili, and Adam Wierman. 2021. Data-driven Competitive Algorithms for Online Knapsack and Set Cover. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 12 (May 2021), 10833–10841. doi:10.1609/aaai.v35i12.17294

[94] Mingze Zhang, Weidong Li, Samson Shenglong Yu, Haixia Wang, and Yu Ba. 2024. Optimal day-ahead large-scale battery dispatch model for multi-regulation participation considering full timescale uncertainties. *Renewable and Sustainable Energy Reviews* 189 (Jan. 2024), 113963. doi:10.1016/j.rser.2023.113963

[95] Ying Zhang, Mohammad H. Hajiesmaili, Sinan Cai, Minghua Chen, and Qi Zhu. 2018. Peak-Aware Online Economic Dispatching for Microgrids. *IEEE Transactions on Smart Grid* 9, 1 (2018), 323–335. doi:10.1109/TSG.2016.2551282

[96] Zijun Zhang, Zongpeng Li, and Chuan Wu. 2017. Optimal Posted Prices for Online Cloud Resource Allocation. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1, 1, Article 23 (June 2017), 26 pages.

[97] Yunhong Zhou, Deeparnab Chakrabarty, and Rajan Lukose. 2008. Budget Constrained Bidding in Keyword Auctions and Online Knapsack Problems. In *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 566–576.

Appendix

A Deferred Experiment Setup and Results

In this section, we provide additional experimental results and details that were deferred from § 7. We start by discussing the runtime of our algorithms in Appendix A.1, before presenting additional experimental results in Appendix A.2 and presenting details of the PALD implementations in Appendix A.3. Note that Fig. 6 plots the "*weighted active jobs*" metric used to model demand using the Alibaba trace [3], described in § 7.

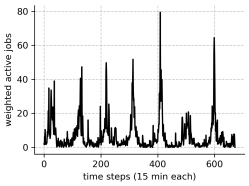


Fig. 6. Weighted active jobs over time (bucketed in 15-min intervals) for the Alibaba trace [3]

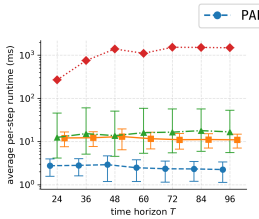


Fig. 7. Average per-time-step runtime (in ms) for all algorithms with varying instance lengths T.

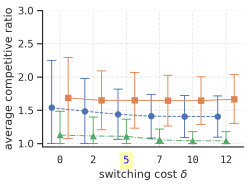


Fig. 8. Average ECR for all algorithms in CAISO with varying delivery switching coefficients δ.

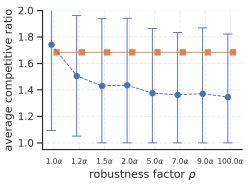


Fig. 9. Average ECR for PALD-S in default CAISO experiment trained with varying ρ.

A.1 Runtime Measurements, Discussion, and Optimizations

Building off the discussion in § 4.1, we provide additional details about the empirical runtime of our algorithms and discuss some practical optimizations that can be used to improve the runtime of PAAD and PALD.

In Fig. 7, we show the average per-time-step runtime (in milliseconds) for all algorithms with varying instance lengths T, averaging over 8,400 instances (1,200 for each setting of T). We also include the runtime of the offline solver (computed as a bilinear program using GurobiPy [31]), normalized by the instance length T to give a fair comparison with the online algorithms. Each instance and each algorithm was run in a single thread on a MacBook Air with M2 processor and 24 GB of RAM. Unsurprisingly, we find that the offline optimal solution is the most computationally expensive, exhibiting a "per-time-step" runtime an order of magnitude higher than the next closest online algorithm. PAAD and PALD-C have comparable runtimes, with averages of 62.5 ms and 65.7 ms per time step, respectively, which remains practical for real-time decision-making.

Interestingly, we find that PALD-S is the fastest online algorithm, with an average runtime of just 13.9 ms per time step. We find that this is because of the re-parameterization of the psuedo-cost minimization problem in both variants of PALD (see Appendix A.3). Specifically, when solving the pseudo-cost minimization problem for each driver (see Algorithm 1), our PAAD implementation solves the exact integral over the threshold function $\phi(\cdot)$, while our PALD implementation uses a squared hinge re-parameterization that is faster for the CVXPY solver. PALD-S is particularly fast because it uses a single set of thresholds for all time steps, allowing the CVXPY solver to cache and re-use computations across time steps. PALD-C makes a forward pass through a neural network to predict the thresholds whenever a driver is created and project them into the feasible sets, which adds some overhead to the per-time-step runtime and brings it closer to PAAD in runtime.

PALD-S's performance improvement demonstrates that several optimizations can be used to speed-up the core psuedo-cost minimization problem, e.g., by caching results, using approximations, or re-parameterizing the integral—we do not explore these optimizations in depth in this work,

but they offer further evidence to suggest that the PAAD and PALD frameworks are practical for real-time decision-making in large-scale systems.

We also remark that additional driver-level optimizations can be employed to reduce the number of times that the psuedo-cost problem needs to be solved without affecting the theoretical results. Perhaps the highest impact of these optimizations is *driver consolidation*—if multiple drivers have the same type and relative state, they can be consolidated into a single driver with a larger size, allowing one solution to the pseudo-cost problem to essentially cover multiple drivers. Formally, if for example, two drivers i and j have the same type (i.e., $d^{(i)} = d^{(j)}$) and the same relative state (i.e., $w_b^{(i)}/d^{(i)} = w_b^{(j)}/d^{(j)}$), then they can be consolidated into a single driver with size $d^{(i)} + d^{(j)}$. We did not implement this optimization in our experiments, but it could be effective in practice to even further reduce the number of times that the pseudo-cost problem needs to be solved.

A.2 Deferred Experimental Results

In this section, we present additional results and figures to complement those in the main body. In subsequent figures (discussed below), we give experiment results for the effect of varying delivery switching coefficient δ and varying the robustness factor ρ in our PALD implementation.

▹ *Delivery Switching Cost.* Fig. 8 shows the effect of varying the switching cost coefficient δ (with γ fixed at $\gamma = 10$) between 0 and 12.5 in CAISO—other parameters are set to the experiment defaults (see § 7). We find that the performance of all algorithms slightly improves as the switching cost coefficient δ increases.

▹ *Robustness Factor ρ.* In Fig. 9, we show the effect of varying the robustness factor ρ in our PALD-S implementation in CAISO. Recall that ρ controls the degree of robustness guaranteed by the PALD framework. In our main experiments, we typically set $\rho = 5\alpha$. As $\rho \to \alpha$, the feasible sets for learning thresholds shrink to approximately those thresholds that would be valid for the robust PAAD algorithm (see § 6.2). On the other hand, as $\rho \to \infty$, the feasible sets become unconstrained, allowing PALD-S to learn thresholds that may not be valid for any competitive online algorithm. For PALD-S, we find that there is a steep drop in empirical competitive ratio as ρ increases from α to 2α, after which the performance stabilizes. This suggests that there is a "sweet spot" for ρ that balances robustness and flexibility, and that PALD-S can learn effective thresholds even when the feasible sets are relatively constrained. This also shows that in the case of PALD-S, where we just learn a single set of thresholds for each month in the LMP data, there is a practical limit to how well PALD-S can perform, as it cannot adapt to a specific instance and must "average over" many instances with different characteristics.

A.3 PALD Implementation Details

In this section, we provide additional details about our implementations of the PALD framework in our case study (see § 7)

We implement the PALD algorithm using CVXPYLayers [2, 23] to enable optimization of the downstream task (i.e., minimizing the empirical competitive ratio on training instances) via back-propagation through the optimization problems that define PALD's decisions.

As outlined in § 6.2, PALD-S directly learns one set of piecewise-affine threshold functions to minimize overall empirical competitive ratio on training instances. We implement projected gradient descent using PyTorch [70] and CVXPY [23] to project the learned thresholds (specifically the parameter vectors) into the feasible sets. To capture some degree of seasonality, we train a set of PALD-S thresholds for each month in a given region's LMP data (i.e., 12 sets of thresholds in total), using 100 random training instances for each month. In Fig. 10, we give a high-level diagram of the PALD-S learning framework.

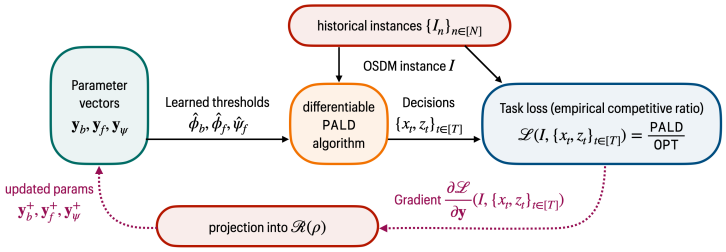


Fig. 10. PALD-S learns a single set of thresholds (via the parameter vectors $\mathbf{y}_b, \mathbf{y}_f, \mathbf{y}_\psi$) to directly optimize a downstream task loss of the empirical competitive ratio.

In contrast, PALD-C is a neural network-based approach (see § 6.2) that uses contextual features to predict a set of piecewise-affine threshold functions for each driver. In our implementation, we use a PyTorch neural network with two hidden layers (64 neurons each) and ReLU activations. The output of this neural network is a monotone head that ensures the output parameter vectors (3 K-length vectors for the base and flexible driver threshold functions) are non-increasing by construction: we use a softplus activation to ensure non-negativity, and then take a cumulative sum in reverse order to ensure monotonicity. The output of the monotone head is then passed through a CVXPYLayer that projects the outputs into the feasible sets. We train PALD-C on 100 random training instances using all 12 months in a given region's LMP data, using the time, month, tracking target (if applicable), and statistics about the day-ahead LMP forecasts for future time steps (i.e., min, max, average, standard deviation), as the contextual features. In Fig. 11, we give a high-level diagram of the PALD-C learning framework.

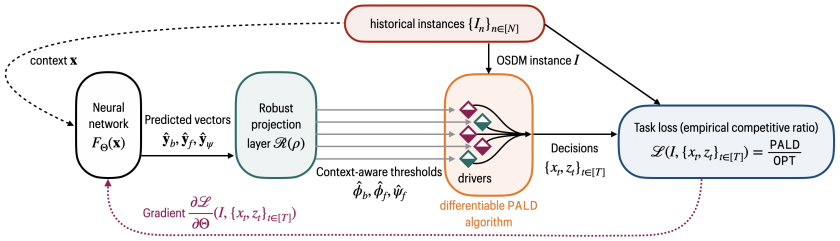


Fig. 11. PALD-C learns a mapping between context about each driver (see Algorithm 2) and the instance-specific threshold functions that minimize overall task loss. Gradients flow through the differentiable PALD implementation and the final projection layer of the neural network to improve this mapping.

In our implementation of both PALD-S and PALD-C, we use $K = 10$ grid points to parameterize the piecewise-affine threshold functions. We restrict the thresholds to lie in a concave region by enforcing that the second-order finite differences of the parameter vectors are non-positive. The reason we do this is because we use a nonnegative sum of squared hinges to parameterize the computation of the integral over the threshold functions in the pseudo-cost minimization problem (i.e., $-\int_w^{w+x} \phi_b(u)du$, $-\int_w^{w+x} \phi_f(u)du$, and $-\int_v^{v+x} \psi_f(u)du$, see Algorithm 1 for details). This parameterization is convex and DPP (s disciplined parameterized program) in x if the thresholds are linear decreasing or concave decreasing. We implement this parameterization to employ CVXPYLayers, which require the optimization problem to be DPP in the optimization variables (i.e., x) and the parameters (i.e., the threshold function parameters). However, we note that this concavity constraint is not required for the theoretical guarantees of PALD and could likely be circumvented with a different implementation—the underlying integral is analytically convex in x for any non-increasing threshold function.

B Deferred Discussion from § 2

In this section, we provide some deferred discussion about motivating examples for the OSDM problem introduced in § 2.

Grid-integrated Data Center with Storage. Consider a grid-connected data center with local energy storage. At each discrete time step t (e.g., 15-minute intervals), the data center must decide how much energy to purchase (x_t) from the grid at a price p_t, and how much energy to deliver (i.e., use now) to meet demand (z_t)—the difference between these quantities is stored in (resp. delivered from) the local energy storage with state of charge s_t. In this setting, base demands b_t model e.g., interactive workloads that must be satisfied immediately, while flexible demands f_t model e.g., delay-tolerant batch jobs that can be deferred until a deadline $\Delta_t > t$. The data center's goal is to minimize its electricity cost ($p_t x_t$) while smoothing its purchasing rate ($\mathcal{S}(\cdot)$), since large fluctuations in electricity consumption negatively impact the grid [69]. The data center may also wish to place a small penalty on the cost of delivering energy ($\mathcal{D}(\cdot)$) from the battery to e.g., amortize the battery's degradation over time, as well as a penalty on the smoothness of the delivery rate (i.e., its own rate of energy consumption within the data center) to reduce wear-and-tear on internal components ($\delta|z_t - z_{t-1}|$). Compared to prior work that has studied demand management in data centers [89], OSDM captures the practical considerations of *both* flexible and inflexible demand, as well as the desire to make "smooth" decisions on the grid- and demand-sides of the problem.

Thermal Energy Demand Management. Consider a thermal energy system with a local energy storage tank [68]. At each time step t, the operator must decide how much thermal energy to purchase (x_t) at a price p_t (e.g., by converting grid electricity into heat), and how much thermal energy to deliver to meet demand (z_t)—the difference between these quantities is stored in (resp. delivered from) the local storage tank with state of charge s_t. Base demands b_t model e.g., heating or cooling loads that must be satisfied immediately, while flexible demands f_t model e.g., delay-tolerant heating or cooling loads that can be deferred until a deadline $\Delta_t > t$. The operator's goal is to minimize their energy cost ($p_t x_t$) while smoothing their purchasing rate ($\mathcal{S}(\cdot)$). The delivery cost ($\mathcal{D}(\cdot)$) captures the extra input energy required to drive the delivery system (e.g., pumps, heat exchangers, etc.), which may depend on the price of energy (p_t). The operator may also wish to place a small penalty on the smoothness of the delivery rate (i.e., its own rate of thermal energy consumption) to reduce wear-and-tear on distribution components such as valves ($\delta|z_t - z_{t-1}|$).

Just-in-Time Manufacturing with Material Inventory. Consider a factory that processes raw materials into a finished product, and has a warehouse to store raw materials. At each time step t, the factory must decide how much raw material to purchase (x_t) at a price p_t, and how much finished product to deliver to meet demand (z_t)—the difference between these quantities is stored in (resp. delivered from) the warehouse with state s_t. Orders for the product arrive over time, where base demands b_t model e.g., orders that must be satisfied immediately, while flexible demands f_t model scheduled orders that can be deferred until a deadline $\Delta_t > t$. The factory's goal is to minimize their material cost ($p_t x_t$) while smoothing their purchasing rate ($\mathcal{S}(\cdot)$), which can reduce supply chain uncertainty and lead to discounts from suppliers. The delivery cost ($\mathcal{D}(\cdot)$) captures the processing cost for converting raw materials into the finished product at time t. The factory may also wish to place a small penalty on the smoothness of the delivery rate (i.e., its own rate of production) to encourage a more predictable production schedule ($\delta|z_t - z_{t-1}|$).

Flow Battery Storage Management. Consider a large energy consumer that operates a grid-connected flow battery to store energy during off-peak hours. Flow batteries are a type of rechargeable battery where energy is stored in the form of two (or more) liquids contained in external tanks, and energy is extracted by pumping the liquids through a cell stack to generate electricity [13]. At each time step t, the operator must decide how much electricity to purchase (x_t) from the grid at a

price p_t, and how much electricity to deliver to meet demand (z_t)—the difference between these quantities is stored in (resp. delivered from) the flow battery with state of charge s_t. The operator's goal is to minimize their electricity cost ($p_t x_t$) while smoothing their purchasing rate ($\mathcal{S}(\cdot)$). The delivery cost ($\mathcal{D}(\cdot)$) captures the extra input energy required to drive the flow battery system (e.g., pumps, valves, etc.), which may depend on the price of energy (p_t).

C Details of the OCS Problem and the RORO Algorithm

In this section, we provide deferred details about the OCS problem and the "ramp-on, ramp-off" (RORO) algorithm proposed to solve it optimally [46], which we discuss in the warmup (see § 3). We start by formally defining the OCS problem, which can be cast as a special case of OSDM with a single unit of flexible demand and no storage. We then describe the RORO algorithm and its guarantee in Appendix C.2.

C.1 Online Conversion with Switching Costs (OCS) [46]

Problem Statement. Consider an operator who must purchase an asset of unit size (without loss of generality) before a given deadline T while minimizing their total cost. At each time step $t \in [T]$, a price p_t arrives online, and the decision-maker must choose the amount of the asset to purchase at the current time step, represented by $x_t \in [0, 1]$. Given a decision x_t, the operator's cost at time t is $p_t x_t + \gamma |x_t - x_{t-1}|$, where the first term is the cost due to the time-varying price, and the second term penalizes the decision-maker for "unsmooth jumps" across time steps. The coefficient $\gamma > 0$ is known a priori. An offline formulation of OCS is as follows:

$$[\text{OCS}] \quad \min_{\mathbf{x} := \{x_t\}_{t \in T}} \underbrace{\sum_{t=1}^{T} p_t x_t}_{\text{total purchasing cost}} + \underbrace{\sum_{t=1}^{T+1} \gamma |x_t - x_{t-1}|}_{\text{switching penalty}}, \tag{6}$$

$$\text{s.t.} \quad \underbrace{\sum_{t=1}^{T} x_t = 1}_{\text{deadline constraint}}, \qquad x_t \in [0, d_t] \ \forall t \in [T]. \tag{7}$$

Lechowicz et al. [46] focus on the online version of OCS, where the decision-maker can only observe price signals up to time t when selecting x_t, and each choice of x_t is irrevocable (i.e., it cannot be revised at future time steps). The goal is to design an online algorithm that achieves a small competitive ratio (see Def. 2.1) They make the following assumptions on the problem:

Assumptions. Prices have bounded support, i.e., $p_t \in [p_{\min}, p_{\max}] \ \forall t \in [T]$, where $p_{\min}$ and $p_{\max}$ are known positive constants. They also assume that the switching coefficient γ is "not too large". Formally, it is bounded within $\gamma \in \left[0, \frac{(p_{\max} - p_{\min})}{2}\right)$. If γ exceeds this range, its impact on the total cost exceeds that of the prices, and any competitive algorithm should simply minimize this term, making decision-making trivial [46]. They typically assume that the deadline T is known in advance to facilitate a "compulsory trade" that ensures the entire asset is purchased before the end of the sequence. If the operator has completed w_T fraction of the workload at time T, they are forced to purchase the remaining $1 - w_T$ fraction at time T.

C.2 The RORO Algorithm [46]

In the classic competitive setting, OCS is solved optimally by an online algorithm framework called "ramp-on, ramp-off" (RORO), shown by [46]. In what follows, we present this algorithm to contextualize our results in § 3.

Algorithm 3 Online Ramp-On, Ramp-Off Algorithm (RORO) [46]

input: threshold function $\phi(w) : [0,1] \to [p_{\min}, p_{\max}]$
initialize: initial decision $x_0 = 0$, current utilization $w_0 = 0$
while price p_t is revealed and $w_{t-1} < 1$ **do**:
    solve **pseudo-cost minimization problem** to obtain decision:

$$x_t \leftarrow \underset{x \in [0, 1-w_{t-1}]}{\arg\min} \; p_t x_t + \gamma |x - x_{t-1}| - \int_{w_{t-1}}^{w_{t-1}+x} \phi(u) du$$

    update utilization as $w^{(t)} = w_{t-1} + x_t$

In RORO [46], the online decision at each time step is made by solving a *pseudo-cost minimization problem* to determine the amount to purchase at the current time step (i.e., $x_t \in [0, 1 - w_{t-1}]$). This minimization balances between the extreme options of buying "too much" early (thus incurring suboptimal costs if prices later drop) and waiting too long to purchase (risking being forced to purchase a large portion at once, potentially at high cost).

Whenever the price is "sufficiently attractive", the pseudo-cost minimization finds the best decision that purchase just enough to maintain a certain competitive ratio. To define this trade-off, Lechowicz et al. [46] introduce a *dynamic threshold function* $\phi(w) : [0,1] \to [p_{\min}, p_{\max}]$, which is a function that maps the current amount that has been purchased w to a price value. It is defined as:

$$\phi(w) = p_{\max} - \gamma + \left(\frac{p_{\max}}{\alpha_{\text{RORO}}} - p_{\max} + 2\gamma \right) \exp\left(\frac{w}{\alpha_{\text{RORO}}} \right), \tag{8}$$

where α_{RORO} is the optimal competitive ratio for OCS, defined as the unique positive solution to $\exp\left(\frac{1}{\alpha_{\text{RORO}}}\right) = \frac{p_{\max} - p_{\min} - 2\gamma}{p_{\max}/\alpha_{\text{RORO}} - p_{\max} - 2\gamma}$ [46]:

$$\alpha_{\text{RORO}} \coloneqq \left[W\left(\left(\frac{2\gamma + p_{\min}}{p_{\max}} - 1 \right) \exp\left(\frac{2\gamma}{p_{\max}} - 1 \right) \right) - \frac{2\gamma}{p_{\max}} + 1 \right]^{-1}. \tag{9}$$

In the above, $W(\cdot)$ is the Lambert W function [20]. Given this definition of α_{RORO}, note that $\phi(\cdot)$ is monotonically decreasing on the interval $w \in [0,1]$ (i.e., non-increasing in the amount of demand that has already been purchased), and captures the marginal trade-off of making progress towards satisfying the single unit of demand. We summarize the RORO algorithm in Algorithm 3.

D Deferred Proofs from § 3

In this section, we give a full description and analysis of the warmup "doubling strategy" for RORO. We first give the pseudocode for this extension in Algorithm 4. The notation $\texttt{instance}_j(d)$ denotes the j^{th} instance of RORO that is created with a demand of size $\hat{d}$. Note that each instance of RORO operates until all of the flexible demands assigned to it are satisfied (or until their deadlines have passed), at which point they always return 0 as their purchasing decision.

We now prove Theorem 3.1, which states that under the assumptions in § 2.5 and § 3, the "doubling extension" of RORO defined in § 3 is ζ-competitive for OSDM with only flexible demand and no delivery cost, where ζ is at least:

$$\zeta \geq \min \left\{ \frac{\alpha_{\text{RORO}}}{1-\sigma} + \frac{\alpha_{\text{RORO}}}{1-\sigma} \cdot \frac{\left(p_{\max} - 2\gamma - \frac{p_{\max}}{\alpha_{\text{RORO}}} \right) \exp\left(\frac{\sigma}{\alpha_{\text{RORO}}} \right) - \frac{\sigma p_{\max}(1-\sigma)}{\alpha_{\text{RORO}}}}{p_{\min}}, \frac{p_{\max} + 2\gamma}{p_{\min}} \right\}.$$

Algorithm 4 Doubling Version of RORO for OSDM *(only flexible demand & no delivery cost)*

initialize initial demand guess $\hat{d} \leftarrow 1$, index $j \leftarrow 0$, set of active RORO instances $\mathcal{R} \leftarrow \emptyset$
create initial instance of RORO $\mathcal{R} \leftarrow \mathcal{R} \cup \texttt{instance}_0(\hat{d})$ and associated demands $\mathcal{A}_0 \leftarrow \emptyset$
for each time step t **do**
  if new non-zero flexible demand $f_t > 0$ arrives **then**
    if $|\mathcal{A}_0| + f_t \le \hat{d}$ **then**
      Assign f_t to current instance of RORO: $\mathcal{A}_j \leftarrow f_t$
    else
      $\hat{d} \leftarrow 2^{j+1}, \quad j \leftarrow j+1$
      $\mathcal{R} \leftarrow \mathcal{R} \cup \texttt{instance}_j(\hat{d})$ and $\mathcal{A}_j \leftarrow f_t$
      Assign f_t to new instance of RORO: $\mathcal{A}_j \leftarrow f_t$
  compute initial purchase decision $x'_t \leftarrow \sum_{\texttt{instance}_j \in \mathcal{R}} x_t^{(j)}$
    ▷ $x_t^{(j)}$ is the purchase decision of $\texttt{instance}_j$
  compute initial delivery decision $z'_t \leftarrow \sum_{\texttt{instance}_j \in \mathcal{R}} x_t^{(j)}$
  if demand is binding, i.e., $\sum_{\tau=1}^{t-1} z_\tau + z_t < \sum_{\tau:\Delta_\tau \le t} f_\tau$ **then**
    increment $x_t \leftarrow x'_t + \left(\sum_{\tau:\Delta_\tau \le t} f_\tau - \sum_{\tau=1}^{t-1} z_\tau - z'_t\right)$, $z_t \leftarrow z'_t + \left(\sum_{\tau:\Delta_\tau \le t} f_\tau - \sum_{\tau=1}^{t-1} z_\tau - z'_t\right)$
  else
    $x_t \leftarrow x'_t, \quad z_t \leftarrow z'_t$

In the above, $\alpha_{\texttt{RORO}}$ is the optimal competitive ratio for OCS, which is defined in the context of OSDM as the unique positive solution to $\exp\left(\frac{1}{\alpha_{\texttt{RORO}}}\right) = \frac{p_{\max} - p_{\min} - 2\gamma}{p_{\max}/\alpha_{\texttt{RORO}} - p_{\max} - 2\gamma}$ [46].

Proof of Theorem 3.1. Consider the following *worst-case* OSDM instance $\mathcal{I}_K$ that satisfies the assumptions given in § 2.5 and § 3. Suppose that there are K instances of RORO created over the time horizon. We denote the j^{th} instance of RORO as $\texttt{instance}_j$, $j \in 0, 1, \ldots K$. Let $\mathcal{A}_j$ denote the set of flexible demands assigned to $\texttt{instance}_j$, and let $D^{(j)} = \sum_{f_t \in \mathcal{A}_j} f_t$ denote the total size of these demands.

We fix an arbitrary amount of total demand D, and a maximum size of each flexible demand $\sigma < 1$, and a small increment factor ι. An adaptive adversary presents RORO with a sequence of prices and flexible demands as follows. Whenever a new instance of RORO is created, the adversary presents a "round" of prices in iteratively descending order, starting from $p_{\max}$ and decreasing by ι at each time step. Whenever RORO purchases a non-zero amount of asset, the adversary's next presented price is $p_{\max}$, forcing RORO to "switch off" and incur a high switching cost.

Suppose the j^{th} instance of RORO has size $S^{(j)} = 2^j$. From the start of the j^{th} "round" of descending prices, the adversary begins presenting flexible demands with size smaller than or equal to σ and a uniform deadline of T, until the total size of flexible demands presented so far is equal to $S^{(j)} - \sigma + \upsilon$, for small $\upsilon > 0$. Note that at this point, the total size of flexible demands assigned to $\texttt{instance}_j$ is exactly $S^{(j)} - \sigma + \upsilon$. For sufficiently small ι, the adversary reaches this point before the descending prices reach $p_{\min}$—for the rest of the "round", the adversary presents zero new flexible demand.

By presenting prices in descending order (and for sufficiently small ι) the adversary ensures that RORO's purchasing is exactly captured by $\int_0^{S^{(j)}-\sigma+\upsilon} \phi^{(j)}(u)du$, where $\phi^{(j)}$ is the pseudo-cost function of the j^{th} instance of RORO. Furthermore, due to the price fluctuations, RORO will have incurred an extra switching cost (not accounted for by the integral) of at least $\gamma \cdot (S^{(j)} - \sigma + \upsilon)$, because the adversary forces RORO to switch off after each purchase.

Once $\texttt{instance}_j$ has purchased the total amount of demand assigned to it, the adversary continues to present descending prices until it reaches $p_{\min}$, presenting this best price up to m times (for

sufficiently large m). Note that RORO cannot purchase at these good prices, because it has already satisfied all of the flexible demand assigned to $\texttt{instance}_j$ and is thus constrained from doing so.

After presenting m copies of $p_{\min}$, the adversary then presents a new flexible demand with size σ, which is large enough to create a new instance of RORO. At this point, a new "round" of descending prices and adversarial demand begins, repeating the above process for $\texttt{instance}_{j+1}$, and so on. This continues until K instances of RORO have been created.

In what follows, we denote $D^{(j)} = S^{(j)} - \sigma + v$ as the total size of flexible demands assigned to $\texttt{instance}_j$ (i.e., the set $\mathcal{A}_j$). With a slight abuse of notation, let $\texttt{OPT}^{(j)}$ denote the cost incurred by OPT to satisfy the flexible demands assigned to $\texttt{instance}_j$. Since all flexible demands assigned to $\texttt{instance}_j$ arrive at or after time $t_-^{(j)}$, and must be satisfied before or at time T, the cost incurred by OPT to satisfy these demands is at least the best price that arrives during this time period (i.e., $p_{\min}$), times the total demand size $D^{(j)}$. We assume switching costs are negligible for sufficiently large m (see the instance description above). Thus, we have:

$$\texttt{OPT}^{(j)} = D^{(j)} p_{\min}. \tag{10}$$

Furthermore, the total cost of OPT on the OSDM instance $\mathcal{I}_K$ is exactly the sum of the costs incurred to satisfy each $\mathcal{A}_j$, i.e.,

$$\texttt{OPT}(\mathcal{I}_K) = \sum_{j=0}^{K} \texttt{OPT}^{(j)}.$$

In the original RORO competitive proof for OCS, they use the following facts to relate the cost of OPT and RORO for a single instance of RORO:

Lemma D.1 ([46, Lemma B.2]). *For a single instance of* RORO $\mathbf{I}$ *with unit-size demand, we have:*

$$\texttt{OPT}(\mathbf{I}) \geq \phi(\bar{w}) - \gamma,$$

where $\bar{w}$ is the total amount of demand satisfied by RORO *(excluding demand purchased due to binding constraints).*

Lemma D.2 ([46, Lemma B.3]). *For a single instance of* RORO $\mathbf{I}$ *with unit-size demand, we have:*

$$\texttt{RORO}(\mathbf{I}) \leq \int_0^{\bar{w}} \phi(u)du + \gamma\bar{w} + (1-\bar{w})p_{\max}.$$

To apply these bounds in our context, we partition the cost of doubling RORO into the sum of the costs incurred by each $\texttt{instance}_j$. With a slight abuse of notation, let $\texttt{RORO}^{(j)}$ denote the cost incurred by $\texttt{instance}_j$ to satisfy the flexible demands captured by $\mathcal{A}_j$. By the definition of the worst-case instance, we have:

$$\texttt{RORO}^{(j)} = \int_0^{D^{(j)}} \phi^{(j)}(u)du + \gamma D^{(j)}.$$

Lemma D.1 suggests that we should be able to lower bound $\texttt{OPT}^{(j)}$ by $D^{(j)}\left[\phi^{(j)}(D^{(j)}) - \gamma\right]$, where $D^{(j)}$ is the total amount of demand satisfied by $\texttt{instance}_j$. However, this is *not true* in OSDM—we already established that $\texttt{OPT}^{(j)} = D^{(j)}p_{\min}$, and the definition of the pseudo-cost function $\phi^{(j)}$ ensures that $\phi^{(j)}(S^{(j)}) - \gamma = p_{\min}$. However, since $D^{(j)} < S^{(j)}$ (because $D^{(j)} = S^{(j)} - \sigma + v$) and $\phi^{(j)}$ is decreasing, we have $\phi^{(j)}(D^{(j)}) - \gamma > p_{\min}$, so the bound does not hold. Thus, we must modify the lower bound on $\texttt{OPT}^{(j)}$ to account for the difference between $D^{(j)}$ and $S^{(j)}$.

Thus, we say that $\phi^{(j)}(\bar{w}^{(j)} + (S^{(j)} - D^{(j)})) - \gamma$ captures the best price that arrives during the lifetime of $\texttt{instance}_j$ if it satisfies $\bar{w}^{(j)}$ amount of demand. Since we have $\bar{w}^{(j)} = D^{(j)}$ in our

worst-case instance, we can use the following lower bound on $\mathtt{OPT}^{(j)}$:

$$\mathtt{OPT}^{(j)} = D^{(j)} \left[\phi^{(j)}(\bar{w}^{(j)} + (S^{(j)} - D^{(j)})) - \gamma\right]. \tag{11}$$

Note that this lower bound is tight in our worst-case instance, because the best price that arrives during the lifetime of $\mathtt{instance}_j$ is exactly $p_{\min} = \phi^{(j)}(S^{(j)}) - \gamma$, and $\bar{w}^{(j)} + (S^{(j)} - D^{(j)}) = S^{(j)}$.

In the case of unit-size demand, it is given that $\int_0^w \phi(u)du + \gamma w + (1-w)p_{\max} = \alpha_{\mathtt{RORO}}(\phi(w) - \gamma)$ for all $w \in [0,1]$ [46, Theorem 3.2]. Extending this to the case of instance j with size $S^{(j)}$, we have the following:

$$\int_0^{\bar{w}^{(j)}} \phi^{(j)}(u)du + \gamma\bar{w}^{(j)} + (S^{(j)} - \bar{w}^{(j)})p_{\max} = \alpha_{\mathtt{RORO}} S^{(j)}(\phi^{(j)}(\bar{w}^{(j)}) - \gamma),$$

$$\int_0^{\bar{w}^{(j)}} \phi^{(j)}(u)du + \gamma\bar{w}^{(j)} + (D^{(j)} - \bar{w}^{(j)})p_{\max} = \alpha_{\mathtt{RORO}} S^{(j)}(\phi^{(j)}(\bar{w}^{(j)}) - \gamma) - (S^{(j)} - D^{(j)})p_{\max}.$$

Since we know that $\mathtt{OPT}^{(j)}$ is lower bounded as in (11), we can relate $S^{(j)}(\phi^{(j)}(\bar{w}^{(j)})$ to $\mathtt{OPT}^{(j)}$ as follows:

$$S^{(j)}(\phi^{(j)}(\bar{w}^{(j)}) - \gamma) = \frac{S^{(j)}}{D^{(j)}} \cdot \left[\mathtt{OPT}^{(j)} + D^{(j)}\left(p_{\max} - 2\gamma - \frac{p_{\max}}{\alpha_{\mathtt{RORO}}}\right)\exp\left(\frac{S^{(j)} - D^{(j)}}{\alpha_{\mathtt{RORO}}}\right)\right],$$

where the trailing term comes from the additive $(S^{(j)} - D^{(j)})$ term inside the $\phi^{(j)}$ function in (11). Combining the above two equations, we have the following characterization of $\mathtt{RORO}^{(j)}$:

$$\mathtt{RORO}^{(j)} = \alpha_{\mathtt{RORO}} \frac{S^{(j)}}{D^{(j)}} \mathtt{OPT}^{(j)} + \alpha_{\mathtt{RORO}} S^{(j)}\left(p_{\max} - 2\gamma - \frac{p_{\max}}{\alpha_{\mathtt{RORO}}}\right)\exp\left(\frac{S^{(j)} - D^{(j)}}{\alpha_{\mathtt{RORO}}}\right) - (S^{(j)} - D^{(j)})p_{\max}. \tag{12}$$

Furthermore, note that the total cost of the doubling RORO instances on the OSDM instance $\mathcal{I}_K$ is exactly the sum of the costs incurred to satisfy each $\mathcal{A}_j$, i.e.,

$$\mathtt{RORO}(\mathcal{I}_K) = \sum_{j=0}^{K} \mathtt{RORO}^{(j)}.$$

In what follows, we let ζ denote the competitive ratio of the doubling RORO algorithm. We have the following $\frac{\mathtt{RORO}(\mathcal{I}_K)}{\mathtt{OPT}(\mathcal{I}_K)}$ for the given adversarial instance $\mathcal{I}_K$, where $D = \sum_{j=0}^{K} D^{(j)}$ is the total demand in the instance:

$$\begin{aligned}\frac{\mathtt{RORO}(\mathcal{I}_K)}{\mathtt{OPT}(\mathcal{I}_K)} &= \frac{\sum_{j=0}^{K} \mathtt{RORO}^{(j)}}{\sum_{j=0}^{K} \mathtt{OPT}^{(j)}}, \\ &= \sum_{j=0}^{K} \frac{D^{(j)}}{D} \frac{\mathtt{RORO}^{(j)}}{\mathtt{OPT}^{(j)}}\end{aligned}$$

We can use the fact that $D^{(j)} = S^{(j)} - \sigma$ to simplify the expression of $\mathtt{RORO}^{(j)}/\mathtt{OPT}^{(j)}$ using the following facts: $S^{(j)} - D^{(j)} = \sigma$, $\frac{S^{(j)}}{D^{(j)}} = \frac{S^{(j)}}{S^{(j)} - \sigma}$, $\mathtt{OPT}^{(j)} \geq D^{(j)} p_{\min}$. Thus, we have:

$$\frac{\mathtt{RORO}^{(j)}}{\mathtt{OPT}^{(j)}} = \frac{S^{(j)}\alpha_{\mathtt{RORO}}}{S^{(j)} - \sigma} + \frac{S^{(j)}\alpha_{\mathtt{RORO}}}{S^{(j)} - \sigma} \cdot \frac{\left(p_{\max} - 2\gamma - \frac{p_{\max}}{\alpha_{\mathtt{RORO}}}\right)\exp\left(\frac{\sigma}{\alpha_{\mathtt{RORO}}}\right) - \frac{\sigma p_{\max}(S^{(j)} - \sigma)}{S^{(j)}\alpha_{\mathtt{RORO}}}}{p_{\min}}.$$

Note that the term $\frac{S^{(j)}}{S^{(j)} - \sigma}$ is maximized when $S^{(j)}$ is closest to σ, and thus the worst-case ratio occurs when $j = 0$ and $S^{(0)} = 1$. This corresponds to the case where $K = 0$ (i.e., only the initial

instance of RORO is created). In this instance, we have:

$$\frac{\mathsf{RORO}(\mathcal{I}_0)}{\mathsf{OPT}(\mathcal{I}_0)} = \frac{D^{(0)}}{D}\frac{\mathsf{RORO}^{(0)}}{\mathsf{OPT}^{(0)}},$$

where $D^{(0)} = D$ and $\frac{\mathsf{RORO}^{(0)}}{\mathsf{OPT}^{(0)}}$ is given by the following:

$$\frac{\mathsf{RORO}(\mathcal{I}_0)}{\mathsf{OPT}(\mathcal{I}_0)} = \frac{\mathsf{RORO}^{(0)}}{\mathsf{OPT}^{(0)}} = \left(\frac{\alpha_{\mathsf{RORO}}}{1-\sigma} + \frac{\alpha_{\mathsf{RORO}}}{1-\sigma}\cdot\frac{\left(p_{\max} - 2\gamma - \frac{p_{\max}}{\alpha_{\mathsf{RORO}}}\right)\exp\left(\frac{\sigma}{\alpha_{\mathsf{RORO}}}\right) - \frac{\sigma p_{\max}(1-\sigma)}{\alpha_{\mathsf{RORO}}}}{p_{\min}}\right).$$

Since the competitive ratio ζ is defined as the supremum over all valid OSDM instances which includes $\mathcal{I}_0$, we have the following lower bound on ζ:

$$\zeta \geq \frac{\alpha_{\mathsf{RORO}}}{1-\sigma} + \frac{\alpha_{\mathsf{RORO}}}{1-\sigma}\cdot\frac{\left(p_{\max} - 2\gamma - \frac{p_{\max}}{\alpha_{\mathsf{RORO}}}\right)\exp\left(\frac{\sigma}{\alpha_{\mathsf{RORO}}}\right) - \frac{\sigma p_{\max}(1-\sigma)}{\alpha_{\mathsf{RORO}}}}{p_{\min}}.$$

We note that in a regime where σ approaches 1, the above lower bound on ζ is ill-defined. In the OCS problem, it is known that the worst-possible competitive ratio is $\frac{p_{\max}+2\gamma}{p_{\min}}$ [46], which is achieved by any feasible algorithm that purchases at any time step (i.e., possibly incurring the worst cost $p_{\max}$), and incurs the worst-possible switching cost to do so (i.e., 2γ).

In the OSDM problem with no delivery costs (i.e., $c = \varepsilon = 0$), no storage, and only flexible demands, we have a similar result—any algorithm is $\frac{p_{\max}+2\gamma}{p_{\min}}$-competitive as long as it is feasible. Thus, we have the following breakpoint for the lower bound on ζ:

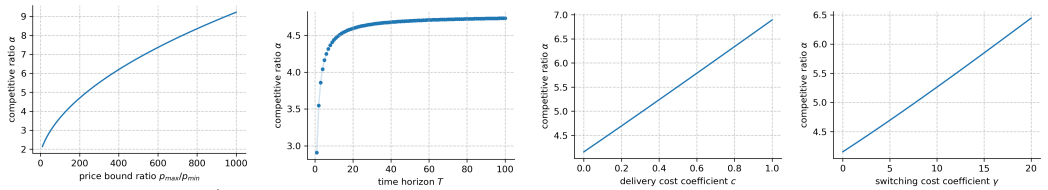
$$\zeta \geq \min\left\{\frac{\alpha_{\mathsf{RORO}}}{1-\sigma} + \frac{\alpha_{\mathsf{RORO}}}{1-\sigma}\cdot\frac{\left(p_{\max} - 2\gamma - \frac{p_{\max}}{\alpha_{\mathsf{RORO}}}\right)\exp\left(\frac{\sigma}{\alpha_{\mathsf{RORO}}}\right) - \frac{\sigma p_{\max}(1-\sigma)}{\alpha_{\mathsf{RORO}}}}{p_{\min}}, \frac{p_{\max}+2\gamma}{p_{\min}}\right\}.$$

This completes the proof.

□

E Deferred Proofs from § 4 (Analysis of PAAD Algorithm)

In this section, we provide full proofs for the results in § 4, which describes and analyzes the PAAD algorithm for the OSDM problem. We start by providing deferred plots of the competitive ratio α as a function of the problem's parameters to showcase the important dependencies in Fig. 12. We then prove Theorem 4.3, which states the competitive bound of PAAD for OSDM-S, before proving Theorem 4.4, which states the corresponding bound for OSDM-T.



(a) Price bound ratio $\frac{p_{\min}}{p_{\max}}$ (b) Time horizon T (c) Delivery coeff. c (d) Switching coeff. γ

Fig. 12. Plotting the competitive ratio α (defined in (3)) as a function of the parameter dependencies described in each subfigure. Unless a parameter is on an x-axis, we set $p_{\min} = 10$, $p_{\max} = 200$, $\gamma = 5$, $\delta = 1$, $c = 0.2$, $\varepsilon = 0.05$, and $T = 48$.

E.1 Proof of Theorem 4.3

In this section, we prove Theorem 4.3, which states that PAAD achieves a competitive ratio of α (defined in (3)) for OSDM-S. We start by proving feasibility of the solution produced by PAAD, before proceeding to prove the competitive ratio.

Proof. Before the main competitive proof, we start our analysis by showing that PAAD produces a feasible solution to OSDM.

Lemma E.1. PAAD *produces a feasible solution to* OSDM*, i.e., it satisfies all demand before their deadlines and never violates the storage capacity constraint.*

Proof. We prove feasibility directly by the definition of the algorithm—there are two primary constraints to check.

First, we must ensure that all demand is covered. In the case of base demand, this is immediate by line 18 of Algorithm 2, where the delivery decision z_t always includes the base demand b_t that arrives at time t. In the case of flexible demand, this is ensured by line 19 and line 23 of Algorithm 2, which checks for binding constraints and projects the purchase and delivery decisions (if necessary) to ensure that all flexible demands are satisfied before their deadlines.

Next, we show that the storage capacity constraint is never violated. We first consider the base demand, as the base demand drivers have the "role" of replenishing the storage. Note that by Definition E.2, the storage is empty ($s_t = 0$) during any inactive period, which trivially satisfies the storage capacity constraint $0 \le s_t \le S$. We thus then consider an arbitrary time step t during an active period. Again by Definition E.2, the total amount of demand purchased by base drivers in this i^{th} active period is less than or equal to $\sum_{j=0}^{\nu_i} \hat{w}_b^{(i,j)}$—this itself is less than or equal to $\sum_{j=0}^{\nu_i} B_b^{(i,j)}$ by the definition of the base demand driver. The cumulative base demand that arrives in the i^{th} active period is equal to $\sum_{j=1}^{\nu_i} B_b^{(i,j)}$ (note the changed index to account for the initial storage manager driver $j = 0$). Thus, the amount stored is at most $\sum_{j=0}^{\nu_i} \hat{w}_b^{(i,j)} - \sum_{j=1}^{\nu_i} B_b^{(i,j)}$, which is at most $B_b^{(i,0)} = S$ by the definition of the storage manager driver. Thus, we have $s_t \le S$ for all t during active periods.

We now consider the flexible demand. At a high level, the structure of PAAD is such that flexible demand drivers do not rely on the storage to satisfy demand—rather, they rely on the temporal flexibility of flexible demand to reduce cost subject to constraints. To see this, first note that when a flexible demand's deadline is binding, it is assumed in § 2.5 to always be feasible to increment *both* z_t and x_t by an amount f_t to ensure that the demand is satisfied by its deadline. In such a case, the storage dynamics leave the storage unchanged, i.e., $s_t = s_{t-1} + f_t - f_t = s_{t-1}$. Thus, the storage state satisfies $s_t \le S$ for all t during flexible demand deadlines. In other time steps, the effect of the flexible demand drivers on the storage is governed by their per-time-step delivery decisions—i.e., if purchasing exceeds delivery, the storage increases, and if delivery increases purchasing, the storage is depleted. However, note that the delivery decisions of flexible demand drivers are always upper bounded by the amount of demand that has been purchased so far, i.e., $z_t^{(i)} \le w_t^{(i)}$ for all $i \in \mathcal{F}$ and $t \in [T]$. Furthermore, due to the analytical structure of the purchasing and delivery threshold functions in the context of a monotone price-dependent delivery cost, the (cumulative) delivery decisions of a flexible demand driver at a given time step are always at least as large as its (cumulative) purchasing decisions, i.e., $v_t^{(i)} \ge w_t^{(i)}$ for all $i \in \mathcal{F}$ and $t \in [T]$. Thus, the net effect of flexible demand drivers on the storage is 0, i.e., $\sum_{i\in\mathcal{F}} x_t^{(i)} = \sum_{i\in\mathcal{F}} z_t^{(i)}$ for all $t \in [T]$, and the storage state satisfies $s_t \le S$ for all t during non-binding flexible demand periods as well.

□

We now proceed to prove the competitive ratio of PAAD. To do so, we introduce some notation and a definition to facilitate our analysis. For notational brevity, the following considers *any arbitrary* $\mathcal{I} \in \Omega$ (i.e., any arbitrary instance of OSDM).

Definition E.2 (Active and inactive periods). In our proof, we partition the time horizon T into two types of **periods**. An **active period** contains a contiguous interval where the storage is non-empty (i.e., $s_t > 0$). An **inactive period** corresponds to any interval that lies in-between two adjacent active periods.

We use the following additional notations. Suppose that for an arbitrary instance, PAAD's actions result in n active periods. During the i^{th} active period, $i \in [n]$, we denote ν_i as the number of base demand drivers that are created. Further, we let $\hat{w}_b^{(i,j)}$ denote the amount of the j^{th} base demand driver that has been purchased by the end of the i^{th} active period, $j \in [\nu_i]$, and let $B_b^{(i,j)}$ denote the total demand associated with the j^{th} base demand driver in the i^{th} active period.

We index flexible demand drivers according to the time at which they were created, $\tau \in [T]$. In particular, we let $\hat{w}_f^{(\tau)}$ and $\hat{v}_f^{(\tau)}$ denote the purchasing and delivery amounts of the τ^{th} flexible demand driver before the deadline. Note that $f_\tau \geq 0$ is the total demand associated with the τ^{th} flexible demand driver, and if $f_\tau = 0$, then $\hat{w}_f^{(\tau)} = \hat{v}_f^{(\tau)} = 0$.

We let $D_b = \sum_{i=1}^{n} \sum_{j=1}^{\nu_i} B_b^{(i,j)}$ and $D_f = \sum_{\tau=1}^{T} f_\tau$ denote the total base and flexible demand, respectively, that arrive over the entire time horizon. We let $D = D_b + D_f$ denote the total demand. We let $\hat{s}$ denote the final status of the storage at the end of the time horizon, i.e., $\hat{s} = s_T$.

Finally, we introduce the following notation to characterize the optimal solution: let $G_i(\beta)$ denote the minimum cost of purchasing β units of asset during the i^{th} active period, and let $H_\tau(f)$ denote the minimum cost of purchasing f units of asset during the period $[\tau, \tau+\Delta_\tau]$ (i.e., during the lifetime of the τ^{th} flexible demand). Finally, let $\tilde{p}$ denote the minimum price during idle periods, and let $\bar{p}$ denote the weighted average price during periods with non-zero base demand, i.e., $\bar{p} = \frac{\sum_{\tau \in [T]} p_t \cdot b_t}{D_b}$.

With these preliminaries, we are ready to proceed to the main competitive proof. We begin by proving a lower bound on the cost of the offline optimal solution OPT.

Lemma E.3. *Given that* PAAD *produces n active periods, let β_i denote the asset purchased towards the base demand by the offline optimal solution during the i^{th} active period, $i \in [n]$, and let $\tilde{p}$ denote the minimum price during inactive periods. Then* $\texttt{OPT}(\mathcal{I})$ *is lower bounded as:*

$$\texttt{OPT}(\mathcal{I}) \geq \sum_{i=1}^{n} G_i(\beta_i) + \left(D_b - \sum_{i=1}^{n} \beta_i\right) \tilde{p} + \varepsilon \bar{p} D_b + \sum_{\tau=1}^{T} (1+\varepsilon)\, H_\tau(f_\tau) + \left(D_b + D_f\right) \frac{2\kappa}{T}$$

Proof. The cost of an optimal offline solution, denoted as $\texttt{OPT}(\mathcal{I})$, can be split into five components. The first two components are the purchasing cost and delivery cost associated with the base demand, respectively. The third and fourth components are the purchasing and delivery costs associated with the flexible demand, respectively. The last component is the switching cost incurred by the optimal solution.

Recall that we let $G_i(\beta)$ denote the minimum cost of purchasing β units of asset during the i^{th} active period, and let β_i denote the amount of asset purchased towards the base demand during the i^{th} active period by the optimal solution. Then the total purchasing cost associated with the base demand is lower bounded by $\sum_{i=1}^{n} G_i(\beta_i) + \left(D_b - \sum_{i=1}^{n} \beta_i\right) \tilde{p}$, where $\tilde{p}$ is the *minimum* price during inactive periods. The delivery cost associated with the base demand is lower bounded by $\varepsilon \bar{p} D_b$, where $\bar{p}$ is the weighted average price during periods with non-zero base demand—note that we use $\bar{p}$ in this case because the base demand must be delivered at the time of its arrival.

Next, recall that we let $H_\tau(f)$ denote the minimum cost of purchasing f units of asset during the period $[\tau, \tau+\Delta_\tau]$ (i.e., during the lifetime of the τ^{th} flexible demand). Then the total purchasing cost associated with the flexible demand is lower bounded by $\sum_{\tau=1}^{T} H_\tau(f_\tau)$. The delivery cost associated with the flexible demand is lower bounded by $\varepsilon \sum_{\tau=1}^{T} H_\tau(f_\tau)$, since the flexible demand can be delivered at any time before its deadline, and the smallest cost to do so is at the best (i.e., smallest) price during its lifetime.

Finally, note that the switching cost incurred by the optimal solution is at least $\left(D_b + D_f\right)\frac{2\kappa}{T}$, where $\kappa = \gamma + \delta$. This follows because the total demand must be purchased and delivered over the entire time horizon, and the switching cost is minimized when the switching cost is evenly distributed over the entire time horizon. This completes the proof. □

After Lemma E.3, we now proceed to prove an upper bound on the cost incurred by PAAD.

Lemma E.4. *Given that* PAAD *produces n active periods, let* $\hat{w}_b^{(i,j)}$ *denote the amount of the* j^{th} *base demand driver that has been purchased by the end of the* i^{th} *active period,* $j \in [v_i]$*, and let* $B_b^{(i,j)}$ *denote the total demand associated with the* j^{th} *base demand driver in the* i^{th} *active period. Further, let* $\hat{w}_f^{(\tau)}$ *and* $\hat{v}_f^{(\tau)}$ *denote the purchasing and delivery amounts of the* τ^{th} *flexible demand driver before the deadline, respectively. Then the cost incurred by* PAAD *is upper bounded as:*

$$\text{PAAD}(\mathcal{I}) \le \sum_{i=1}^{n}\sum_{j=1}^{v_i}\int_0^{\hat{w}_b^{(i,j)}} \phi_b^{(i,j)}(u)du + \sum_{\tau=1}^{T}\left(\int_0^{\hat{w}_f^{(\tau)}} \phi_f^{(\tau)}(u)du + \int_0^{\hat{v}_f^{(\tau)}} \psi^{(\tau)}(z)dz\right) \tag{13}$$

$$+ \left(D_b + D_f - \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} - \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}\right)(p_{\max} + 2\gamma) \tag{14}$$

$$+ D_b\left(\bar{p}(c+\varepsilon) + 2\delta\right) + \left(D_f - \sum_{\tau=1}^{T}\hat{v}_f^{(\tau)}\right)(p_{\max}(c+\varepsilon) + 2\delta) \tag{15}$$

$$- cp_{\min}\left(\sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} + \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}\right) + \hat{s}p_{\max}. \tag{16}$$

Proof. This proof follows by explicitly characterizing the worst-case cost of PAAD using the definition in Algorithm 2. In the equation above, (13) corresponds to the worst-case cost that can be charged to each driver's purchasing and delivery threshold functions, (14) corresponds to the worst-case cost incurred by PAAD for purchasing any remaining demand that has not been purchased by the drivers, (15) corresponds to the worst-case cost incurred by PAAD for delivering demand (in the flexible demand case, this corresponds to any demand that has not been delivered thus far), and (16) corresponds to the worst-case *improvement* in delivery costs due to increases in the storage state, plus the final storage level. We unpack each below.

First, note that by the definition of each driver, PAAD only purchases (or delivers) when the market price p_t is sufficiently low to make the threshold minimization problem in lines 13 and 17 negative (see Algorithm 2). For instance, we can say that the purchasing and switching cost of the j^{th} base demand driver in the i^{th} active period is upper bounded by $\int_0^{\hat{w}_b^{(i,j)}} \phi_b^{(i,j)}(u)du$. Similar logic follows for the purchasing and delivery costs of the flexible demand drivers. Aggregating over all drivers gives (13). We remark that the definition of the pseudo-cost minimization problem (Line 3, Algorithm 1) charges the current procurement cost ($p_t x$), the current switching cost $\gamma|x - x_{t-1}|$,

and the *future* (worst-case) switching cost $\gamma|x|$ to the threshold function.[6] This is feasible because the threshold function includes enough "buffer" to accommodate switching costs by definition, and captures any costs incurred before e.g., constraints become binding.

Any additional purchasing needed to satisfy all demands is exactly captured by the difference between $D_b + D_f$ (the total base and flexible demand) and the cumulative purchasing by the drivers $\sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} + \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}$. In the worst case, this additional purchasing is done at the highest price $p_{\max}$, and incurs a switching cost of 2γ (i.e., a switch up and a switch down). This gives (14).

Next, the delivery cost is upper bounded as follows. For the base demands that must be delivered immediately, the worst-case delivery cost is $\bar{p}(c+\varepsilon) + 2\delta$ per unit, where $\bar{p}$ is the weighted average price during periods with non-zero base demand. Note that $c + \varepsilon$ is the worst-case for the delivery coefficient (i.e., the largest cost PAAD can pay as a function of the storage state), and assumes that a maximal switching cost of 2δ is incurred. For the flexible demands, the worst-case delivery cost is incurred for any remaining undelivered demand that is not already covered by the flexible demand driver—this is upper bounded by $p_{\max}(c+\varepsilon) + 2\delta$ per unit. Note that the same worst-case assumptions are made as above, except that in the flexible demand case, this undelivered demand is delivered at the worst-case (highest) price $p_{\max}$. This gives (15).

Finally, (16) credits PAAD for any improvements in delivery costs that occur due to increases in the storage state. In particular, recall that in this setting of OSDM, the delivery cost coefficient decreases as a function of the storage state, so any purchasing that feeds the storage will reduce future delivery costs. In the worst-case however, this improvement is realized when it has the smallest possible impact on the delivery cost, i.e., when the overall price is $c_{\min}$. Finally, the term $\hat{s}p_{\max}$ captures any excess left in the storage at the end of the time horizon, which was purchased at a price of at most $p_{\max}$. This gives (16), and completes the proof.

□

In addition to the above lemmas, the following technical lemmas are necessary to prove a relation between the threshold functions and the optimal cost. We restate Lemma 4.5 and Lemma 4.6 as follows:

By the definition of the threshold function $\phi_b^{(i,j)}(\cdot)$, the following relation always holds:

$$\int_0^w \phi_b(u)du + (1-w)(p_{\max}+2\gamma) + p_{\max}(c+\varepsilon) + 2\delta - cwp_{\min} = \alpha\left[\phi_b(w) - 2\gamma + \varepsilon p_{\max} + \frac{2\kappa}{T}\right] \forall w \in [0,1].$$

By the definitions of the threshold functions $\phi_f(\cdot)$ and $\psi_f(\cdot)$, the following relation always holds:

$$\int_0^w \phi_f(u)du + (1-w)(p_{\max}+2\gamma) - cwp_{\min} + \int_0^v \psi_f(z)dz + (1-v)(p_{\max}(c+\varepsilon)+2\delta) = \alpha'\left[\phi_f(w) + \psi_f(v) - 2\kappa + \frac{2\kappa(1+c+\varepsilon)}{T(1+\varepsilon)}\right] \forall w \in [0,1], v \in [0,w].$$

Using the previous lemmas, we now prove the competitive ratio. First, in the simple case where $D_b + D_f = 0$, we have that the optimal cost is 0 and the cost of PAAD is at most $\hat{s}p_{\max} \le Sp_{\max}$, so PAAD is α-competitive under Definition 2.1 by setting $C = Sp_{\max}$. We thus focus on the more general case in which $D_b + D_f > 0$. Using the results in Lemmas E.3, E.4, 4.5, and 4.6, we claim that the following holds:

$$\frac{\mathsf{PAAD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})} \le \alpha.$$

[6]The future switching cost captures the fact that the final global purchasing/delivery decisions x_{T+1} and z_{T+1} are both 0 by construction, and the pseudo-cost minimization captures this future cost in advance.

To show this result, we first substitute the bounds from Lemmas E.3 and E.4 into the left-hand side of the above equation. We define some shorthand notation to facilitate the presentation.

Let $Q = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\int_0^{\hat{w}_b^{(i,j)}}\phi_b^{(i,j)}(u)du + \sum_{\tau=1}^{T}\left(\int_0^{\hat{w}_f^{(\tau)}}\phi_f^{(\tau)}(u)du + \int_0^{\hat{v}_f^{(\tau)}}\psi^{(\tau)}(z)dz\right)$ denote the integrals over the thresholds. Let $\hat{W}_b = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)}$, let $\hat{W}_f = \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}$, and let $\hat{W} = \hat{W}_b + \hat{W}_f$ denote the total purchasing by all drivers.

Further, let $\hat{V}_f = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{v}_f^{(\tau)}$ denote the total delivery by all flexible demand drivers.

Let $\boldsymbol{\beta} = \sum_{i=1}^{n}\beta_i$ denote the total amount of asset purchased towards the base demand by the optimal solution during all active periods, noting that $D_b - \boldsymbol{\beta} \geq 0$ by definition. Finally, let $D = D_b + D_f$ denote the total demand. Substituting the bounds from Lemmas E.3 and E.4 into the left-hand side of the above equation, we have:

$$\frac{\mathsf{PAAD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})} \leq \frac{Q + (D-\hat{W})(p_{\max}+2\gamma) + D_b\,(\bar{p}(c+\varepsilon)+2\delta) + (D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta) - cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i) + (D_b-\boldsymbol{\beta})\tilde{p} + \varepsilon\bar{p}D_b + \sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau) + D\frac{2\kappa}{T}}$$

Using the fact that $D_b - \boldsymbol{\beta} \geq 0$, the following is equivalent:

$$= \frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\gamma)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta) + (D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}+(D_b-\boldsymbol{\beta})(p_{\max}+2\kappa+\bar{p}(c+\varepsilon))}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\kappa}{T}+(D_b-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\kappa}{T})}$$

Then, we have the following:

$$\leq \max\left\{\frac{Q + (\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\gamma) + \boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta) + (D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta) - cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i) + \varepsilon\bar{p}\boldsymbol{\beta} + \sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau) + (D_f+\boldsymbol{\beta})\frac{2\kappa}{T}},\right.$$
$$\left.\frac{(D_b-\boldsymbol{\beta})(p_{\max}+2\kappa+\bar{p}(c+\varepsilon))}{(D_b-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\kappa}{T})}\right\},$$

where the definition of $\tilde{p}$ ensures that the second term in the max is at most α. We now focus on the first term. For the sake of contradiction, suppose that

$$\frac{Q + (\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\gamma) + \boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta) + (D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta) - cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i) + \varepsilon\bar{p}\boldsymbol{\beta} + \sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau) + (D_f+\boldsymbol{\beta})\frac{2\kappa}{T}} > \alpha. \tag{17}$$

Instead of working directly with the expression in terms of $\boldsymbol{\beta}$, we first reason about how the cost of OPT and PAAD relate to one another in terms of $\sum_{i=1}^{n}\sum_{j=1}^{v_i}B_b^{(i,j)}$, the total demand assigned to base drivers. We introduce the following notation for the sake of brevity: let $D_b = \sum_{i=1}^{n}\sum_{j=1}^{v_i}B_b^{(i,j)}$ denote the total base demand. Then, we have the following relation:

$$\frac{Q + (D_b+D_f-\hat{W})(p_{\max}+2\gamma) + D_b\,(\bar{p}(c+\varepsilon)+2\delta) + (D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta) - cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(B_i) + \varepsilon\bar{p}D_b + \sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau) + (D_f+D_b)\frac{2\kappa}{T}},$$
$$\geq \frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\gamma)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}+(D_b-\boldsymbol{\beta})(p_{\max}+\bar{p}(c+\varepsilon)+2\kappa)}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\kappa}{T}+(D_b-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\kappa}{T})},$$
$$> \alpha.$$

where in the second inequality, we have used the fact that there is a worst-case input instance such that for all $0 < \beta < \beta' < \sum_{j=1}^{v_i}B_b^{(i,j)} - \hat{b}^{(i)}$, we have $G_i(\beta') - G_i(\beta) \leq \tilde{p}(\beta'-\beta)$, where $\hat{b}^{(i)}$ is

the initial state of the storage at the start of the i^{th} active period in OPT's solution. See Lemma E.5 for a formal proof of this.

Lemma E.5. *Defining $\hat{b}^{(i)}$ as the initial state of the storage at the start of the i^{th} active period in* OPT*'s solution, there exists a worst-case input instance such that for all $0 < \beta < \beta' < \sum_{j=1}^{v_i} B_b^{(i,j)} - \hat{b}^{(i)}$, we have:*

(1) $G_i(0) = 0$,
(2) $G_i(\beta') > G_i(\beta)$,
(3) $G_i(\beta') - G_i(\beta) \leq \tilde{p}(\beta' - \beta)$.

Proof. Statements 1 and 2 are immediate from the definition of $G_i(\cdot)$. Recall that $\tilde{p}$ is defined as the minimum price during inactive periods, which corresponds with the worst (highest) price that PAAD's base drivers are willing to pay to fill the storage.

To prove statement 3, consider any input instance $\mathcal{I} = [(p_t, b_t, f_t, \Delta_t)]_{t\in[T]}$. An adversary can construct a new input instance $\mathcal{I}' = [(p'_t, b'_t, f'_t, \Delta'_t)]_{t\in[T']}$ as follows:

$$\mathcal{I}' \coloneqq [(\tilde{p}, 0, 0, \cdot), (p_1, b_1, f_1, \Delta_1), (\tilde{p}, 0, 0, \cdot), (p_2, b_2, f_2, \Delta_2), \ldots, (p_T, b_T, f_T, \Delta_T)].$$

The purpose of constructing this new input instance is to guarantee that the purchasing cost of PAAD does not change, and the purchasing cost of OPT will not increase (for this worst-case logic, we only consider the purchasing cost, ignoring additional switching costs that would increase OPT's cost). Note that this is possible because the adversary can set the length of the time horizon. Let $G_i(\beta)$ be the minimum cost of purchasing β units of asset during the i^{th} active period. When OPT purchases another $\beta' - \beta, \beta' \leq \sum_{j=1}^{v_i} B_b^{(i,j)} - \hat{b}^{(i)}$ units of asset during the i^{th} active period, the cost will not be larger than $\tilde{p}(\beta' - \beta)$, since OPT can always purchase at any newly added time slots. Thus, there always exists a worst-case input instance such that $G_i(\beta') - G_i(\beta) \leq \tilde{p}(\beta' - \beta)$.

□

Given the result in Lemma E.5, we proceed to work with the expression in terms of D_b. During the i^{th} active period and the lifetime of the j^{th} base demand driver, the minimum marginal purchasing price observed is given by $\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma$ by the definition of the threshold function. Similarly, during the lifetime of the τ^{th} flexible demand driver, the minimum marginal purchasing and delivery cost observed is given by $\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2\kappa\right)$.
This gives the following lower bounds on the terms that depend on G_i and H_τ, respectively:

$$\sum_{i=1}^{n} G_i\left(\sum_{j=1}^{v_i} B_b^{(i,j)}\right) \geq \sum_{i=1}^{n}\sum_{j=1}^{v_i}\left(\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma\right) \times B_b^{(i,j)}$$

$$\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau) \geq \sum_{\tau=1}^{T}\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2\kappa\right) \times f_\tau.$$

Substituting these bounds into the previous expression, we have that the left-hand-side of (17) is less than or equal to:

$$\leq \frac{Q + (D_b + D_f - \hat{W})(p_{\max} + 2\gamma) + D_b\left(\bar{p}(c+\varepsilon) + 2\delta\right) + (D_f - \hat{V}_f)(p_{\max}(c+\varepsilon) + 2\delta) - cp_{\min}\hat{W}}{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left(\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma\right)B_b^{(i,j)} + \varepsilon\bar{p}D_b + \sum_{\tau=1}^{T}\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2\kappa\right)f_\tau + (D_f + D_b)\frac{2\kappa}{T}}.$$

By rearranging the terms in the above and substituting for Q, we obtain the following:

$$= \frac{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left[X_b^{(i,j)}\right] + \sum_{\tau=1}^{T}\left[X_f^{(\tau)}\right]}{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left[\left(\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma + \varepsilon\bar{p} + \frac{2\kappa}{T}\right)B_b^{(i,j)}\right] + \sum_{\tau=1}^{T}\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2\kappa\right)f_\tau + f_\tau\frac{2\kappa}{T}\right]}, \tag{18}$$

where $X_b^{(i,j)}$ and $X_f^{(\tau)}$ are defined as follows (for each driver):

$$X_b^{(i,j)} = \int_0^{\hat{w}_b^{(i,j)}} \phi_b^{(i,j)}(u)du + (B_b^{(i,j)} - \hat{w}_b^{(i,j)})(p_{\max} + 2\gamma) + (p_{\max}(c+\varepsilon) + 2\delta)\, B_b^{(i,j)} - c\hat{w}_b^{(i,j)}p_{\min},$$

$$X_f^{(\tau)} = \int_0^{\hat{w}_f^{(\tau)}} \phi_f^{(\tau)}(u)du + (f_\tau - \hat{w}_f^{(\tau)})(p_{\max} + 2\gamma) + \int_0^{\hat{v}_f^{(\tau)}} \psi_f^{(\tau)}(u)du + (p_{\max}(c+\varepsilon) + 2\delta)\,(f_\tau - \hat{v}_f^{(\tau)}) - c\hat{w}_f^{(\tau)}p_{\min}.$$

Since (18) is $> \alpha$ by assumption, one of the following cases must be true:

Case I: There exists some $i \in [n]$ and $j \in [v_i]$ such that

$$X_b^{(i,j)} > \alpha B_b^{(i,j)}\left(\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma + \varepsilon\bar{p} + \frac{2\kappa}{T}\right).$$

Case II: There exists some $\tau \in [T]$ such that

$$X_f^{(\tau)} > \alpha' f_\tau\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2\kappa\right) + \frac{2\kappa}{T}\right],$$

But, if either of these two cases are true (i.e., for any unit of base or flexible demand), they contradict Lemmas 4.5 and 4.6, respectively. Thus, we have a contradiction, and the original assumption that (18) is $> \alpha$ must be false. This completes the proof that $\frac{\mathsf{PAAD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})} \leq \alpha$.
Using this result, we have:

$$\mathsf{PAAD}(\mathcal{I}) \leq \alpha\mathsf{OPT}(\mathcal{I}) + p_{\max}\hat{s} \leq \alpha\mathsf{OPT}(\mathcal{I}) + p_{\max}S,$$

where $p_{\max}S$ is a constant. This shows that PAAD is α-competitive under Definition 2.1, which completes the proof.

□

E.2 Proof of Corollary 4.4

In this section, we prove Corollary 4.4, which states that under the assumptions in § 2.5, PAAD is α_T-competitive for OSDM-T, where α_T is given by:

$$\alpha_\mathsf{T} \geq \frac{\omega}{\left[W\left(-\frac{((1+c+\varepsilon)p_{\max} - (1+\varepsilon)p_{\min})e^{\frac{\omega 2\delta/T - p_{\min}c - (1+c+\varepsilon)p_{\max}}{(1+c+\varepsilon)p_{\max}+2(\eta+\delta)}}}{(1+c+\varepsilon)p_{\max}+2(\eta+\delta)}\right) + \frac{(1+c+\varepsilon)p_{\max}+p_{\min}c-\omega 2\delta/T}{(1+c+\varepsilon)p_{\max}+2(\eta+\delta)}\right]}, \tag{19}$$

where $\omega = \frac{(1+c+\varepsilon)}{(1+\varepsilon)}$. We start with a (re-)definition of PAAD's threshold functions and pseudo-cost minimization problem for the OSDM-T setting.

Definition E.6 (Threshold functions and pseudo-cost minimization for PAAD *in* OSDM-T*). Given a base driver with size denoted by d, the purchasing threshold function ϕ_b for* OSDM-T *is defined as:*

$$\phi_b(w) = p_{\max} + 2\eta + p_{\min}c + \left(\frac{p_{\max}(1+c+\varepsilon) + 2(\eta+\delta)}{\alpha_\mathsf{T}} - \left(p_{\max}(1+\varepsilon) + p_{\min}c + \frac{2\delta}{T}\right)\right)\exp\left(\frac{w}{\alpha_\mathsf{T}d}\right) : w \in [0,d].$$

Furthermore, given a flexible driver with size denoted by d, the purchasing threshold function ϕ_f and delivery threshold function ψ_f for OSDM-T *are defined as:*

$$\phi_f(w) = p_{\max} + p_{\min} c + 2\eta + \left(\frac{p_{\max} + 2\eta}{\alpha'_{\mathsf{T}}} - (p_{\max} + p_{\min} c)\right) \exp\left(\frac{w}{\alpha'_{\mathsf{T}} d}\right) : w \in [0, d],$$

$$\psi(v) = p_{\max}(c+\varepsilon) + 2\delta + \left(\frac{p_{\max}(c+\varepsilon) + 2\delta}{\alpha'_{\mathsf{T}}} - \left(p_{\max}(c+\varepsilon) + \frac{2\delta}{T}\omega\right)\right) \exp\left(\frac{v}{\alpha'_{\mathsf{T}} d}\right) : v \in [0, d],$$

where $\omega = \frac{(1+c+\varepsilon)}{(1+\varepsilon)}$ *and* $\alpha'_{\mathsf{T}} = \alpha_{\mathsf{T}}/\omega$.

Since OSDM-T *includes a tracking cost on the purchasing side (i.e., rather than a switching cost), we modify the pseudo-cost minimization problem solved by the base demand driver and the flexible demand driver (just on the purchasing side) accordingly. In Algorithm 1, we replace the input of a switching coeff. γ with inputs of a tracking coeff. η and target a_t in line 1. We replace line 2 with:*

$$\textbf{\textit{define}}\ \textit{pseudo-target}\ \hat{a}_t^{(i)} \leftarrow \frac{a_t \cdot d^{(i)}}{\sum_{i \in \mathcal{B} \cup \mathcal{F}} d^{(i)}}$$

and replace line 3 with:

$$\textbf{\textit{solve}}\ x_t^{(i)} \leftarrow \textit{argmin}_{x \in [0, d^{(i)} - w_{t-1}^{(i)}]}\ p_t x + \eta |x - \hat{a}_t^{(i)}| - \Phi^{(i)}(w_{t-1}^{(i)}, w_{t-1}^{(i)} + x)$$

Proof. The result largely follows by the proof of Theorem 4.3, with modifications to account for the tracking cost instead of the switching cost on the grid side. Thus, to show the result, we state several lemmas that replace the corresponding logic in the proof of Theorem 4.3. We start by stating a lower bound on the cost of the offline optimal solution OPT for OSDM-T, analogous to Lemma E.3.

Lemma E.7. *Given that* PAAD *produces n active periods, let β_i denote the asset purchased towards the base demand by the offline optimal solution during the i^{th} active period, $i \in [n]$, and let $\tilde{p}$ denote the minimum price during inactive periods. Then* $\mathtt{OPT}(\mathcal{I})$ *is lower bounded as:*

$$\mathtt{OPT}(\mathcal{I}) \geq \sum_{i=1}^{n} G_i(\beta_i) + \left(D_b - \sum_{i=1}^{n} \beta_i\right) \tilde{p} + \varepsilon \bar{p} D_b + \sum_{\tau=1}^{T} (1+\varepsilon)\, H_\tau(f_\tau) + \left(D_b + D_f\right) \frac{2\delta}{T}$$

Proof. The proof follows by the same logic as Lemma E.3, except instead of a switching cost on the grid side (i.e., with coefficient γ), there is a tracking cost with coefficient η. The switching cost on the demand side remains the same (with coefficient δ). The worst case for PAAD is when OPT pays zero tracking cost, yielding that the total tracking & switching cost incurred by the optimal solution is at least $\left(D_b + D_f\right) \frac{2\delta}{T}$. □

Lemma E.8. *Given that* PAAD *produces n active periods, let $\hat{w}_b^{(i,j)}$ denote the amount of the j^{th} base demand driver that has been purchased by the end of the i^{th} active period, $j \in [v_i]$, and let $B_b^{(i,j)}$ denote the total demand associated with the j^{th} base demand driver in the i^{th} active period. Further, let $\hat{w}_f^{(\tau)}$ and $\hat{v}_f^{(\tau)}$ denote the purchasing and delivery amounts of the τ^{th} flexible demand driver before the*

deadline, respectively. Then the cost incurred by PAAD *is upper bounded as:*

$$\mathsf{PAAD}(\mathcal{I}) \le \sum_{i=1}^{n}\sum_{j=1}^{v_i}\int_0^{\hat{w}_b^{(i,j)}}\phi_b^{(i,j)}(u)du + \sum_{\tau=1}^{T}\left(\int_0^{\hat{w}_f^{(\tau)}}\phi_f^{(\tau)}(u)du + \int_0^{\hat{v}_f^{(\tau)}}\psi^{(\tau)}(z)dz\right) \tag{20}$$

$$+\left(D_b + D_f - \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} - \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}\right)(p_{\max} + 2\eta) \tag{21}$$

$$+ D_b\left(\bar{p}(c+\varepsilon) + 2\delta\right) + \left(D_f - \sum_{\tau=1}^{T}\hat{v}_f^{(\tau)}\right)(p_{\max}(c+\varepsilon) + 2\delta) \tag{22}$$

$$- cp_{\min}\left(\sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} + \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}\right) + \hat{s}p_{\max}. \tag{23}$$

Proof. The proof follows by the same logic as Lemma E.4—the main change is with respect to (21), to capture the worst-case purchasing cost incurred by PAAD for purchasing the remaining demand that has not been purchased by the drivers.

First, note that when the threshold functions ϕ_b and ϕ_f are defined according to Corollary 4.4, the pseudo-cost terms in (20) capture the worst-case *tracking* cost incurred by PAAD due to deviations from the signal wherever demand *has* been purchased by the drivers.

Finally, any additional purchasing needed to satisfy demand is exactly captured by the difference between $D_b + D_f$ (the total base and flexible demand) and the cumulative purchasing by the drivers $\sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} + \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}$. In the worst case, this additional purchasing is done at the highest price $p_{\max}$, and incurs a tracking cost of 2η (capturing the case where purchasing is simultaneously not done when the tracking target is non-zero and is done when the tracking target is zero). This gives (21), and completes the proof. □

Due to the redefinition of the threshold functions ϕ_b, ϕ_f and ψ_f in Corollary 4.4, we have the following technical lemmas to relate the threshold functions and the optimal cost:

Lemma E.9. *By the (re-)definition of the threshold function $\phi_b^{(i,j)}(\cdot)$ in Corollary 4.4, the following relation always holds:*

$$\int_0^w \phi_b(u)du + (1-w)(p_{\max}+2\eta) + p_{\max}(c+\varepsilon) + 2\delta - cwp_{\min} = \alpha_{\mathsf{T}}\left[\phi_b(w) - 2\eta + \varepsilon p_{\max} + \frac{2\delta}{T}\right] \forall w \in [0,1].$$

Lemma E.10. *By the (re-)definitions of the threshold functions $\phi_f^{(\tau)}(\cdot)$ and $\psi^{(\tau)}(\cdot)$ in Corollary 4.4, the following relation always holds:*

$$\int_0^w \phi_f(u)du + (1-w)(p_{\max}+2\eta) - cwp_{\min} + \int_0^v \psi_f(z)dz + (1-v)(p_{\max}(c+\varepsilon)+2\delta) =$$
$$\alpha'_{\mathsf{T}}\left[\phi_f(w) + \psi_f(v) - 2(\eta+\delta) + \frac{2\delta(1+c+\varepsilon)}{T(1+\varepsilon)}\right] \forall w \in [0,1], v \in [0,w],$$

Using the results in Lemmas E.7, E.8, E.9, and E.10, we claim that the following holds:

$$\frac{\mathsf{PAAD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})} \le \alpha_{\mathsf{T}}.$$

To show this result, we first substitute the bounds from Lemmas E.7 and E.8 into the left-hand side of the above equation. As in the proof of Theorem 4.3, we define some shorthand notation to facilitate the presentation.

Let $Q = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\int_0^{\hat{w}_b^{(i,j)}}\phi_b^{(i,j)}(u)du + \sum_{\tau=1}^{T}\left(\int_0^{\hat{w}_f^{(\tau)}}\phi_f^{(\tau)}(u)du + \int_0^{\hat{v}_f^{(\tau)}}\psi^{(\tau)}(z)dz\right)$ denote the integrals over the thresholds. Let $\hat{W}_b = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)}$, let $\hat{W}_f = \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}$, and let $\hat{W} = \hat{W}_b + \hat{W}_f$ denote the total purchasing by all drivers.
Further, let $\hat{V}_b = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{v}_b^{(i,j)}$ denote the total delivery by all flexible demand drivers.
Let $\boldsymbol{\beta} = \sum_{i=1}^{n}\beta_i$ denote the total amount of asset purchased towards the base demand by the optimal solution during all active periods, noting that $D_b - \boldsymbol{\beta} \geq 0$ by definition. Finally, let $D = D_b + D_f$ denote the total demand. Substituting the bounds from Lemmas E.7 and E.8 into the left-hand side of the above inequality, we have:

$$\frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}+(D_b-\boldsymbol{\beta})(p_{\max}+2(\eta+\delta)+\bar{p}(c+\varepsilon))}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\delta}{T}+(D_b-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\delta}{T})}$$

Then, we have the following:

$$\leq \max\left\{\frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\delta}{T}},\right.$$
$$\left.\frac{(D_b-\boldsymbol{\beta})(p_{\max}+2(\eta+\delta)+\bar{p}(c+\varepsilon))}{(D_b-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\delta}{T})}\right\},$$

where the definition of $\tilde{p}$ ensures that the second term in the max is at most α_{T}. We now focus on the first term. For the sake of contradiction, suppose that

$$\frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\delta}{T}} > \alpha_{\mathsf{T}}. \tag{24}$$

Instead of working directly with the expression in terms of $\boldsymbol{\beta}$, we first reason about how the cost of OPT and PAAD relate to one another in terms of $\sum_{i=1}^{n}\sum_{j=1}^{v_i}B_b^{(i,j)}$, the total demand assigned to base drivers. We introduce the following notation for the sake of brevity: let $\boldsymbol{B} = \sum_{i=1}^{n}\sum_{j=1}^{v_i}B_b^{(i,j)}$ denote the total base demand. Then, we have the following relation:

$$\frac{Q+(\boldsymbol{B}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{B}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(B_i)+\varepsilon\bar{p}\boldsymbol{B}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{B})\frac{2\delta}{T}},$$
$$\geq \frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}+(\boldsymbol{B}-\boldsymbol{\beta})(p_{\max}+\bar{p}(c+\varepsilon)+2\kappa)}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\delta}{T}+(\boldsymbol{B}-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\delta}{T})},$$
$$> \alpha_{\mathsf{T}}.$$

where in the second inequality, we have used Lemma E.5. We proceed to work with the expression in terms of $\boldsymbol{B}$. During the i^{th} active period and the lifetime of the j^{th} base demand driver, the minimum marginal purchasing price observed is given by $\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\eta$ by the definition of the threshold function. Similarly, during the lifetime of the τ^{th} flexible demand driver, the minimum marginal purchasing and delivery cost observed is given by $\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\eta+\delta)\right)$.

This gives the following lower bounds on the terms that depend on G_i and H_τ, respectively:

$$\sum_{i=1}^{n} G_i\left(\sum_{j=1}^{v_i} B_b^{(i,j)}\right) \geq \sum_{i=1}^{n}\sum_{j=1}^{v_i}\left(\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\eta\right) \times B_b^{(i,j)}$$

$$\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau) \geq \sum_{\tau=1}^{T}\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\eta+\delta)\right) \times f_\tau.$$

Substituting these bounds into the previous expression, we have that the left-hand-side of (24) is less than or equal to:

$$\leq \frac{Q + (\boldsymbol{B} + D_f - \hat{W})(p_{\max} + 2\eta) + \boldsymbol{B}\,(\bar{p}(c+\varepsilon) + 2\delta) + (D_f - \hat{V}_f)(p_{\max}(c+\varepsilon) + 2\delta) - cp_{\min}\hat{W}}{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left(\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\eta\right)B_b^{(i,j)} + \varepsilon\bar{p}\boldsymbol{B} + \sum_{\tau=1}^{T}\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\eta+\delta)\right)f_\tau + (D_f + \boldsymbol{B})\frac{2\delta}{T}}.$$

By rearranging the terms in the above and substituting for Q, we obtain the following:

$$= \frac{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left[X_b^{(i,j)}\right] + \sum_{\tau=1}^{T}\left[X_f^{(\tau)}\right]}{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left[\left(\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\eta + \varepsilon\bar{p} + \frac{2\delta}{T}\right)B_b^{(i,j)}\right] + \sum_{\tau=1}^{T}\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\eta+\delta)\right)f_\tau + f_\tau\frac{2\delta}{T}\right]}, \tag{25}$$

where $X_b^{(i,j)}$ and $X_f^{(\tau)}$ are defined as follows (for each driver):

$$X_b^{(i,j)} = \int_0^{\hat{w}_b^{(i,j)}}\phi_b^{(i,j)}(u)du + (B_b^{(i,j)} - \hat{w}_b^{(i,j)})(p_{\max} + 2\eta) + (p_{\max}(c+\varepsilon) + 2\delta)\,B_b^{(i,j)} - c\hat{w}_b^{(i,j)}p_{\min},$$

$$X_f^{(\tau)} = \int_0^{\hat{w}_f^{(\tau)}}\phi_f^{(\tau)}(u)du + (f_\tau - \hat{w}_f^{(\tau)})(p_{\max} + 2\eta) + \int_0^{\hat{v}_f^{(\tau)}}\psi_f^{(\tau)}(u)du + (p_{\max}(c+\varepsilon) + 2\delta)\,(f_\tau - \hat{v}_f^{(\tau)}) - c\hat{w}_f^{(\tau)}p_{\min}.$$

Since (25) is $> \alpha_{\mathsf{T}}$ by assumption, one of the following cases must be true:

Case I: There exists some $i \in [n]$ and $j \in [v_i]$ such that

$$X_b^{(i,j)} > \alpha_{\mathsf{T}}B_b^{(i,j)}\left(\phi_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\eta + \varepsilon\bar{p} + \frac{2\delta}{T}\right).$$

Case II: There exists some $\tau \in [T]$ such that

$$X_f^{(\tau)} > \alpha'_{\mathsf{T}}f_\tau\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\phi_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \psi_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\eta+\delta)\right) + \frac{2\delta}{T}\right],$$

But, if either of these two cases are true (i.e., for any unit of base or flexible demand), they contradict Lemmas E.9 and E.10, respectively. Thus, we have a contradiction, and the original assumption that (25) is $> \alpha_{\mathsf{T}}$ must be false. This completes the proof that $\frac{\mathsf{PAAD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})} \leq \alpha_{\mathsf{T}}$.
Using this result, we have:

$$\mathsf{PAAD}(\mathcal{I}) \leq \alpha_{\mathsf{T}}\mathsf{OPT}(\mathcal{I}) + p_{\max}\hat{s} \leq \alpha_{\mathsf{T}}\mathsf{OPT}(\mathcal{I}) + p_{\max}S,$$

where $p_{\max}S$ is a constant. This shows that PAAD is α_{T}-competitive under Definition 2.1, which completes the proof. □

F Deferred Proofs from § 5

In this section, we provide full proofs for the results in § 5, which establishes lower bounds on the competitive ratio achievable by deterministic online algorithms for several instantiations of OSDM. We start by proving Theorem 5.1, which shows a lower bound on the best competitive ratio achievable by any deterministic online algorithm for OSDM-S. Then, we prove Corollary 5.2, which shows a lower bound on the best competitive ratio achievable for OSDM-S with only base demand,

before proving Theorem 5.3, which shows that the decreasing delivery cost case is strictly harder than the increasing delivery cost case within the class of monotone delivery costs defined by Def. 2.4. Finally, we prove Theorem 5.4, which shows a lower bound on the best competitive ratio achievable by any deterministic online algorithm for OSDM-T.

F.1 Proof of Theorem 5.1

In this section, we prove Theorem 5.1, which states that no deterministic online algorithm can achieve a competitive ratio better than α (defined in (3)) for OSDM-S. We start by defining a family of difficult instances for OSDM-S, before proceeding to prove the lower bound on the competitive ratio achievable by any deterministic online algorithm.

Proof. To show this result, we first define a family of instances, and then show that the competitive ratio of any deterministic algorithm is lower bounded under these instances. Consider the following set of instances $\{\mathcal{I}_x\}_{x\in[p_{\min},p_{\max}]}$, where $\mathcal{I}_x$ is called an *x-decreasing instance*.

Definition F.1 (x-decreasing instance for OSDM-S*).* Let $n, m \in \mathbb{N}$ be sufficiently large, and denote $\varrho \coloneqq {p_{\max}-p_{\min}}/{n}$.

For $x \in [p_{\min}, p_{\max}]$, we define a decreasing market price sequence as follows: The sequence is partitioned into $n_x \coloneqq 2 \cdot \lceil {(x-p_{\min})}/{\varrho} \rceil + 1$ alternating batches of prices. The i^{th} batch ($i \in [n_x - 2]$) contains m prices $p_{\max}$ if i is odd, and 1 cost function with coefficient $p_{\max} - (\lceil i/2 \rceil)\varrho$ if i is even. The last two batches consist of m cost functions with coefficient $x + \iota$, followed by a final batch of 1 cost function with coefficient $p_{\max}$.

The demand sequence is defined as follows: At $t = 1$, a batch of flexible demand $f_1 = 1$ arrives with deadline $\Delta_1 = T$. All other b_t and f_t are zero.

Let $h(x)$ and $z(x)$ denote *conversion functions* that both map $[p_{\min}, p_{\max}] \to [0, 1]$, respectively. Suppose these arbitrary functions fully describe the actions of a deterministic ALG for OSDM on an instance $\mathcal{I}_x$. Specifically, suppose that $h(x)$ describes ALG's purchasing decisions, and $z(x)$ describes ALG's delivery decisions *before* the arrival of the last batch of prices ($p_{\max}$) in the instance. Note that by definition, $z(x) \le h(x)$, since the delivery cannot exceed what has been purchased thus far. Note that for large n, processing an instance $\mathcal{I}_{x-\varrho}$ is equivalent to first processing $\mathcal{I}_x$ (besides the last two batches), and then processing batches with prices $x - \varrho$ and $p_{\max}$.

Since ALG is deterministic and both conversions (i.e., purchasing and delivery) are both unidirectional (irrevocable), we must have that $h(x - \varrho) \ge h(x)$ and $z(x - \varrho) \ge z(x)$, i.e., $h(x)$ and $z(x)$ are both non-increasing in $[p_{\min}, p_{\max}]$. Intuitively, the entire demand should be purchased before the end of the sequence if the lowest price appears, i.e., $h(p_{\min}) = 1$. Furthermore, since the lowest price appears at least twice,

ALG pays switching cost proportional to $h(x)$ and $z(x)$, because each good price is "interrupted" by bad prices in the $\mathcal{I}_x$ instance, captured by $2\gamma h(x)$ and $2\delta z(x)$, respectively. However, if ALG is forced to purchase at the deadline T, it also pays a switching cost of $2\gamma(1 - h(x))$ and $2\delta(1 - z(x))$, respectively. Thus, the total worst-case switching cost paid by ALG is $2\gamma + 2\delta$.

Recall that in the price-dependent decreasing delivery cost case, the worst case delivery coefficient is $c + \varepsilon \le 1$, when the storage is empty. We bound the total *effective* delivery cost coefficient (i.e., across all time steps) as follows:

$$\sum_{t\in[T]} (c - cs_t + \varepsilon) = c + \varepsilon - \sum_{t\in[T]} cs_t \le c + \varepsilon - ch(x),$$

where we have used the fact that for any ALG, $\sum_{t\in[T]} s_t \le h(x)$. Since the delivery cost is price-dependent, the worst-case for ALG occurs when the *savings* in the delivery cost are realized at the

lowest price, i.e., $p_{\min} ch(x)$. Then the total cost of ALG on instance $\mathcal{I}_x$ is described by:

$$\begin{aligned}\mathtt{ALG}(\mathcal{I}_x) = h(\ell)\ell - \int_\ell^x u dh(u) + (1-h(x))p_{\max} + 2\gamma - p_{\min} ch(x) + \\ (c+\varepsilon)\ell z(\ell) - (c+\varepsilon)\int_\ell^x u dz(u) + (1-z(x))(c+\varepsilon)p_{\max} + 2\delta.\end{aligned}$$

The optimal cost on instance $\mathcal{I}_x$ (if all prices are known in advance) is given by:

$$\mathtt{OPT}(\mathcal{I}_x) = x + \frac{2\gamma}{T} + \varepsilon x + \frac{2\delta}{T}.$$

If ALG is $\alpha^\star$-competitive, then a necessary condition is that for all $x \in [p_{\min}, p_{\max}]$, $\mathtt{ALG}(\mathcal{I}_x) \le \alpha^\star \mathtt{OPT}(\mathcal{I}_x)$. This imposes a necessary condition on $h(x)$ and $z(x)$, which can be expressed as the following differential inequality:

$$\begin{aligned}h(\ell)\ell - \int_\ell^x u dh(u) + (1-h(x))p_{\max} + 2\gamma - p_{\min} ch(x) + \\ (c+\varepsilon)\ell z(\ell) - (c+\varepsilon)\int_\ell^x u dz(u) + (1-z(x))(c+\varepsilon)p_{\max} + 2\delta \\ \le \alpha^\star\left(x + \frac{2\gamma}{T} + \varepsilon x + \frac{2\delta}{T}\right).\end{aligned}$$

Recall that $z(x) \le h(x)$. On the instances described above, under the assumption that $\kappa = \gamma + \delta \le \frac{p_{\max} - p_{\min}}{2}$ (see § 2.5), note that for sufficiently good x (i.e., $x < \ell$) it is strictly better for ALG to deliver as much as possible, i.e., $z(x) = h(x)$, reducing its last-minute delivery cost. Thus, we proceed by assuming that $z(x) = h(x)$. Using integration by parts and letting $\kappa = \gamma + \delta$, this gives the following condition on $h(x)$:

$$\begin{aligned}h(x)x - \int_\ell^x h(u)du + (1-h(x))p_{\max} + 2\kappa - p_{\min} ch(x) + \\ (c+\varepsilon)xh(x) - (c+\varepsilon)\int_\ell^x h(u)du + (1-h(x))(c+\varepsilon)p_{\max} \\ \le \alpha^\star\left(x + \frac{2\kappa}{T} + \varepsilon x\right).\end{aligned}$$

To solve for $\alpha^\star$, we can use Grönwall's inequality [60, Theorem 1, p. 356]. First, observe that $h(x)$ must satisfy the following:

$$h(x) \ge \frac{p_{\max} + \frac{2\kappa}{(1+c+\varepsilon)} - \alpha^\star \frac{x + \frac{2\kappa}{T} + \varepsilon x}{1+c+\varepsilon}}{p_{\max} - x + \frac{p_{\min} c}{1+c+\varepsilon}} - \frac{1}{p_{\max} - x + \frac{p_{\min} c}{1+c+\varepsilon}} \int_\ell^x h(u)du.$$

By Grönwall's inequality [60, Theorem 1, p. 356], it follows that:

$$\begin{aligned}h(x) &\ge \frac{p_{\max} + \frac{2\kappa}{(1+c+\varepsilon)} - \alpha^\star \frac{x + \frac{2\kappa}{T} + \varepsilon x}{1+c+\varepsilon}}{p_{\max} - x + \frac{p_{\min} c}{1+c+\varepsilon}} - \int_\ell^x \frac{p_{\max} + \frac{2\kappa}{(1+c+\varepsilon)} - \alpha^\star \frac{u + \frac{2\kappa}{T} + \varepsilon u}{1+c+\varepsilon}}{\left(p_{\max} - u + \frac{p_{\min} c}{1+c+\varepsilon}\right)^2} du, \\ h(x) &\ge \alpha^\star \frac{1+\varepsilon}{1+c+\varepsilon} \ln\left[\frac{(1+c+\varepsilon)(p_{\max} - x) + p_{\min} c}{(1+c+\varepsilon)(p_{\max} - \ell) + p_{\min} c}\right].\end{aligned}$$

Recall that we have a boundary condition that $h(p_{\min}) = 1$—we can combine this with the above to obtain:

$$1 \geq \alpha^\star \frac{1+\varepsilon}{1+c+\varepsilon} \ln\left[\frac{(1+c+\varepsilon)(p_{\max}-p_{\min})+p_{\min}c}{(1+c+\varepsilon)(p_{\max}-\ell)+p_{\min}c}\right].$$

Letting $\ell = \frac{(1+c+\varepsilon)p_{\max}+2\kappa}{\alpha^\star(1+\varepsilon)} - \frac{2\kappa}{T(1+\varepsilon)}$, the optimal $\alpha^\star$ is obtained when the above inequality is binding, so we have:

$$1 = \alpha^\star \frac{1+\varepsilon}{1+c+\varepsilon} \ln\left[\frac{(1+c+\varepsilon)p_{\max}-(1+\varepsilon)p_{\min}}{(1+c+\varepsilon)p_{\max} - \frac{(1+c+\varepsilon)[(1+c+\varepsilon)p_{\max}+2\kappa]}{\alpha^\star(1+\varepsilon)} + \frac{(1+c+\varepsilon)2\kappa}{T(1+\varepsilon)} + p_{\min}c}\right].$$

Solving the above yields that the optimal $\alpha^\star$ for any ALG solving OSDM is lower bounded by:

$$\alpha^\star \geq \frac{\omega}{\left[W\left(-\frac{((1+c+\varepsilon)p_{\max}-(1+\varepsilon)p_{\min})e^{\frac{-\omega 2\kappa/T - p_{\min}c-(1+c+\varepsilon)p_{\max}}{(1+c+\varepsilon)p_{\max}+2\kappa}}}{(1+c+\varepsilon)p_{\max}+2\kappa}\right) + \frac{(1+c+\varepsilon)p_{\max}+p_{\min}c-\omega 2\kappa/T}{(1+c+\varepsilon)p_{\max}+2\kappa}\right]},$$

where $\kappa = \gamma + \delta$ and $\omega = \frac{(1+c+\varepsilon)}{(1+\varepsilon)}$. This completes the proof.

□

F.2 Proof of Corollary 5.2

In this section, we prove Corollary 5.2, which states that no deterministic online algorithm can achieve a competitive ratio better than α_{B} (defined in (5)) for OSDM-S instances that only contain base demand. To do so, we define a tweaked set of difficult instances for OSDM-S with only base demand, and use a similar conversion function argument to bound the best competitive ratio.

Proof. To show the result, we start by defining a set of instances similar to those defined in Def. F.1, where the instance contains only base demand.

Definition F.2 (x-decreasing instance for OSDM-S *with just base demand).* Recall the price sequence defined in Def. F.1. We define a similar set of instances $\{\mathcal{I}'_x\}_{x\in[p_{\min},p_{\max}]}$, where $\mathcal{I}'_x$ is called an *x-decreasing instance with just base demand.*

The demand sequence is defined as follows: At $t = T$, a batch of base demand $b_T = 1$ arrives. All other b_t and f_t are zero.

We now show that the competitive ratio of any deterministic online algorithm ALG on the above instances is lower bounded by α_{B} defined in (5). We assume without loss of generality that ALG knows the value of $b_T = 1$ in advance, since this can only help ALG.

Let $h(x)$ denote a *conversion function* that maps $[p_{\min}, p_{\max}] \to [0, 1]$—this describes ALG's purchasing decisions before the final batch of prices ($p_{\max}$) in the instance. Since ALG is deterministic and purchasing is unidirectional (irrevocable), we must have that $h(x-\varrho) \geq h(x)$, i.e., $h(x)$ is non-increasing in $[p_{\min}, p_{\max}]$. Intuitively, the entire demand should be purchased before the end of the sequence if the lowest price appears, i.e., $h(p_{\min}) = 1$.

ALG pays switching cost proportional to $h(x)$, because each good price is "interrupted" by bad prices in the $\mathcal{I}_x$ instance, captured by $2\gamma h(x)$. If ALG is forced to purchase when the demand arrives, it also pays a switching cost of $2\gamma(1-h(x))$. Thus, the total worst-case switching cost on the purchasing side paid by ALG is 2γ.

The switching cost on the delivery side for both ALG and OPT is 2δ, since the entire base demand arrives at the end of the sequence, and it must be delivered at that time.

Recall that in the price-dependent decreasing delivery cost case, the worst case delivery coefficient is $c + \varepsilon \leq 1$, when the storage is empty. We bound the total *effective* delivery cost coefficient (i.e., across all time steps) as follows:

$$\sum_{t\in[T]} (c - cs_t + \varepsilon) = c + \varepsilon - \sum_{t\in[T]} cs_t \leq c + \varepsilon - ch(x),$$

where we have used the fact that for any ALG, $\sum_{t\in[T]} s_t \leq h(x)$. Since the delivery cost is price-dependent, the *savings* in the delivery cost are realized at the price at the final time step, i.e., $p_{\max}ch(x)$. Then the total cost of ALG on instance $\mathcal{I}'_x$ is described by:

$$\texttt{ALG}(\mathcal{I}'_x) = h(\ell)\ell - \int_\ell^x udh(u) + (1 - h(x))p_{\max} + 2\gamma - p_{\max}ch(x) + (c + \varepsilon)p_{\max} + 2\delta.$$

The optimal cost on instance $\mathcal{I}'_x$ (if all prices are known in advance) is given by:

$$\texttt{OPT}(\mathcal{I}'_x) \geq x + \frac{2\gamma}{T} + \varepsilon p_{\max} + 2\delta.$$

If ALG is $\alpha^\star$-competitive, then a necessary condition is that for all $x \in [p_{\min}, p_{\max}]$, $\texttt{ALG}(\mathcal{I}_x) \leq \alpha^\star\texttt{OPT}(\mathcal{I}_x)$. This imposes a necessary condition on $h(x)$, which can be expressed as the following differential inequality:

$$h(\ell)\ell - \int_\ell^x udh(u) + (1 - h(x))p_{\max} + 2\gamma - p_{\max}ch(x) + (c + \varepsilon)p_{\max} + 2\delta \leq \alpha^\star\left(x + \frac{2\gamma}{T} + \varepsilon p_{\max} + 2\delta\right).$$

Using integration by parts and letting $\kappa = \gamma + \delta$, this gives the following condition on $h(x)$:

$$h(x) \geq \frac{p_{\max}(1 + c + \varepsilon) + 2\kappa - \alpha^\star\left(x + \frac{2\gamma}{T} + \varepsilon p_{\max} + 2\delta\right)}{p_{\max} + p_{\max}c - x} - \frac{1}{p_{\max} + p_{\max}c - x}\int_\ell^x h(u)du.$$

By Grönwall's inequality [60, Theorem 1, p. 356], it follows that:

$$h(x) \geq \frac{p_{\max}(1+c+\varepsilon) + 2\kappa - \alpha^\star\left(x + \frac{2\gamma}{T} + \varepsilon p_{\max} + 2\delta\right)}{p_{\max} + p_{\max}c - x} - \int_\ell^x \frac{p_{\max}(1+c+\varepsilon) + 2\kappa - \alpha^\star\left(u + \frac{2\gamma}{T} + \varepsilon p_{\max} + 2\delta\right)}{(p_{\max} + p_{\max}c - u)^2}du,$$

$$h(x) \geq \alpha^\star \ln\left[\frac{p_{\max} + p_{\max}c - x}{p_{\max} + p_{\max}c - \ell}\right],$$

where $\ell = \frac{p_{\max}(1+c+\varepsilon)+2\kappa}{\alpha^\star} - \frac{2\gamma}{T} - p_{\max}\varepsilon - 2\delta$.
Recall that we have a boundary condition that $h(p_{\min}) = 1$—we can combine this with the above to obtain:

$$1 \geq \alpha^\star \ln\left[\frac{p_{\max} + p_{\max}c - p_{\min}}{p_{\max} + p_{\max}c - \ell}\right].$$

Letting $\ell = \frac{(1+c+\varepsilon)p_{\max}+2\kappa}{\alpha^\star} - \frac{2\kappa}{T} - p_{\max}\varepsilon - 2\delta$, the optimal $\alpha^\star$ is obtained when the above inequality is binding, so we have:

$$1 = \alpha^\star \ln\left[\frac{(1 + c)p_{\max} - p_{\min}}{(1 + c)p_{\max} - \frac{(1+c+\varepsilon)p_{\max}+2\kappa}{\alpha^\star} + \frac{2\gamma}{T} + p_{\max}\varepsilon + 2\delta}\right].$$

Solving the above yields that the optimal $\alpha^\star$ for any ALG solving OSDM-S with just base demand is lower bounded by:

$$\alpha^\star \geq \left[W\left(-\frac{((1+c)p_{\max} - p_{\min})\, e^{-\frac{(1+c+\varepsilon)p_{\max}+2\delta+\frac{2\gamma}{T}}{(1+c+\varepsilon)p_{\max}+2\kappa}}}{(1+c+\varepsilon)p_{\max}+2\kappa} \right) + \frac{(1+c+\varepsilon)p_{\max}+2\delta+\frac{2\gamma}{T}}{(1+c+\varepsilon)p_{\max}+2\kappa} \right]^{-1},$$

which completes the proof. □

F.3 Proof of Theorem 5.3

In this section, we prove Theorem 5.3, which states that amongst the class of monotone delivery costs defined in Def. 2.4, the best achievable competitive ratios for OSDM-S with an *increasing delivery cost* are strictly better than those for OSDM-S with a decreasing delivery cost (i.e., the cases considered in Theorem 5.1 and Corollary 5.2)

Proof. To show this result, we prove two lower bounds (one in the case of just base demand, and one in the case of just flexible demand) to compare against the lower bounds in Theorem 5.1 and Corollary 5.2, respectively. We start with the case of flexible demand and the following lemma:

Lemma F.3. *There exists a set of* OSDM-S *instances with only flexible demand and a monotone increasing delivery cost such that no deterministic online algorithm* ALG *can achieve a competitive ratio better than* $\alpha_{\texttt{IF}}$*, given by:*

$$\alpha_{\texttt{IF}} \geq \left[W\left(-\frac{(p_{\max} - p_{\min})\, e^{-1}}{p_{\max} + \frac{2\kappa}{T(1+\varepsilon)}} \right) + 1 \right]^{-1}. \tag{26}$$

Proof. We recall Def. F.1 and consider the same set of instances $\{\mathcal{I}_x\}_{x\in[p_{\min},p_{\max}]}$, where $\mathcal{I}_x$ is called an *x-decreasing instance*.

Let $h(x)$ and $z(x)$ denote *conversion functions* that both map $[p_{\min}, p_{\max}] \to [0,1]$, respectively. Suppose these arbitrary functions fully describe the actions of a deterministic ALG for OSDM with a monotone increasing delivery cost on an instance $\mathcal{I}_x$. Specifically, $h(x)$ describes ALG's purchasing decisions and $z(x)$ describes ALG's delivery decisions *before* the arrival of the last batch of prices ($p_{\max}$) in the instance. Note that by definition, $z(x) \leq h(x)$, since the delivery cannot exceed what has been purchased thus far.

Since ALG is deterministic and both conversions (i.e., purchasing and delivery) are both unidirectional (irrevocable), we must have that $h(x-\varrho) \geq h(x)$ and $z(x-\varrho) \geq z(x)$, i.e., $h(x)$ and $z(x)$ are both non-increasing in $[p_{\min}, p_{\max}]$. Intuitively, the entire demand should be purchased before the end of the sequence if the lowest price appears, i.e., $h(p_{\min}) = 1$. Furthermore, since the lowest price appears at least twice, ALG pays switching cost proportional to $h(x)$ and $z(x)$, because each good price is "interrupted" by bad prices in the $\mathcal{I}_x$ instance, captured by $2\gamma h(x)$ and $2\delta z(x)$, respectively. However, if ALG is forced to purchase at the deadline T, it also pays a switching cost of $2\gamma(1-h(x))$ and $2\delta(1-z(x))$, respectively. Thus, the total worst-case switching cost paid by ALG is $2\gamma + 2\delta$.

Recall that in the price-dependent increasing delivery cost case, the worst case delivery coefficient is $c + \varepsilon \leq 1$, when the storage is fully charged. Thus, in this case, ALG's optimal decision is to deliver the demand as early as possible (i.e., $h(x) = z(x) \forall x \in [p_{\min}, p_{\max}]$), reducing its delivery cost. Under this assumption, ALG's effective delivery coefficient is ε. Then the total cost of ALG on instance $\mathcal{I}_x$ is described by:

$$\texttt{ALG}(\mathcal{I}_x) = h(\ell)\ell - \int_\ell^x u dh(u) + (1-h(x))p_{\max} + 2\gamma + \varepsilon\ell z(\ell) - \varepsilon\int_\ell^x u dz(u) + (1-z(x))\varepsilon p_{\max} + 2\delta.$$

And the optimal cost on instance $\mathcal{I}_x$ (if all prices are known in advance) is given by:

$$\texttt{OPT}(\mathcal{I}_x) = x + \frac{2\gamma}{T} + \varepsilon x + \frac{2\delta}{T}.$$

If ALG is $\alpha^\star$-competitive, then a necessary condition is that for all $x \in [p_{\min}, p_{\max}]$, $\texttt{ALG}(\mathcal{I}_x) \leq \alpha^\star \texttt{OPT}(\mathcal{I}_x)$. This imposes a necessary condition on $h(x)$ and $z(x)$, which can be expressed as the following differential inequality:

$$h(\ell)\ell - \int_\ell^x u dh(u) + (1-h(x))p_{\max} + 2\gamma + \varepsilon\ell z(\ell) - \varepsilon\int_\ell^x u dz(u) + (1-z(x))\varepsilon p_{\max} + 2\delta \\ \leq \alpha^\star\left(x + \frac{2\gamma}{T} + \varepsilon x + \frac{2\delta}{T}\right).$$

Note that since $z(x) \leq h(x)$, we can proceed by assuming that $z(x) = h(x)$, since this minimizes the effective delivery cost that ALG pays. Using integration by parts and letting $\kappa = \gamma + \delta$, this gives the following condition on $h(x)$:

$$h(x) \geq \frac{p_{\max} + p_{\max}\varepsilon + 2\kappa - \alpha^\star\left(x + \frac{2\kappa}{T} + \varepsilon x\right)}{p_{\max} + p_{\max}\varepsilon - x - \varepsilon x} - \frac{1+\varepsilon}{p_{\max} + p_{\max}\varepsilon - x - \varepsilon x}\int_\ell^x h(u)du.$$

By Grönwall's inequality [60, Theorem 1, p. 356], it follows that:

$$h(x) \geq \frac{p_{\max} + p_{\max}\varepsilon + 2\kappa - \alpha^\star\left(x + \frac{2\kappa}{T} + \varepsilon x\right)}{p_{\max} + p_{\max}\varepsilon - x - \varepsilon x} - (1+\varepsilon)\int_\ell^x \frac{p_{\max} + p_{\max}\varepsilon + 2\kappa - \alpha^\star\left(u + \frac{2\kappa}{T} + \varepsilon u\right)}{(p_{\max} + p_{\max}\varepsilon - u - \varepsilon u)^2}du,$$

$$h(x) \geq \alpha^\star \ln\left[\frac{p_{\max} - x}{p_{\max} - \ell}\right],$$

where $\ell = \frac{(1+\varepsilon)p_{\max}+2\kappa}{\alpha^\star(1+\varepsilon)} - \frac{2\kappa}{T(1+\varepsilon)}$.
Recall that we have a boundary condition that $h(p_{\min}) = 1$—we can combine this with the above to obtain:

$$1 \geq \alpha^\star \ln\left[\frac{p_{\max} - p_{\min}}{p_{\max} - \ell}\right].$$

Letting $\ell = \frac{(1+\varepsilon)p_{\max}+2\kappa}{\alpha^\star(1+\varepsilon)} - \frac{2\kappa}{T(1+\varepsilon)}$, the optimal $\alpha^\star$ is obtained when the above inequality is binding, so we have:

$$1 = \alpha^\star \ln\left[\frac{p_{\max} - p_{\min}}{p_{\max} - \frac{(1+\varepsilon)p_{\max}+2\kappa}{\alpha^\star(1+\varepsilon)} + \frac{2\kappa}{T(1+\varepsilon)}}\right].$$

Solving the above yields that the optimal $\alpha^\star$ for any ALG solving OSDM-S with just flexible demand and increasing delivery cost is lower bounded by:

$$\alpha^\star \geq \left[W\left(-\frac{(p_{\max} - p_{\min})\, e^{-1}}{p_{\max} + \frac{2\kappa}{T(1+\varepsilon)}}\right) + 1\right]^{-1},$$

where $\kappa = \gamma + \delta$, completing the proof. □

Next, we prove a similar lower bound in the case of base demand:

Lemma F.4. *There exists a set of* OSDM-S *instances with only base demand and a monotone increasing delivery cost such that no deterministic online algorithm* ALG *can achieve a competitive ratio better than* α_{IB}*, given by:*

$$\alpha_{\text{IB}} \geq \left[W\left(-\frac{((1-c)p_{\max} - p_{\min})\, e^{-\frac{(1+\varepsilon)p_{\max}+2\delta+\frac{2\gamma}{T}}{(1+\varepsilon)p_{\max}+2\kappa}}}{(1+\varepsilon)p_{\max}+2\kappa} \right) + \frac{(1+\varepsilon)p_{\max}+2\delta+\frac{2\gamma}{T}}{(1+\varepsilon)p_{\max}+2\kappa} \right]^{-1}. \tag{27}$$

Proof. We recall Def. F.2 and consider the same set of instances $\{\mathcal{I}'_x\}_{x\in[p_{\min},p_{\max}]}$, where $\mathcal{I}'_x$ is called an *x-decreasing instance*.

Let $h(x)$ denote a *conversion function* that maps $[p_{\min}, p_{\max}] \to [0,1]$—this describes ALG's purchasing decisions before the final batch of prices ($p_{\max}$) in the instance. Since ALG is deterministic and purchasing is unidirectional (irrevocable), we must have that $h(x-\varrho) \geq h(x)$, i.e., $h(x)$ is non-increasing in $[p_{\min}, p_{\max}]$. Intuitively, the entire demand should be purchased before the end of the sequence if the lowest price appears, i.e., $h(p_{\min}) = 1$.

ALG pays switching cost proportional to $h(x)$, because each good price is "interrupted" by bad prices in the $\mathcal{I}'_x$ instance, captured by $2\gamma h(x)$. If ALG is forced to purchase when the demand arrives, it also pays a switching cost of $2\gamma(1-h(x))$. Thus, the total worst-case switching cost on the purchasing side paid by ALG is 2γ.

The switching cost on the delivery side for both ALG and OPT is 2δ, since the entire base demand arrives at the end of the sequence, and it must be delivered at that time.

Recall that in the price-dependent increasing delivery cost case, the worst case delivery coefficient is $c+\varepsilon \leq 1$, when the storage is full. We bound the total *effective* delivery cost coefficient (i.e., across all time steps) as follows:

$$\sum_{t\in[T]} (cs_t + \varepsilon) = \varepsilon + \sum_{t\in[T]} cs_t \leq \varepsilon + ch(x),$$

where we have used the fact that for any ALG, $\sum_{t\in[T]} s_t \leq h(x)$. Since the delivery cost is price-dependent, the extra delivery cost is realized at the price at the final time step, i.e., $p_{\max}ch(x)$. Then the total cost of ALG on instance $\mathcal{I}'_x$ is described by:

$$\texttt{ALG}(\mathcal{I}'_x) = h(\ell)\ell - \int_\ell^x udh(u) + (1-h(x))p_{\max} + 2\gamma + p_{\max}\varepsilon + p_{\max}ch(x) + 2\delta.$$

The optimal cost on instance $\mathcal{I}'_x$ (if all prices are known in advance) is given by:

$$\texttt{OPT}(\mathcal{I}'_x) \geq \min\left\{ x + \frac{2\gamma}{T} + (c+\varepsilon)p_{\max} + 2\delta,\, p_{\max} + \frac{2\gamma}{T} + 2\delta + p_{\max}\varepsilon \right\}.$$

If ALG is $\alpha^\star$-competitive, then a necessary condition is that for all $x \in [p_{\min}, p_{\max}]$, $\texttt{ALG}(\mathcal{I}'_x) \leq \alpha^\star \texttt{OPT}(\mathcal{I}'_x)$. This imposes a necessary condition on $h(x)$, which can be expressed as the following differential inequality:

$$h(\ell)\ell - \int_\ell^x udh(u) + (1-h(x))p_{\max} + 2\gamma + p_{\max}\varepsilon + p_{\max}ch(x) + 2\delta \leq \alpha^\star\left(x + \frac{2\gamma}{T} + (c+\varepsilon)p_{\max} + 2\delta \right).$$

Using integration by parts and letting $\kappa = \gamma + \delta$, this gives the following condition on $h(x)$:

$$h(x) \geq \frac{p_{\max}(1+\varepsilon) + 2\kappa - \alpha^\star\left(x + \frac{2\gamma}{T} + (c+\varepsilon)p_{\max} + 2\delta \right)}{p_{\max} - p_{\max}c - x} - \frac{1}{p_{\max} - p_{\max}c - x}\int_\ell^x h(u)du.$$

By Grönwall's inequality [60, Theorem 1, p. 356], it follows that:

$$h(x) \geq \frac{p_{\max}(1+\varepsilon)+2\kappa-\alpha^\star\left(x+\frac{2\gamma}{T}+(c+\varepsilon)p_{\max}+2\delta\right)}{p_{\max}-p_{\max}c-x}-\int_\ell^x \frac{p_{\max}(1+\varepsilon)+2\kappa-\alpha^\star\left(u+\frac{2\gamma}{T}+(c+\varepsilon)p_{\max}+2\delta\right)}{(p_{\max}-p_{\max}c-u)^2}du,$$

$$h(x) \geq \alpha^\star \ln\left[\frac{p_{\max}-p_{\max}c-x}{p_{\max}-p_{\max}c-\ell}\right],$$

where $\ell = \frac{p_{\max}(1+\varepsilon)+2\kappa}{\alpha^\star}-\frac{2\gamma}{T}-p_{\max}(\varepsilon+c)-2\delta$.
Recall that we have a boundary condition that $h(p_{\min}) = 1$—we can combine this with the above to obtain:

$$1 \geq \alpha^\star \ln\left[\frac{p_{\max}-p_{\max}c-p_{\min}}{p_{\max}-p_{\max}c-\ell}\right].$$

Letting $\ell = \frac{p_{\max}(1+\varepsilon)+2\kappa}{\alpha^\star}-\frac{2\gamma}{T}-p_{\max}(\varepsilon+c)-2\delta$, the optimal $\alpha^\star$ is obtained when the above inequality is binding, so we have:

$$1 = \alpha^\star \ln\left[\frac{(1-c)p_{\max}-p_{\min}}{p_{\max}-\frac{p_{\max}(1+\varepsilon)+2\kappa}{\alpha^\star}+\frac{2\gamma}{T}+p_{\max}\varepsilon+2\delta}\right].$$

Solving the above yields that the optimal $\alpha^\star$ for any ALG solving OSDM-S with just base demand with increasing delivery cost is lower bounded by:

$$\alpha^\star \geq \left[W\left(-\frac{((1-c)p_{\max}-p_{\min})\,e^{-\frac{(1+\varepsilon)p_{\max}+2\delta+\frac{2\gamma}{T}}{(1+\varepsilon)p_{\max}+2\kappa}}}{(1+\varepsilon)p_{\max}+2\kappa}\right)+\frac{(1+\varepsilon)p_{\max}+2\delta+\frac{2\gamma}{T}}{(1+\varepsilon)p_{\max}+2\kappa}\right]^{-1},$$

completing the proof. □

By Lemmas F.3 and F.4, we have that the best competitive ratio achievable by any deterministic online algorithm for OSDM-S with a monotone increasing delivery cost is $\alpha_{\mathtt{IF}}$ in a general case and $\alpha_{\mathtt{IB}}$ in the case of just base demand. Given parameters of $p_{\min}, p_{\max}, c, \varepsilon, \gamma, \delta, T$, it can be verified that $\alpha_{\mathtt{IF}} < \alpha$ and $\alpha_{\mathtt{IB}} < \alpha_{\mathtt{B}}$, where α and $\alpha_{\mathtt{B}}$ are defined in (3) and (5), respectively, and represent the corresponding optimal competitive bounds for the case of *monotone decreasing delivery cost.* This completes the proof. □

F.4 Proof of Theorem 5.4

In this section, we prove Theorem 5.4, which states that there exists a set of OSDM-T instances (i.e., with a tracking cost) such that no deterministic online algorithm ALG can achieve a competitive ratio better than $\alpha_{\mathtt{T}}$ (for $\alpha_{\mathtt{T}}$ defined in (4)).

Proof. To show this result, we first define a family of instances, and then show that the competitive ratio of any deterministic algorithm is lower bounded under these instances. Consider the following set of instances $\{\mathcal{I}_x\}_{x\in[p_{\min},p_{\max}]}$, where $\mathcal{I}_x$ is called an *x-decreasing instance.*

Definition F.5 (x-decreasing instance for OSDM-T*).* Let $n, m \in \mathbb{N}$ be sufficiently large, and denote $\varrho \coloneqq p_{\max}-p_{\min}/n$.

For $x \in [p_{\min}, p_{\max}]$, we define a decreasing market price sequence as follows: The sequence is partitioned into $n_x \coloneqq 2\cdot\lceil (x-p_{\min})/\varrho\rceil + 1$ alternating batches of prices. The i^{th} batch ($i \in [n_x-2]$) contains m prices $p_{\max}$ if i is odd, and 2 cost functions with coefficient $p_{\max}-(\lceil i/2\rceil)\varrho$ if i is even. The last two batches consist of m cost functions with coefficient $x+\iota$, followed by a final batch of m cost functions with coefficient $p_{\max}$.

The tracking target is defined as follows: $a_t = 0$ for all batches of prices except the second-to-last batch, where $a_t = 1/m$ for all m prices. Note that $\sum_{t=1}^{T} a_t = 1$.

The demand sequence is defined as follows: At $t = 1$, a batch of flexible demand $f_1 = 1$ arrives with deadline $\Delta_1 = T$. All other b_t and f_t are zero.

Let $h(x)$ and $z(x)$ denote *conversion functions* that both map $[p_{\min}, p_{\max}] \to [0, 1]$, respectively. Suppose these arbitrary functions fully describe the actions of a deterministic ALG for OSDM on an instance $\mathcal{I}_x$. Specifically, suppose that $h(x)$ describes ALG's purchasing decisions, and $z(x)$ describes ALG's delivery decisions *before* the arrival of the last batch of prices ($p_{\max}$) in the instance. Note that by definition, $z(x) \leq h(x)$, since the delivery cannot exceed what has been purchased thus far. Note that for large n, processing an instance $\mathcal{I}_{x-\varrho}$ is equivalent to first processing $\mathcal{I}_x$ (besides the last two batches), and then processing batches with prices $x - \varrho$ and $p_{\max}$.

Since ALG is deterministic and both conversions (i.e., purchasing and delivery) are both unidirectional (irrevocable), we must have that $h(x - \varrho) \geq h(x)$ and $z(x - \varrho) \geq z(x)$, i.e., $h(x)$ and $z(x)$ are both non-increasing in $[p_{\min}, p_{\max}]$. Intuitively, the entire demand should be purchased before the end of the sequence if the lowest price appears, i.e., $h(p_{\min}) = 1$. Furthermore, since the lowest price appears at least twice,

Due to the tracking target, any procurement before the second-to-last batch incurs a tracking penalty of $\eta h(x)$, since the target is zero during this time. Furthermore, during the last two batches, ALG incurs a tracking penalty of $\eta + \eta(1 - h(x))$, since the target is $1/m$ during the first m prices in the batch, and zero during the second m prices in the batch—note that ALG must purchase $(1 - h(x))$ at the last price to satisfy the deadline. Thus, the total tracking penalty incurred by ALG is given by 2η.

Recall that in the price-dependent decreasing delivery cost case, the worst case delivery coefficient is $c + \varepsilon \leq 1$, when the storage is empty. We bound the total *effective* delivery cost coefficient (i.e., across all time steps) as follows:

$$\sum_{t \in [T]} (c - cs_t + \varepsilon) = c + \varepsilon - \sum_{t \in [T]} cs_t \leq c + \varepsilon - ch(x),$$

where we have used the fact that for any ALG, $\sum_{t \in [T]} s_t \leq h(x)$. Since the delivery cost is price-dependent, the worst-case for ALG occurs when the *savings* in the delivery cost are realized at the lowest price, i.e., $p_{\min} ch(x)$. Then the total cost of ALG on instance $\mathcal{I}_x$ is described by:

$$\begin{aligned}\texttt{ALG}(\mathcal{I}_x) = h(\ell)\ell - \int_{\ell}^{x} u dh(u) + (1 - h(x))p_{\max} - p_{\min} ch(x) + 2\eta + \\ (c + \varepsilon)\ell z(\ell) - (c + \varepsilon) \int_{\ell}^{x} u dz(u) + (1 - z(x))(c + \varepsilon)p_{\max} + 2\delta.\end{aligned}$$

The optimal cost on instance $\mathcal{I}_x$ (if all prices are known in advance) is given by:

$$\texttt{OPT}(\mathcal{I}_x) = x + \varepsilon x + \frac{2\delta}{T},$$

where note that OPT incurs no tracking penalty by perfectly matching the target. If ALG is $\alpha^\star$-competitive, then a necessary condition is that for all $x \in [p_{\min}, p_{\max}]$, $\texttt{ALG}(\mathcal{I}_x) \leq \alpha^\star \texttt{OPT}(\mathcal{I}_x)$. This imposes a necessary condition on $h(x)$ and $z(x)$, which can be expressed as the following

differential inequality:

$$h(\ell)\ell - \int_{\ell}^{x} u dh(u) + (1-h(x))p_{\max} + 2\eta - p_{\min} c h(x) + (c+\varepsilon)\ell z(\ell) - (c+\varepsilon)\int_{\ell}^{x} u dz(u) + (1-z(x))(c+\varepsilon)p_{\max} + 2\delta \leq \alpha^{\star}\left(x + \varepsilon x + \frac{2\delta}{T}\right).$$

Recall that $z(x) \leq h(x)$. On the instances described above, under the assumption that $\kappa = \gamma + \delta \leq \frac{p_{\max} - p_{\min}}{2}$ (see § 2.5), note that for sufficiently good x (i.e., $x < \ell$) it is strictly better for ALG to deliver as much as possible, i.e., $z(x) = h(x)$, reducing its last-minute delivery cost. Thus, we proceed by assuming that $z(x) = h(x)$. Using integration by parts, this gives the following condition on $h(x)$:

$$h(x)x - \int_{\ell}^{x} h(u)du + (1-h(x))p_{\max} + 2\eta + 2\delta - p_{\min} c h(x) + (c+\varepsilon)xh(x) - (c+\varepsilon)\int_{\ell}^{x} h(u)du + (1-h(x))(c+\varepsilon)p_{\max} \leq \alpha^{\star}\left(x + \frac{2\delta}{T} + \varepsilon x\right).$$

To solve for $\alpha^{\star}$, we can use Grönwall's inequality [60, Theorem 1, p. 356]. First, observe that $h(x)$ must satisfy the following:

$$h(x) \geq \frac{p_{\max} + \frac{2\eta+2\delta}{(1+c+\varepsilon)} - \alpha^{\star}\frac{x+\frac{2\delta}{T}+\varepsilon x}{1+c+\varepsilon}}{p_{\max} - x + \frac{p_{\min} c}{1+c+\varepsilon}} - \frac{1}{p_{\max} - x + \frac{p_{\min} c}{1+c+\varepsilon}}\int_{\ell}^{x} h(u)du.$$

By Grönwall's inequality, it follows that:

$$h(x) \geq \frac{p_{\max} + \frac{2\eta+2\delta}{(1+c+\varepsilon)} - \alpha^{\star}\frac{x+\frac{2\delta}{T}+\varepsilon x}{1+c+\varepsilon}}{p_{\max} - x + \frac{p_{\min} c}{1+c+\varepsilon}} - \int_{\ell}^{x} \frac{p_{\max} + \frac{2\eta+2\delta}{(1+c+\varepsilon)} - \alpha^{\star}\frac{u+\frac{2\delta}{T}+\varepsilon u}{1+c+\varepsilon}}{\left(p_{\max} - u + \frac{p_{\min} c}{1+c+\varepsilon}\right)^2} du,$$

$$h(x) \geq \alpha^{\star}\frac{1+\varepsilon}{1+c+\varepsilon}\ln\left[\frac{(1+c+\varepsilon)(p_{\max}-x) - 2\delta + \frac{2\delta}{m} + p_{\min} c}{(1+c+\varepsilon)(p_{\max}-\ell) - 2\delta + \frac{2\delta}{m} + p_{\min} c}\right].$$

Recall that we have a boundary condition that $h(p_{\min}) = 1$—we can combine this with the above to obtain:

$$1 \geq \alpha^{\star}\frac{1+\varepsilon}{1+c+\varepsilon}\ln\left[\frac{(1+c+\varepsilon)(p_{\max}-p_{\min}) + p_{\min} c}{(1+c+\varepsilon)(p_{\max}-\ell) + p_{\min} c}\right].$$

Letting $\ell = \frac{(1+c+\varepsilon)p_{\max} + 2(\eta+\delta)}{\alpha^{\star}(1+\varepsilon)} - \frac{2\delta}{T(1+\varepsilon)}$, the optimal $\alpha^{\star}$ is obtained when the above inequality is binding, so we have:

$$1 = \alpha^{\star}\frac{1+\varepsilon}{1+c+\varepsilon}\ln\left[\frac{(1+c+\varepsilon)p_{\max} - (1+\varepsilon)p_{\min}}{(1+c+\varepsilon)p_{\max} - \frac{(1+c+\varepsilon)[(1+c+\varepsilon)p_{\max} + 2(\eta+\delta)]}{\alpha^{\star}(1+\varepsilon)} + \frac{(1+c+\varepsilon)2\delta}{T(1+\varepsilon)} + p_{\min} c}\right].$$

Solving the above yields that the optimal $\alpha^{\star}$ for any ALG solving OSDM-T is lower bounded by:

$$\alpha^{\star} \geq \frac{\omega}{\left[W\left(-\frac{((1+c+\varepsilon)p_{\max} - (1+\varepsilon)p_{\min})e^{\frac{-\omega 2\delta/T - p_{\min} c - (1+c+\varepsilon)p_{\max}}{(1+c+\varepsilon)p_{\max} + 2(\eta+\delta)}}}{(1+c+\varepsilon)p_{\max} + 2(\eta+\delta)}\right) + \frac{(1+c+\varepsilon)p_{\max} + p_{\min} c - \omega 2\delta/T}{(1+c+\varepsilon)p_{\max} + 2(\eta+\delta)}\right]},$$

where $\omega = \frac{(1+c+\varepsilon)}{(1+\varepsilon)}$. This completes the proof. □

G Deferred Proofs from § 6.1 (Analysis of PALD Framework)

In this section, we provide full proofs for the results in § 6.1, which describes and analyzes the PALD learning-augmented framework for the OSDM problem. We start by proving Theorem 6.3, which provides a robustness certificate that guarantees PALD's competitive ratio when given learned threshold functions that lie within feasible sets. We also state and prove Corollary G.8, which provides an analogous robustness certificate for the case of OSDM-T, before proving Lemma 6.4, which shows that the feasible sets are convex (and thus efficient to enforce) when the threshold functions are parameterized as piecewise-affine functions.

G.1 Proof of Theorem 6.3

In this section, we prove Theorem 6.3, which states that given learned threshold functions that lie in the feasible sets $\hat{\phi}_b \in \mathcal{R}_b(\rho)$ and $(\hat{\phi}_f, \hat{\psi}_f) \in \mathcal{R}_f(\rho)$ for some $\rho > \alpha$ (α defined in (3)), the PALD algorithm is ρ-robust for OSDM-S.

Proof. Before the main robustness proof, we show that PALD produces a feasible solution to OSDM.

Corollary G.1. PALD *produces a feasible solution to* OSDM*, i.e., it satisfies all demand before their deadlines and never violates the storage capacity constraint.*

Proof. This corollary follows by the same logic as Lemma E.1, since PALD is identical to PALD except for the threshold functions used, and the feasibility proof does not depend on the specific form of the threshold functions. □

We now proceed to prove the worst-case competitive ratio of PALD—for notational brevity, the following considers *any arbitrary* $\mathcal{I} \in \Omega$ (i.e., any arbitrary instance of OSDM). We recall the definition of active and inactive periods from Definition E.2, and the additional notations relating to driver indexing used in the proof of Theorem 4.3 in Appendix E.

We let $D_b = \sum_{i=1}^{n} \sum_{j=1}^{v_i} B_b^{(i,j)}$ and $D_f = \sum_{i=1}^{n} \sum_{j=1}^{\hat{\phi}_i} B_f^{(i,j)}$ denote the total base and flexible demand, respectively, that arrive over the entire time horizon. We let $D = D_b + D_f$ denote the total demand. We let $\hat{s}$ denote the final status of the storage at the end of the time horizon, i.e., $\hat{s} = s_T$.

Finally, we introduce the following notation to characterize the optimal solution: let $G_i(\beta)$ denote the minimum cost of purchasing β units of asset during the i^{th} active period, and let $H_\tau(f)$ denote the minimum cost of purchasing f units of asset during the period $[\tau, \tau+\Delta_\tau]$ (i.e., during the lifetime of the τ^{th} flexible demand). Finally, let $\tilde{p}$ denote the minimum price during idle periods, and let $\bar{p}$ denote the weighted average price during periods with non-zero base demand, i.e., $\bar{p} = \frac{\sum_{\tau \in [T]} p_t \cdot b_t}{D_b}$.

We begin the proof by stating a lower bound on the cost of the offline optimal solution OPT, leveraging the exact same logic as Lemma E.3.

Corollary G.2. *Given that* PALD *produces* n *active periods, let* β_i *denote the asset purchased towards the base demand by the offline optimal solution during the* i^{th} *active period,* $i \in [n]$*, and let* $\tilde{p}$ *denote the minimum price during inactive periods. Then* $\texttt{OPT}(\mathcal{I})$ *is lower bounded as:*

$$\texttt{OPT}(\mathcal{I}) \geq \sum_{i=1}^{n} G_i(\beta_i) + \left(D_b - \sum_{i=1}^{n} \beta_i\right) \tilde{p} + \varepsilon \bar{p} D_b + \sum_{\tau=1}^{T} (1+\varepsilon)\, H_\tau(f_\tau) + \left(D_b + D_f\right) \frac{2\kappa}{T}$$

Next, we state an upper bound on the cost incurred by PALD, leveraging the same logic as Lemma E.4, but with the threshold functions $\hat{\phi}_b$ and $(\hat{\phi}_f, \hat{\psi}_f)$ used by PALD.

Lemma G.3. *Given that* PALD *produces n active periods, let* $\hat{w}_b^{(i,j)}$ *denote the amount of the* j^{th} *base demand driver that has been purchased by the end of the* i^{th} *active period,* $j \in [v_i]$*, and let* $B_b^{(i,j)}$ *denote the total demand associated with the* j^{th} *base demand driver in the* i^{th} *active period. Further, let* $\hat{w}_f^{(\tau)}$ *and* $\hat{v}_f^{(\tau)}$ *denote the purchasing and delivery amounts of the* τ^{th} *flexible demand driver before the deadline, respectively. Then the cost incurred by* PALD *is upper bounded as:*

$$\mathsf{PALD}(\mathcal{I}) \le \sum_{i=1}^{n}\sum_{j=1}^{v_i}\int_0^{\hat{w}_b^{(i,j)}}\hat{\phi}_b^{(i,j)}(u)du + \sum_{\tau=1}^{T}\left(\int_0^{\hat{w}_f^{(\tau)}}\hat{\phi}_f^{(\tau)}(u)du + \int_0^{\hat{v}_f^{(\tau)}}\hat{\psi}^{(\tau)}(z)dz\right) \tag{28}$$

$$+\left(D_b + D_f - \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} - \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}\right)(p_{\max} + 2\gamma) \tag{29}$$

$$+ D_b\left(\bar{p}(c+\varepsilon) + 2\delta\right) + \left(D_f - \sum_{\tau=1}^{T}\hat{v}_f^{(\tau)}\right)(p_{\max}(c+\varepsilon) + 2\delta) \tag{30}$$

$$- cp_{\min}\left(\sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} + \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}\right) + \hat{s}p_{\max}. \tag{31}$$

Proof. The proof follows by the same logic as Lemma E.4—the main change is with respect to (28), to capture the fact that the threshold functions used by PALD are the learned variants, denoted by $\hat{\phi}_b$ and $(\hat{\phi}_f, \hat{\psi}_f)$, instead of the analytical variants ϕ_b and (ϕ_f, ψ_f) used by PAAD.

Any additional purchasing needed to satisfy demand (i.e., that isn't captured by the integrals over the threshold functions) is exactly captured by the existing logic in the proof of Lemma E.4, completing the proof. □

Leveraging the definition of the feasible robustness sets $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$ in Definitions 6.1 and 6.2, respectively, we have the following technical lemmas to relate the threshold functions and the optimal cost:

Lemma G.4. *Given a learned threshold function* $\hat{\phi}_b(\cdot)$ *that lies within the feasible set* $\mathcal{R}_b(\rho)$ *for some* $\rho > \alpha$*, the following relation always holds:*

$$\int_0^w \hat{\phi}_b(u)du + (1-w)(p_{\max}+2\gamma) + p_{\max}(c+\varepsilon) + 2\delta - cwp_{\min} = \rho\left[\hat{\phi}_b(w) - 2\gamma + \varepsilon p_{\max} + \frac{2\kappa}{T}\right] \forall w \in [0,1].$$

Lemma G.5. *Given learned threshold functions* $\hat{\phi}_f(\cdot)$ *and* $\hat{\psi}(\cdot)$ *that lie within the feasible set* $\mathcal{R}_f(\rho)$ *for some* $\rho > \alpha$*, the following relation always holds:*

$$\int_0^w \hat{\phi}_f(u)du + (1-w)(p_{\max}+2\gamma) - cwp_{\min} + \int_0^v \hat{\psi}_f(z)dz + (1-v)(p_{\max}(c+\varepsilon)+2\delta) =$$
$$\rho\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f(w) + \hat{\psi}_f(v) - 2\kappa\right) + \frac{2\kappa}{T}\right] \forall w \in [0,1], v \in [0,w],$$

The proofs of both lemmas follow directly from the definitions of the feasible sets $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$. Using the results in Corollary G.2 and Lemmas G.3, G.4, and G.5, we claim that the following holds:

$$\frac{\mathsf{PALD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})} \le \rho.$$

To show this result, we first substitute the bounds from Corollary G.2 and Lemma G.3 into the left-hand side of the above equation. As in the proof of Theorem 4.3, we define some shorthand notation to facilitate the presentation.

Let $Q = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\int_0^{\hat{w}_b^{(i,j)}}\hat{\phi}_b^{(i,j)}(u)du + \sum_{\tau=1}^{T}\left(\int_0^{\hat{w}_f^{(\tau)}}\hat{\phi}_f^{(\tau)}(u)du + \int_0^{\hat{v}_f^{(\tau)}}\hat{\psi}^{(\tau)}(z)dz\right)$ denote the integrals over the thresholds. Let $\hat{W}_b = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)}$, let $\hat{W}_f = \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}$, and let $\hat{W} = \hat{W}_b + \hat{W}_f$ denote the total purchasing by all drivers.
Further, let $\hat{V}_b = \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{v}_b^{(i,j)}$ denote the total delivery by all flexible demand drivers.
Let $\boldsymbol{\beta} = \sum_{i=1}^{n}\beta_i$ denote the total amount of asset purchased towards the base demand by the optimal solution during all active periods, noting that $D_b - \boldsymbol{\beta} \geq 0$ by definition. Finally, let $D = D_b + D_f$ denote the total demand. Substituting the bounds from Corollary G.2 and Lemma G.3 into the left-hand side of the above inequality, we have:

$$\frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\gamma)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}+(D_b-\boldsymbol{\beta})(p_{\max}+2\kappa+\bar{p}(c+\varepsilon))}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\kappa}{T}+(D_b-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\kappa}{T})}$$

Then, we have the following:

$$\leq \max\left\{\frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\gamma)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\kappa}{T}},\right.$$
$$\left.\frac{(D_b-\boldsymbol{\beta})(p_{\max}+2\kappa+\bar{p}(c+\varepsilon))}{(D_b-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\kappa}{T})}\right\},$$

where the definition of $\tilde{p}$ ensures that the second term in the max is at most ρ. Note that this follows because $\tilde{p}$ is the best price during inactive periods (i.e., a price that is not accepted by any base demand driver), and by the definition of the feasible set $\mathcal{R}_b(\rho)$, any ρ-robust threshold function $\hat{\phi}_b$ must satisfy $p_{\max}(1+c+\varepsilon)+2\kappa = \rho\left[\hat{\phi}_b(0) - 2\gamma + \varepsilon p_{\max} + \frac{2\kappa}{T}\right]$ for $w = 0$.
We now focus on the first term. For the sake of contradiction, suppose that

$$\frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\gamma)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\kappa}{T}} > \alpha_{\mathsf{T}}. \tag{32}$$

Instead of working directly with the expression in terms of $\boldsymbol{\beta}$, we first reason about how the cost of OPT and PALD relate to one another in terms of $\sum_{i=1}^{n}\sum_{j=1}^{v_i}B_b^{(i,j)}$, the total demand assigned to base drivers. We introduce the following notation for the sake of brevity: let $\boldsymbol{B} = \sum_{i=1}^{n}\sum_{j=1}^{v_i}B_b^{(i,j)}$ denote the total base demand. Then, we have the following relation:

$$\frac{Q+(\boldsymbol{B}+D_f-\hat{W})(p_{\max}+2\gamma)+\boldsymbol{B}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(B_i)+\varepsilon\bar{p}\boldsymbol{B}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{B})\frac{2\kappa}{T}},$$
$$\geq \frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\gamma)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}+(\boldsymbol{B}-\boldsymbol{\beta})(p_{\max}+\bar{p}(c+\varepsilon)+2\kappa)}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\kappa}{T}+(\boldsymbol{B}-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\kappa}{T})},$$
$$> \rho.$$

where in the second inequality, we have used Lemma E.5. We proceed to work with the expression in terms of $\boldsymbol{B}$.

During the i^{th} active period and the lifetime of the j^{th} base demand driver, the minimum marginal purchasing price observed is given by $\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma$ by the definition of the threshold function.

Note that this follows because the feasible set $\mathcal{R}_b(\rho)$ ensures that $\hat{\phi}_b^{(i,j)}(w)$ is monotone non-increasing for $w \in [0,1]$, and it is given that $\hat{\phi}_b^{(i,j)}(1) \le p_{\min} + 2\gamma$, guaranteeing that the true minimum price is captured by the threshold function.

Using the same logic and by definition of the feasible set $\mathcal{R}_f(\rho)$, we have that during the lifetime of the τ^{th} flexible demand driver, the minimum marginal purchasing and delivery cost observed is given by $\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2\kappa\right)$. This gives the following lower bounds on the terms that depend on G_i and H_τ, respectively:

$$\sum_{i=1}^{n} G_i\left(\sum_{j=1}^{v_i} B_b^{(i,j)}\right) \ge \sum_{i=1}^{n}\sum_{j=1}^{v_i}\left(\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma\right) \times B_b^{(i,j)}$$

$$\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau) \ge \sum_{\tau=1}^{T}\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\gamma+\delta)\right) \times f_\tau.$$

Substituting these bounds into the previous expression, we have that the left-hand-side of (32) is less than or equal to:

$$\le \frac{Q + (\boldsymbol{B} + D_f - \hat{W})(p_{\max} + 2\gamma) + \boldsymbol{B}\,(\bar{p}(c+\varepsilon) + 2\delta) + (D_f - \hat{V}_f)(p_{\max}(c+\varepsilon) + 2\delta) - cp_{\min}\hat{W}}{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left(\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma\right)B_b^{(i,j)} + \varepsilon\bar{p}\boldsymbol{B} + \sum_{\tau=1}^{T}\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2\kappa\right)f_\tau + (D_f + \boldsymbol{B})\frac{2\kappa}{T}}.$$

By rearranging the terms in the above and substituting for Q, we obtain the following:

$$= \frac{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left[X_b^{(i,j)}\right] + \sum_{\tau=1}^{T}\left[X_f^{(\tau)}\right]}{\sum_{i=1}^{n}\sum_{j=1}^{v_i}\left[\left(\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma + \varepsilon\bar{p} + \frac{2\kappa}{T}\right)B_b^{(i,j)}\right] + \sum_{\tau=1}^{T}\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2\kappa\right)f_\tau + f_\tau\frac{2\kappa}{T}\right]}, \tag{33}$$

where $X_b^{(i,j)}$ and $X_f^{(\tau)}$ are defined as follows (for each driver):

$$X_b^{(i,j)} = \int_0^{\hat{w}_b^{(i,j)}} \hat{\phi}_b^{(i,j)}(u)du + (B_b^{(i,j)} - \hat{w}_b^{(i,j)})(p_{\max} + 2\gamma) + (p_{\max}(c+\varepsilon) + 2\delta)\,B_b^{(i,j)} - c\hat{w}_b^{(i,j)}p_{\min},$$

$$X_f^{(\tau)} = \int_0^{\hat{w}_f^{(\tau)}} \hat{\phi}_f^{(\tau)}(u)du + (f_\tau - \hat{w}_f^{(\tau)})(p_{\max} + 2\gamma) + \int_0^{\hat{v}_f^{(\tau)}} \hat{\psi}_f^{(\tau)}(u)du + (p_{\max}(c+\varepsilon) + 2\delta)\,(f_\tau - \hat{v}_f^{(\tau)}) - c\hat{w}_f^{(\tau)}p_{\min}.$$

Since (33) is $> \rho$ by assumption, one of the following cases must be true:

Case I: There exists some $i \in [n]$ and $j \in [v_i]$ such that

$$X_b^{(i,j)} > \rho B_b^{(i,j)}\left(\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\gamma + \varepsilon\bar{p} + \frac{2\kappa}{T}\right).$$

Case II: There exists some $\tau \in [T]$ such that

$$X_f^{(\tau)} > \rho f_\tau\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\gamma+\delta)\right) + \frac{2\kappa}{T}\right],$$

But, if either of these two cases are true (i.e., for any unit of base or flexible demand), they contradict Lemmas G.4 and G.5, respectively. Thus, we have a contradiction, and the original assumption that (33) is $> \rho$ must be false. This completes the proof that $\frac{\mathsf{PALD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})} \le \rho$.
Using this result, we have:

$$\mathsf{PALD}(\mathcal{I}) \le \rho\mathsf{OPT}(\mathcal{I}) + p_{\max}\hat{s} \le \rho\mathsf{OPT}(\mathcal{I}) + p_{\max}S,$$

where $p_{\max}S$ is a constant. This shows that PALD is ρ-competitive under Definition 2.1, which completes the proof. □

G.2 PALD Robustness Certificate for OSDM-T

In this section, we define the feasible robust sets for PALD in the context of the OSDM-T problem (i.e., in the tracking case), and state and prove the corresponding robustness certificate. These (re-)definitions account for the tracking cost on the purchasing side of the problem.

Definition G.6 (Robust threshold set for base drivers in OSDM-T*). Given a target robustness $\rho > \alpha_{\mathsf{T}}$ (where α_{T} is defined in (4)), a learned threshold function $\hat{\phi}_b$ for base demand drivers must lie in the following feasible set $\mathcal{R}'_b(\rho)$:*

$$\mathcal{R}'_b(\rho) = \Bigg\{ \phi_b : [0,1] \to [p_{\min}, p_{\max}] \;\Bigg|\; \phi_b \text{ is monotone non-increasing, } \phi_b(1) \le p_{\min} + 2\eta, \text{ and } \forall w \in [0,1] : \\ \Phi_b(0,w) + (1-w)(p_{\max} + 2\eta) + p_{\max}(c+\varepsilon) + 2\delta - cwp_{\min} \le \rho\left[\phi_b(w) - 2\eta + \varepsilon p_{\max} + {}^{2\delta}/_{T}\right] \Bigg\}.$$

Definition G.7 (Robust threshold set for flexible drivers in OSDM-T*). Given a target robustness $\rho > \alpha_{\mathsf{T}}$ (where α_{T} is defined in (4)), the learned threshold functions $\hat{\phi}_f$ and $\hat{\psi}_f$ for flexible demand drivers must lie in the following joint feasible set $\mathcal{R}'_f(\rho)$, where $\omega = \frac{1+c+\varepsilon}{1+\varepsilon}$:*

$$\mathcal{R}'_f(\rho) = \Bigg\{ (\phi_f, \psi_f) : [0,1]^2 \to [p_{\min}, p_{\max}]^2 \;\Bigg|\; \phi_f, \psi_f \text{ are monotone non-increasing,} \\ \phi_f(1) \le p_{\min} + 2\eta, \psi_f(1) \le p_{\min}(c+\varepsilon) + 2\delta, \text{ and } \forall w \in [0,1], v \in [0,w] : \\ \Phi_f(0,w) + (1-w)(p_{\max} + 2\eta) - cwp_{\min} + \Psi_f(0,v) + (1-v)(p_{\max}(c+\varepsilon) + 2\delta) \\ \le \rho\left[{}^{1}/_{\omega}\left(\phi_f(w) + \psi_f(v) - 2(\eta+\delta)\right) + {}^{2\delta}/_{T}\right] \Bigg\}.$$

Under Def. G.6 and Def. G.7, we have the following robustness certificate for PALD when given any learned thresholds $\hat{\phi}_b, \hat{\phi}_f$ and $\hat{\psi}_f$ that lie in the respective feasible sets:

Corollary G.8. *Given learned threshold functions that lie in the feasible sets $\hat{\phi}_b \in \mathcal{R}'_b(\rho)$ and $(\hat{\phi}_f, \hat{\psi}_f) \in \mathcal{R}'_f(\rho)$ for some $\rho > \alpha_{\mathsf{T}}$ (α_{T} defined in (4)), the* PALD *algorithm is ρ-robust for* OSDM-T.

Proof. The feasibility of PALD's solution follows from Corollary G.1.

We now proceed to prove the worst-case competitive ratio of PALD—for notational brevity, the following considers *any arbitrary* $\mathcal{I} \in \Omega$ (i.e., any arbitrary instance of OSDM). We begin the proof by stating a lower bound on the cost of the offline optimal solution OPT, leveraging the same logic as Lemma E.7.

Corollary G.9. *Given that* PALD *produces n active periods for* OSDM*, let β_i denote the asset purchased towards the base demand by the offline optimal solution during the i^{th} active period, $i \in [n]$, and let $\tilde{p}$ denote the minimum price during inactive periods. Then* $\mathsf{OPT}(\mathcal{I})$ *is lower bounded as:*

$$\mathsf{OPT}(\mathcal{I}) \ge \sum_{i=1}^{n} G_i(\beta_i) + \left(D_b - \sum_{i=1}^{n} \beta_i\right)\tilde{p} + \varepsilon\bar{p}D_b + \sum_{\tau=1}^{T} (1+\varepsilon)\, H_\tau(f_\tau) + \left(D_b + D_f\right)\frac{2\delta}{T}$$

Next, we state an upper bound on the cost incurred by PALD for OSDM-T, leveraging the same logic as Lemma E.8, but with the threshold functions $\hat{\phi}_b$ and $(\hat{\phi}_f, \hat{\psi}_f)$ used by PALD.

Lemma G.10. *Given that* PALD *produces n active periods, let $\hat{w}_b^{(i,j)}$ denote the amount of the j^{th} base demand driver that has been purchased by the end of the i^{th} active period, $j \in [v_i]$, and let $B_b^{(i,j)}$ denote the total demand associated with the j^{th} base demand driver in the i^{th} active period. Further, let*

$\hat{w}_f^{(\tau)}$ and $\hat{v}_f^{(\tau)}$ denote the purchasing and delivery amounts of the τ^{th} flexible demand driver before the deadline, respectively. Then the cost incurred by PALD *is upper bounded as:*

$$\mathsf{PALD}(\mathcal{I}) \leq \sum_{i=1}^{n}\sum_{j=1}^{v_i}\int_0^{\hat{w}_b^{(i,j)}}\hat{\phi}_b^{(i,j)}(u)du + \sum_{\tau=1}^{T}\left(\int_0^{\hat{w}_f^{(\tau)}}\hat{\phi}_f^{(\tau)}(u)du + \int_0^{\hat{v}_f^{(\tau)}}\hat{\psi}^{(\tau)}(z)dz\right) \tag{34}$$

$$+\left(D_b + D_f - \sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} - \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}\right)(p_{\max}+2\eta) \tag{35}$$

$$+ D_b\left(\bar{p}(c+\varepsilon)+2\delta\right) + \left(D_f - \sum_{\tau=1}^{T}\hat{v}_f^{(\tau)}\right)\left(p_{\max}(c+\varepsilon)+2\delta\right) \tag{36}$$

$$- cp_{\min}\left(\sum_{i=1}^{n}\sum_{j=1}^{v_i}\hat{w}_b^{(i,j)} + \sum_{\tau=1}^{T}\hat{w}_f^{(\tau)}\right) + \hat{s}p_{\max}. \tag{37}$$

Proof. The proof follows by the same logic as Lemma G.3—the main change is with respect to (35), to capture the tracking cost (with parameter η on the purchasing side of the problem). □

Leveraging the definition of the feasible robustness sets $\mathcal{R}_b'(\rho)$ and $\mathcal{R}_f'(\rho)$ in Definitions G.6 and G.7, respectively, we have the following lemmas to relate the threshold functions and the optimal cost:

Lemma G.11. *Given a learned threshold function $\hat{\phi}_b(\cdot)$ that lies within the feasible set $\mathcal{R}_b'(\rho)$ for some $\rho > \alpha_{\mathsf{T}}$, the following relation always holds:*

$$\int_0^w \hat{\phi}_b(u)du + (1-w)(p_{\max}+2\eta) + p_{\max}(c+\varepsilon) + 2\delta - cwp_{\min} = \rho\left[\hat{\phi}_b(w) - 2\eta + \varepsilon p_{\max} + \frac{2\delta}{T}\right] \forall w \in [0,1].$$

Lemma G.12. *Given learned threshold functions $\hat{\phi}_f(\cdot)$ and $\hat{\psi}(\cdot)$ that lie within the feasible set $\mathcal{R}_f'(\rho)$ for some $\rho > \alpha_{\mathsf{T}}$, the following relation always holds:*

$$\int_0^w \hat{\phi}_f(u)du + (1-w)(p_{\max}+2\eta) - cwp_{\min} + \int_0^v \hat{\psi}_f(z)dz + (1-v)(p_{\max}(c+\varepsilon)+2\delta) =$$
$$\rho\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f(w) + \hat{\psi}_f(v) - 2(\eta+\delta)\right) + \frac{2\delta}{T}\right] \forall w \in [0,1], v \in [0,w],$$

The proofs of both lemmas follow directly from the definitions of the feasible sets $\mathcal{R}_b'(\rho)$ and $\mathcal{R}_f'(\rho)$. Using Corollary G.9 and Lemmas G.10, G.11, and G.12, we claim that the following holds:

$$\frac{\mathsf{PALD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathsf{OPT}(\mathcal{I})} \leq \rho.$$

To show this result, we first substitute the bounds from Corollary G.2 and Lemma G.3 into the left-hand side of the above equation. We use the same shorthand notation as in the proof of Theorem 6.3 to facilitate the presentation.

$$\frac{Q + (\boldsymbol{\beta} + D_f - \hat{W})(p_{\max}+2\eta) + \boldsymbol{\beta}\left(\bar{p}(c+\varepsilon)+2\delta\right) + (D_f - \hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta) - cp_{\min}\hat{W} + (D_b - \boldsymbol{\beta})(p_{\max}+2(\eta+\delta)+\bar{p}(c+\varepsilon))}{\sum_{i=1}^{n} G_i(\beta_i) + \varepsilon\bar{p}\boldsymbol{\beta} + \sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau) + (D_f + \boldsymbol{\beta})\frac{2\delta}{T} + (D_b - \boldsymbol{\beta})(\tilde{p} + \varepsilon\bar{p} + \frac{2\delta}{T})}$$

Then, we have the following:

$$\leq \max\left\{\frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\delta}{T}}, \right.$$
$$\left. \frac{(D_b-\boldsymbol{\beta})(p_{\max}+2(\eta+\delta)+\bar{p}(c+\varepsilon))}{(D_b-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\delta}{T})}\right\},$$

where the definition of $\tilde{p}$ ensures that the second term in the max is at most ρ.
We now focus on the first term. For the sake of contradiction, suppose that

$$\frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\delta}{T}}>\alpha_{\mathrm{T}}. \tag{38}$$

Instead of working directly with the expression in terms of $\boldsymbol{\beta}$, we first reason about how the cost of OPT and PALD relate to one another in terms of $\sum_{i=1}^{n}\sum_{j=1}^{\nu_i}B_b^{(i,j)}$, the total demand assigned to base drivers. We introduce the following notation for the sake of brevity: let $\boldsymbol{B}=\sum_{i=1}^{n}\sum_{j=1}^{\nu_i}B_b^{(i,j)}$ denote the total base demand. Then, we have the following relation:

$$\frac{Q+(\boldsymbol{B}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{B}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}G_i(B_i)+\varepsilon\bar{p}\boldsymbol{B}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{B})\frac{2\delta}{T}},$$
$$\geq \frac{Q+(\boldsymbol{\beta}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{\beta}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}+(\boldsymbol{B}-\boldsymbol{\beta})(p_{\max}+\bar{p}(c+\varepsilon)+2\kappa)}{\sum_{i=1}^{n}G_i(\beta_i)+\varepsilon\bar{p}\boldsymbol{\beta}+\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)+(D_f+\boldsymbol{\beta})\frac{2\delta}{T}+(\boldsymbol{B}-\boldsymbol{\beta})(\tilde{p}+\varepsilon\bar{p}+\frac{2\delta}{T})},$$
$$>\rho.$$

where in the second inequality, we have used Lemma E.5. We proceed to work with the expression in terms of $\boldsymbol{B}$. During the i^{th} active period and the lifetime of the j^{th} base demand driver, the minimum marginal purchasing price observed is given by $\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)})-2\eta$ by the definition of the threshold function. Note that this follows because the feasible set $\mathcal{R}'_b(\rho)$ ensures that $\hat{\phi}_b^{(i,j)}(w)$ is monotone non-increasing for $w\in[0,1]$, and it is given that $\hat{\phi}_b^{(i,j)}(1)\leq p_{\min}+2\eta$, guaranteeing that the true minimum price is captured by the threshold function.

Using the same logic and by definition of the feasible set $\mathcal{R}'_f(\rho)$, we have that during the lifetime of the τ^{th} flexible demand driver, the minimum marginal purchasing and delivery cost observed is given by $\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)})+\hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)})-2(\eta+\delta)\right)$. This gives the following lower bounds on the terms that depend on G_i and H_τ, respectively:

$$\sum_{i=1}^{n}G_i\left(\sum_{j=1}^{\nu_i}B_b^{(i,j)}\right)\geq\sum_{i=1}^{n}\sum_{j=1}^{\nu_i}\left(\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)})-2\eta\right)\times B_b^{(i,j)}$$
$$\sum_{\tau=1}^{T}(1+\varepsilon)H_\tau(f_\tau)\geq\sum_{\tau=1}^{T}\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)})+\hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)})-2(\eta+\delta)\right)\times f_\tau.$$

Substituting these bounds into the previous expression, we have that the left-hand-side of (38) is less than or equal to:

$$\leq\frac{Q+(\boldsymbol{B}+D_f-\hat{W})(p_{\max}+2\eta)+\boldsymbol{B}\,(\bar{p}(c+\varepsilon)+2\delta)+(D_f-\hat{V}_f)(p_{\max}(c+\varepsilon)+2\delta)-cp_{\min}\hat{W}}{\sum_{i=1}^{n}\sum_{j=1}^{\nu_i}\left(\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)})-2\eta\right)B_b^{(i,j)}+\varepsilon\bar{p}\boldsymbol{B}+\sum_{\tau=1}^{T}\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)})+\hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)})-2(\eta+\delta)\right)f_\tau+(D_f+\boldsymbol{B})\frac{2\delta}{T}}.$$

By rearranging the terms in the above and substituting for Q, we obtain the following:

$$= \frac{\sum_{i=1}^{n}\sum_{j=1}^{\nu_i}\left[X_b^{(i,j)}\right] + \sum_{\tau=1}^{T}\left[X_f^{(\tau)}\right]}{\sum_{i=1}^{n}\sum_{j=1}^{\nu_i}\left[\left(\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\eta + \varepsilon\bar{p} + \frac{2\delta}{T}\right) B_b^{(i,j)}\right] + \sum_{\tau=1}^{T}\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\eta+\delta)\right) f_\tau + f_\tau \frac{2\delta}{T}\right]}, \tag{39}$$

where $X_b^{(i,j)}$ and $X_f^{(\tau)}$ are defined as follows (for each driver):

$$X_b^{(i,j)} = \int_0^{\hat{w}_b^{(i,j)}} \hat{\phi}_b^{(i,j)}(u)du + (B_b^{(i,j)} - \hat{w}_b^{(i,j)})(p_{\max} + 2\eta) + (p_{\max}(c+\varepsilon) + 2\delta)\, B_b^{(i,j)} - c\hat{w}_b^{(i,j)} p_{\min},$$

$$X_f^{(\tau)} = \int_0^{\hat{w}_f^{(\tau)}} \hat{\phi}_f^{(\tau)}(u)du + (f_\tau - \hat{w}_f^{(\tau)})(p_{\max} + 2\eta) + \int_0^{\hat{v}_f^{(\tau)}} \hat{\psi}_f^{(\tau)}(u)du + (p_{\max}(c+\varepsilon) + 2\delta)\,(f_\tau - \hat{v}_f^{(\tau)}) - c\hat{w}_f^{(\tau)} p_{\min}.$$

Since (39) is $> \rho$ by assumption, one of the following cases must be true:

Case I: There exists some $i \in [n]$ and $j \in [\nu_i]$ such that

$$X_b^{(i,j)} > \rho B_b^{(i,j)}\left(\hat{\phi}_b^{(i,j)}(\hat{w}_b^{(i,j)}) - 2\eta + \varepsilon\bar{p} + \frac{2\delta}{T}\right).$$

Case II: There exists some $\tau \in [T]$ such that

$$X_f^{(\tau)} > \rho f_\tau\left[\frac{1+\varepsilon}{1+c+\varepsilon}\left(\hat{\phi}_f^{(\tau)}(\hat{w}_f^{(\tau)}) + \hat{\psi}_f^{(\tau)}(\hat{v}_f^{(\tau)}) - 2(\eta+\delta)\right) + \frac{2\delta}{T}\right],$$

But, if either of these two cases are true (i.e., for any unit of base or flexible demand), they contradict Lemmas G.11 and G.12, respectively. Thus, we have a contradiction, and the original assumption that (39) is $> \rho$ must be false. This completes the proof that $\frac{\mathtt{PALD}(\mathcal{I}) - p_{\max}\hat{s}}{\mathtt{OPT}(\mathcal{I})} \le \rho$.

Using this result, we have:

$$\mathtt{PALD}(\mathcal{I}) \le \rho\mathtt{OPT}(\mathcal{I}) + p_{\max}\hat{s} \le \rho\mathtt{OPT}(\mathcal{I}) + p_{\max}S,$$

where $p_{\max}S$ is a constant. This shows that PALD is ρ-competitive for OSDM-T under Definition 2.1, which completes the proof.

□

G.3 Proof of Lemma 6.4

In this section, we prove Lemma 6.4, which states that if ϕ_b, ϕ_f, ψ_f are piecewise-affine functions parameterized by $\mathbf{y}_b, \mathbf{y}_f, \mathbf{y}_\psi \in \mathbb{R}^K$ for $K > 1$, the feasible sets $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$ defined in Def. 6.1 and Def. 6.2 are convex sets in $\mathbf{y}_b$ and $(\mathbf{y}_f, \mathbf{y}_\psi)$, respectively.

Proof of Lemma 6.4. Denote the grid discretization points by $0 = w_1 < w_2 < \cdots < w_K = 1$, with $w_j = \frac{j-1}{K-1}$. Each threshold function $\phi : [0,1] \to \mathbb{R}$ or $\psi : [0,1] \to \mathbb{R}$ is parameterized by its knot values $\mathbf{y} = (y_1, \ldots, y_K)$ and linear interpolation on each interval $[w_j, w_{j+1}]$.

Then we want to show that $\mathcal{R}_b(\rho)$ is a convex set in $\mathbf{y}_b$ and $\mathcal{R}_f(\rho)$ is a convex set in $(\mathbf{y}_f, \mathbf{y}_\psi)$.

First, we introduce the hat-basis representation of piecewise-affine functions, which will be useful in the proof. Then we prove the convexity of $\mathcal{R}_b(\rho)$ and $\mathcal{R}_f(\rho)$ in their respective parameter spaces.

Let $\{h_j(\cdot)\}_{j=1}^K$ be the standard PWA "tent" basis: each h_j is piecewise-linear, $h_j(w_i) = \mathbf{1}\{i = j\}$, supported on $[w_{j-1}, w_{j+1}]$ (with boundary conventions for $j = 1, K$), and $\sum_j h_j(u) = 1$ for all $u \in [0,1]$. Under linear interpolation, we have:

$$\phi(u) = \sum_{j=1}^{K} y_j h_j(u), \qquad \Phi(0,w) = \int_0^w \phi(u)\,du = \sum_{j=1}^{K} y_j H_j(w),$$

so for fixed u, w, both $\phi(u)$ and $\Phi(0, w)$ are affine in the parameter vector $\mathbf{y}$.

Base Drivers. All defining constraints of $\mathcal{R}_b(\rho)$ are affine in $\mathbf{y}_b$: (i) range and endpoint: $p_{\min} \le y_{b,j} \le p_{\max}$ and $y_{b,K} \le p_{\min} + 2\gamma$ (box and halfspace); (ii) monotone non-increasing: $y_{b,j+1} - y_{b,j} \le 0$ for $j = 1, \dots, K-1$ (segment slopes); (iii) robustness inequality for fixed $w \in [0, 1]$:

$$\sum_{j=1}^{K} \big(H_j(w) - \rho\, h_j(w)\big)\, y_{b,j} \;\le\; \rho\big(\varepsilon p_{\max} + \tfrac{2\kappa}{T} - 2\gamma\big) - (1-w)(p_{\max} + 2\gamma) - p_{\max}(c+\varepsilon) - 2\delta + cwp_{\min},$$

which is a single affine halfspace in $\mathbf{y}_b$. The condition "for all w" is the intersection (possibly uncountable) of affine halfspaces, hence convex. Intersecting with the box and linear inequalities preserves convexity. Thus $\mathcal{R}_b(\rho)$ is convex.

Flexible Drivers. Analogously, in the product space $(\mathbf{y}_f, \mathbf{y}_\psi)$: (i) range and endpoints give affine constraints on $y_{f,j}$ and $y_{\psi,j}$; (ii) monotonicity gives $y_{f,j+1} - y_{f,j} \le 0$ and $y_{\psi,j+1} - y_{\psi,j} \le 0$; (iii) for fixed (w, v) with $0 \le v \le w \le 1$, the robustness inequality

$$\sum_{j=1}^{K} H_j(w)\, y_{f,j} + \sum_{j=1}^{K} H_j(v)\, y_{\psi,j} \le$$
$$\rho\left[\tfrac{1}{\omega}\left(\sum_{j=1}^{K} h_j(w)\, y_{f,j} + \sum_{j=1}^{K} h_j(v)\, y_{\psi,j} - 2\kappa\right) + \tfrac{2\kappa}{T}\right] - (1-w)(p_{\max} + 2\gamma) - (1-v)(p_{\max}(c+\varepsilon) + 2\delta) + cwp_{\min}$$

is an affine halfspace in $(\mathbf{y}_f, \mathbf{y}_\psi)$. Intersecting these halfspaces over all (w, v) and with the linear constraints yields a convex set. Hence $\mathcal{R}_f(\rho)$ is convex.

Therefore, both feasible sets are convex in their respective parameter spaces. □