Robust RegretNet for Combinatorial Auctions: Distributionally-Robust Approximate DSIC Under Wasserstein Shift

Liz Lemma Future Detective

January 16, 2026

Abstract

Differentiable mechanism design (e.g., RegretNet and its combinatorial extensions such as CANet/CAFormer) learns revenue-maximizing auctions by penalizing expected ex-post regret as a relaxation of DSIC. A central deployment weakness is that both regret estimation and the training distribution can be misspecified: small IC violations may be amplified by distribution shift and by algorithmic bidders that search for profitable deviations. We propose Robust RegretNet, which replaces in-distribution expected regret with a distributionally-robust regret objective over an ambiguity set around the empirical training distribution. For Wasserstein ambiguity sets, we obtain a clean dual characterization showing the robust objective equals empirical risk plus an explicit Lipschitz-penalty term, yielding a tractable training criterion. We then derive high-probability deployment guarantees: for any valuation distribution within Wasserstein distance δ of the training distribution, the learned mechanism's expected ex-post regret is bounded by its robust empirical regret plus a capacity-dependent generalization term, and revenue is within a corresponding gap of the best-in-class robust mechanism. We sketch an optimization procedure compatible with CAFormer-style feasibility layers and evaluate robustness under synthetic and platform-inspired shift stress tests. The results advance differentiable economics from empirical approximation toward deployable, policy-relevant guarantees in modern (2026) strategic environments.

Table of Contents

1. 1. Introduction and Motivation: why distribution shift + strategic search break in-distribution regret guarantees; contributions and positioning relative to RegretNet and CAFormer/CANet.

2. 2. Baseline: Differentiable Mechanism Design with Regret Penalties: formal definition of regret, IR by architecture, combinatorial feasibility

constraints; recap of where existing guarantees stop (expected regret on training distribution).

3. 3. A Clean Model of Distribution Shift: ambiguity sets (Wasserstein and f-divergence) and why they are natural in platform settings; definitions of robust regret and robust revenue.

4. 4. Robust Objective and Dual Representations: show closed-form/dual forms for Wasserstein balls (empirical mean + Lipschitz radius term); discuss when numerical inner maximization is needed (e.g., general f-divergences or non-Lipschitz losses).

5. 5. Lipschitz and Capacity Control for Auction Networks: sufficient conditions (bounded weights, Lipschitz activations, bounded valuation domains) implying Lipschitz regret and revenue; covering-number/Rademacher-based complexity control adapted from RegretNet generalization arguments.

6. 6. Main Theorems: Deployment-Time Regret and Revenue Guarantees Under Shift: high-probability bounds; interpretation as robust approximate DSIC; quantile-regret corollaries.

7. 7. Algorithms: Robust Training with Strong Deviation Oracles: practical training loop (inner deviation maximization + robust outer objective); how to estimate/upper-bound Lipschitz constants; implementation with CAFormer/CANet feasibility layers.

8. 8. Experiments: Synthetic Shifts and Platform-Inspired Stress Tests: (i) controlled Wasserstein shift, (ii) covariate/context shift, (iii) adversarial reweighting; compare Robust RegretNet vs RegretNet/CAFormer; evaluate regret against stronger deviation searches.

9. 9. Discussion and Policy/Engineering Implications: choosing ambiguity radius, monitoring shift, safe updates; limitations (oracle strength, nonconvexity) and next steps.

1 Introduction and Motivation

Neural approaches to mechanism design have made it possible to search over rich, high-dimensional auction formats that would be analytically intractable if we insisted on closed-form characterizations. In combinatorial settings—where bidders may value bundles of items in complex, non-additive ways—this flexibility is especially appealing: the designer can parameterize an allocation rule and payment rule by a neural network, enforce feasibility by construction, and then train the parameters to maximize revenue subject to approximate incentive constraints. The empirical success of this paradigm has been widely documented, and it has opened a practical route to deploying learned mechanisms in applications ranging from procurement and ad allocation to logistics and resource scheduling.

At the same time, the economic logic underlying incentive compatibility is inherently out-of-distribution: a mechanism is safe only if it remains hard to manipulate under the valuations that actually arise at deployment, and under the misreport strategies bidders actually discover. These are not merely technical caveats. In real markets, the distribution of bidder types shifts for mundane reasons (seasonality, entry and exit, new product categories, regulatory changes, macroeconomic shocks), and bidders are strategic learners who adapt to the rules they face. A mechanism that is "approximately truthful" for yesterday's population, evaluated against a weak deviation search, can become substantially exploitable for tomorrow's population once sophisticated bidding software is trained against it.

Two distinct gaps drive this vulnerability. The first is *distribution shift*. Most learning-based mechanism design pipelines optimize expected performance under an empirical training distribution. Even when the objective includes a regret penalty, the guarantee is fundamentally local: low expected regret on the training distribution does not imply low regret under a different distribution that is close in a semantic sense but far in statistical distance. This is not a pathology unique to auctions; it is the standard generalization problem, aggravated by the fact that regret depends on a *maximization* over deviations and therefore can spike sharply when the type profile moves into a region where incentive constraints are tight. In combinatorial domains, where valuations live in a space of dimension exponential in the number of items, it is easy for the training sample to underrepresent precisely those corners of the type space where a learned allocation rule creates profitable misreports.

The second gap is *strategic search at deployment*. Regret in these systems is typically estimated using a deviation oracle: for each sampled valuation profile, one searches over misreports to approximate a best response. This is a computational necessity, but it introduces an "evaluation mismatch" that is economically consequential. If the oracle is imperfect during training (few gradient steps, limited restarts, restricted misreport class), then the

learned mechanism is incentivized to exploit precisely that weakness: it can appear nearly DSIC under the training-time oracle while admitting profitable deviations that a stronger search procedure would uncover. At deployment, however, bidders may effectively run a stronger oracle—either because they invest more computation, because they share strategies, or because third-party tools automate best-response search. In that sense, the mechanism is being tested against an adversary that is both *shifted* (the type distribution changes) and *strengthened* (the deviation search improves).

These two forces interact. Distribution shift can move probability mass toward profiles where the mechanism is more sensitive to misreports, and a stronger deviation search can then exploit that sensitivity. The practical implication is that in-distribution regret guarantees, while valuable, are not a sufficient safety certificate for deployment in strategic environments. From a policy perspective, this matters because learned mechanisms may be deployed precisely in settings where the designer cannot perfectly control participation, information, or external tooling. Robustness, in the economic sense, should therefore be understood as performance stability not only to stochastic variation in types but also to plausible changes in the environment that influence strategic behavior.

We respond to these concerns by importing a familiar idea from robust decision-making into differentiable mechanism design: rather than optimizing expected revenue and regret under a single estimated distribution, we train against an *ambiguity set* of nearby distributions. Conceptually, this asks the mechanism to perform well not just on the observed sample, but also on distributions that could have generated similar samples and on distributions that represent plausible deployment shifts. The key modeling choice is how to measure "nearness" between distributions. We focus on Wasserstein neighborhoods because they align with a transportation-cost interpretation: a small radius corresponds to perturbing valuations by a limited amount under a chosen metric on the type space. This is attractive in combinatorial auctions, where we often have domain knowledge about which perturbations are plausible (e.g., correlated changes across bundles, or larger variability for some items than others), and such knowledge can be encoded in the metric.

Our robust perspective also clarifies the role of smoothness and architecture. When the mechanism maps reports to allocations and payments in a way that is well-behaved (in a Lipschitz sense), then regret as a function of the valuation profile cannot change too abruptly. This regularity links distributional robustness to incentive robustness: if regret is smooth, then controlling it on a training distribution within a Wasserstein ball translates into control on nearby deployment distributions. Conversely, if the learned mechanism is highly non-smooth, a small shift in types can lead to large incentive violations, and robustness demands either architectural constraints or explicit regularization. The robust objective thus serves as a diagnostic and a design principle: it rewards mechanisms whose performance is stable

under economically meaningful perturbations.

Positioning relative to RegretNet. RegretNet popularized the idea of training neural mechanisms with a regret penalty, producing mechanisms that are approximately DSIC on the training distribution and empirically competitive in revenue. We view our approach as complementary rather than competing. RegretNet's core insight is that incentive constraints can be relaxed into a differentiable penalty, making large-scale training feasible. Our contribution is to address the next-order question: what happens when the distribution used to form that penalty is not the one encountered at deployment, and when the effective deviation oracle improves? Robustification does not replace regret penalties; it changes the *distributional lens* through which those penalties are evaluated, and it yields explicit tradeoffs between conservatism (robust low regret) and revenue.

Positioning relative to CAFormer/CANet-style architectures. Recent work on transformer-based or attention-based architectures for combinatorial auctions (often motivated by scalability in the number of items and by capturing structure in bidder valuations) improves the expressive power and computational properties of learned allocation and payment rules. Our robust training principle is architecture-agnostic: it can be layered onto these designs because it operates at the level of the training objective and the evaluation of regret and revenue under distributional perturbations. Indeed, richer architectures can be a double-edged sword: they may fit complex allocation patterns that increase revenue, but they may also introduce sharper non-linearities that magnify sensitivity to shift. Robust training provides a disciplined way to manage this tension.

Contributions and limitations. We make three high-level contributions. First, we articulate the economic failure modes of in-distribution regret guarantees in the presence of distribution shift and stronger strategic search, emphasizing that both are natural in real deployments. Second, we propose a Wasserstein-robust training objective that hedges against such shifts and that admits a tractable upper bound under standard regularity, turning an intractable worst-case distributional problem into a computable surrogate. Third, we provide a generalization perspective: robust training yields deployment-time guarantees that degrade gracefully with sample size and with the complexity of the mechanism class, making explicit the statistical price of robustness.

Our approach has limitations that are important for interpretation. Robustness is only as meaningful as the metric that defines the Wasserstein neighborhood; choosing it requires domain knowledge, and a poor choice can either over-regularize (sacrificing revenue unnecessarily) or under-protect

(missing the relevant shifts). Moreover, robust training cannot eliminate the fundamental computational difficulty of regret estimation; it mitigates the consequences of shift, but if the deviation oracle is systematically weak, additional tooling (stronger best-response search, adversarial training, or certified bounds) may be required. Finally, robustness entails a real economic tradeoff: by insuring against adverse shifts, we may forgo some surplus extraction in favorable regimes. The goal is not to avoid this tradeoff, but to quantify it and to choose it deliberately in light of deployment risk.

In sum, the lesson we draw is that learned mechanisms should be evaluated and trained as *policies under uncertainty* in a strategic environment. The robust framework we develop is meant to illuminate, and operationalize, the central tension: mechanisms that aggressively optimize revenue on the observed distribution can be fragile, while mechanisms that are stable to shift and strategic adaptation may need to be more conservative. The remainder of the paper formalizes this tradeoff and shows how to incorporate it into differentiable mechanism design pipelines.

2 Baseline: Differentiable Mechanism Design with Regret Penalties

We begin from the now-standard template for *differentiable* mechanism design in combinatorial auctions: we parameterize an allocation rule and a payment rule by a neural network (or other differentiable function class), enforce feasibility and individual rationality by construction, and then train the parameters to optimize a revenue–incentives tradeoff measured through regret. This section fixes notation and clarifies what, precisely, is certified by the usual regret-based training objective.

Environment and reports. There are n bidders and m heterogeneous items. Let $M = \{1, \dots, m\}$ denote items and let $K = 2^M \setminus \{\emptyset\}$ denote the set of non-empty bundles; thus $|K| = k = 2^m - 1$. Bidder i has a combinatorial valuation vector $v_i = (v_{iS})_{S \in K} \in \mathcal{V}_i \subset [0, V_{\max}]^k$, where $\mathcal{V}_i$ is compact. A report profile is $b = (b_1, \dots, b_n) \in \mathcal{V} := \prod_i \mathcal{V}_i$, and in a direct-revelation analysis we take the message space to coincide with the valuation space, $b_i \in \mathcal{V}_i$.

A mechanism $w \in \mathcal{W}$ specifies an allocation rule g_w and a payment rule p_w. Since the differentiable literature typically works with fractional or randomized allocations, we represent outcomes by bundle allocation probabilities $z = g_w(b) \in [0,1]^{n \times k}$, where $z_{iS}(b; w)$ is the probability (or fraction) with which bidder i receives bundle S. Payments are $p = p_w(b) \in \mathbb{R}^n_+$.

Feasibility as a differentiable constraint. The combinatorial feasibility constraints are those of a packing problem: each bidder receives at most one

bundle and each item is allocated at most once in expectation. Writing them explicitly,

$$\sum_{S\in K} z_{iS}(b;w) \le 1 \quad \forall i, \qquad \sum_{i\in N}\sum_{S\in K:\, j\in S} z_{iS}(b;w) \le 1 \quad \forall j \in M, \qquad 0 \le z_{iS}(b;w) \le 1. \tag{1}$$

In practice, these constraints are enforced either by a specialized output layer (e.g., a differentiable projection onto a polytope, a Sinkhorn-style normalization in a structured representation, or a relaxation that is exact at the vertices) or by parametrizing g_w so that (1) holds for all b by construction. The key modeling point is that the training objective must be differentiable in w, and hence feasibility enforcement is typically embedded into the computational graph rather than imposed via a non-differentiable solver.

Utilities and ex-post individual rationality. Given true valuations v_i and report profile b, bidder i's quasi-linear utility under mechanism w is

$$u_i(v_i; b; w) = \sum_{S\in K} v_{iS}\, z_{iS}(b;w) - p_i(b;w). \tag{2}$$

The seller's revenue at a truthful profile v is $\mathrm{rev}(w;v) = \sum_{i=1}^{n} p_i(v;w)$.

A common architectural device is to enforce ex-post individual rationality (IR) for truthful reports. One sufficient (and widely used) approach is to predict an "unnormalized" payment factor $\tilde{p}_i(b;w) \in [0,1]$ and then set

$$p_i(b;w) = \tilde{p}_i(b;w) \cdot \Big(\sum_{S\in K} b_{iS}\, z_{iS}(b;w) \Big), \tag{3}$$

which guarantees $p_i(v;w) \le \sum_S v_{iS} z_{iS}(v;w)$ at truthful reporting and hence $u_i(v_i; v; w) \ge 0$ pointwise. More generally, one can build IR into the mechanism by ensuring that payments never exceed reported value for the allocated lottery. This kind of "hard" IR is attractive because it removes a set of constraints from the optimization problem and focuses attention on incentives and revenue.

Regret as an incentive-violation metric. Because exact DSIC is typically out of reach for expressive neural mechanism classes, the baseline differentiable approach replaces incentive constraints with a regret penalty. For a fixed valuation profile v and bidder i, the *ex-post regret* under mechanism w is defined as

$$\mathrm{rgt}_i(w;v) = \max_{v_i' \in \mathcal{V}_i} \Big\{ u_i\big(v_i; (v_i', v_{-i}); w\big) - u_i\big(v_i; v; w\big) \Big\}, \tag{4}$$

and the average regret at profile v is $\mathrm{rgt}(w;v) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{rgt}_i(w;v)$. Regret provides a quantitative relaxation of DSIC: if $\mathrm{rgt}_i(w;v) = 0$ for all i and all

v, then truth-telling is a dominant strategy; if regret is small, then profitable deviations exist only up to a small additive gain in utility.

In the learning pipeline, we are typically interested in *expected* regret under some distribution over valuation profiles. Let $P_\star$ denote the (unknown) distribution of types that generates training data, and let $\hat{P}_L$ be the empirical distribution on L i.i.d. samples $\{v^{(\ell)}\}_{\ell=1}^{L}$. The in-sample estimate of expected regret is

$$\widehat{\mathrm{ER}}(w) = \mathbb{E}_{v\sim\hat{P}_L}[\mathrm{rgt}(w;v)] = \frac{1}{L}\sum_{\ell=1}^{L}\mathrm{rgt}\big(w;v^{(\ell)}\big), \tag{5}$$

and analogously the in-sample revenue is $\widehat{\mathrm{Rev}}(w) = \mathbb{E}_{\hat{P}_L}[\mathrm{rev}(w;v)]$.

Training objective and the role of the deviation oracle. The baseline optimization problem is a Lagrangian relaxation that trades off revenue against expected regret:

$$\min_{w\in\mathcal{W}} \ \mathbb{E}_{v\sim\hat{P}_L}\big[-\mathrm{rev}(w;v) + \lambda\,\mathrm{rgt}(w;v)\big], \tag{6}$$

for some $\lambda > 0$. This objective is differentiable in w except for the maximization in (4). The standard remedy is to approximate the maximization with a *deviation oracle*: for each bidder i and sampled profile v, one performs a (typically gradient-based) inner-loop search over $v_i' \in \mathcal{V}_i$ to find a high-utility misreport against v_{-i}. Denoting the oracle's output by $\tilde{v}_i'(v;w)$, the practical regret term becomes

$$\widetilde{\mathrm{rgt}}_i(w;v) = u_i\big(v_i;(\tilde{v}_i'(v;w),v_{-i});w\big) - u_i(v_i;v;w),$$

and training proceeds by backpropagation through the outer objective using these approximate regrets (often with the inner-loop treated as a stop-gradient operation, or using implicit differentiation if one seeks more faithful gradients). The economic interpretation is straightforward: we train the mechanism against a particular computational model of bidder deviations, namely the one represented by the oracle.

Where the baseline guarantees stop. The baseline formulation yields a clear but limited certificate. If training succeeds in driving $\widehat{\mathrm{ER}}(w) \le \varepsilon$, then the learned mechanism is *approximately DSIC in expectation under the empirical training distribution*, up to the gap between true regret (4) and the oracle-based approximation used in optimization. With additional uniform convergence arguments (as in prior RegretNet analyses), one can sometimes translate $\widehat{\mathrm{ER}}(w)$ into a bound on $\mathbb{E}_{v\sim P_\star}[\mathrm{rgt}(w;v)]$ that improves with L and worsens with the complexity of the mechanism class.

However, two distinctions are crucial for deployment. First, expected regret under $P_\star$ (or $\hat{P}_L$) is not a pointwise guarantee: even if $\mathbb{E}_{P_\star}[\mathrm{rgt}(w;v)]$ is

small, the mechanism may admit large incentive violations on low-probability regions of the type space, precisely because regret is defined by a supremum over deviations and can spike on "knife-edge" profiles. Second, the guarantee is distribution-specific: it is anchored to the population that generated the training sample. If the deployment environment induces a shifted distribution of valuations, the baseline objective (6) provides no direct control of $\mathbb{E}_{v\sim Q}[\mathrm{rgt}(w;v)]$ for $Q \neq P_\star$, even when Q is close to $P_\star$ in an economically natural sense. These observations motivate moving from in-distribution training criteria to explicitly *robust* notions of regret and revenue under distribution shift, to which we turn next.

3 A Clean Model of Distribution Shift: Ambiguity Sets and Robust Performance

The baseline objective in (6) treats the empirical training distribution $\hat{P}_L$ as the relevant description of the environment. In many platform settings, that is an incomplete representation of what the mechanism will face. The distribution of bidder types can drift because the platform enters a new geography, changes eligibility rules, modifies product bundles, alters recommendation or matching policies, or simply because the population of bidders and their outside options evolve over time. Even absent any deliberate policy change, day-to-day composition effects (who shows up) can induce systematic shifts in the joint distribution of valuations. Our goal, therefore, is to articulate a model of *deployment uncertainty* that is both economically interpretable and mathematically tractable.

Deployment distributions and the designer's uncertainty. We let Q denote an arbitrary distribution over valuation profiles $v \in \mathcal{V}$ that may govern online instances at deployment time. The designer does not know Q ex ante. What we assume, instead, is that Q is *close* to the distribution that generated the data used for training—either the true population distribution $P_\star$ or its empirical approximation $\hat{P}_L$. This yields the standard robust-design posture: rather than optimizing performance under a single reference distribution, we optimize (or certify) performance uniformly over a neighborhood of plausible deployment distributions.

The key modeling choice is how to formalize "closeness" between distributions over high-dimensional combinatorial valuations. Two families of neighborhoods are particularly natural in auction platforms: Wasserstein (optimal-transport) balls, which measure how much one must *move probability mass* in type space, and f-divergence balls, which measure how much one must *reweight* probability mass already present under a reference distribution.

Wasserstein ambiguity sets: shift as bounded transportation. Fix a metric $d(\cdot,\cdot)$ on $\mathcal{V}$ (for instance, an ℓ_1 metric on the concatenated valuation vectors, possibly with bundle-dependent weights). The 1-Wasserstein distance between distributions Q and P on $(\mathcal{V}, d)$ is

$$W_1(Q,P) \;=\; \inf_{\pi\in\Pi(Q,P)} \; \mathbb{E}_{(v,\tilde{v})\sim\pi}\big[d(v,\tilde{v})\big],$$

where $\Pi(Q,P)$ is the set of couplings with marginals Q and P. Given a radius $\delta \geq 0$, we define the Wasserstein ambiguity set around a reference distribution P as

$$\mathcal{U}_\delta(P) \;:=\; \big\{Q:\; W_1(Q,P) \leq \delta\big\}.$$

Economically, δ is a *shift budget*: on average, probability mass can be transported by at most δ units of type-distance. This is particularly appealing in our setting for three reasons. First, Wasserstein neighborhoods allow for *support shift*. If bidders at deployment have valuations in regions of $\mathcal{V}$ that were rare (or even unseen) in the training sample, Wasserstein distance can still be small provided those regions are not far in d from the training support. This matches the practical reality that platforms often extrapolate locally (e.g., slightly larger budgets, slightly different complementarities) rather than encountering entirely unrelated preferences.

Second, Wasserstein ambiguity sets force us to make explicit what changes are "small" through the metric d. In a combinatorial auction, an ℓ_1 metric treats each bundle component symmetrically, but a platform might reasonably downweight large bundles or focus the transportation cost on item-level marginals. This flexibility is a feature rather than a bug: robust guarantees are only meaningful relative to a domain-relevant notion of proximity.

Third, as we show in the next section, Wasserstein balls interact favorably with Lipschitz continuity. When performance measures vary smoothly with valuations (as is typically encouraged by continuous network architectures and bounded output layers), worst-case expectations over $\mathcal{U}_\delta(P)$ admit sharp dual upper bounds. This is precisely the kind of structure that turns robustness from an intractable min–max problem into a regularized empirical objective.

f-divergence ambiguity sets: shift as bounded reweighting. An alternative, widely used in robust statistics and distributionally robust optimization, is to constrain the deployment distribution through an f-divergence. Let $f:\mathbb{R}_+\to\mathbb{R}$ be convex with $f(1)=0$. For distributions Q and P with $Q \ll P$, the f-divergence is

$$D_f(Q\|P) \;:=\; \mathbb{E}_{v\sim P}\left[f\left(\frac{dQ}{dP}(v)\right)\right].$$

Given a radius $\rho \geq 0$, the corresponding ambiguity set is

$$\mathcal{U}_\rho^f(P) \;:=\; \big\{Q :\; D_f(Q\|P) \leq \rho\big\}.$$

Concrete examples include KL divergence (relative entropy), χ^2-divergence, and total variation distance (via an appropriate choice of f). These sets have a natural platform interpretation when we believe deployment differs from training primarily through *composition reweighting* rather than new support: for instance, the same kinds of bidders appear, but their frequencies change because of marketing, seasonality, or selection effects. From an estimation perspective, f-divergence balls are also closely tied to importance weighting: a small $D_f(Q\|P)$ constrains how extreme the likelihood ratio dQ/dP can be on average.

At the same time, there is a substantive limitation that matters in combinatorial valuation spaces. If the reference distribution is empirical, $P = \hat{P}_L$ is discrete with finite support. Then $Q \ll \hat{P}_L$ forces Q to be supported *only* on the observed samples, meaning that f-divergence ambiguity sets do not naturally capture out-of-sample support shift. In contrast, Wasserstein balls around $\hat{P}_L$ can place mass near the samples while still representing genuinely new valuation profiles. For this reason, in settings where extrapolation beyond observed types is operationally plausible, Wasserstein neighborhoods provide a more faithful model of uncertainty.

Robust regret and robust revenue. Given an ambiguity set $\mathcal{U}$ (of either form), we can define robust performance criteria by taking worst-case expectations over $Q \in \mathcal{U}$. Two objects are central for incentives and seller objectives.

First, we define *robust expected regret* (robust regret for short) as

$$\mathrm{RR}_{\mathcal{U}}(w) \;:=\; \sup_{Q\in\mathcal{U}} \; \mathbb{E}_{v\sim Q}\big[\mathrm{rgt}(w;v)\big]. \tag{7}$$

When $\mathcal{U} = \mathcal{U}_\delta(\hat{P}_L)$ we write $\widehat{\mathrm{RR}}_\delta(w)$, consistent with the notation used in the generalization statement in the enclosing scope. Economically, (7) is a uniform-in-distribution incentive certificate: if $\mathrm{RR}_{\mathcal{U}}(w) \leq \varepsilon$, then no deployment environment within the modeled neighborhood can generate expected regret larger than ε. This is the distributional analogue of worst-case (ex-post) incentive constraints, but calibrated to a plausible uncertainty set rather than the entire type space.

Second, we define *robust revenue* as the worst-case expected revenue over $\mathcal{U}$,

$$\mathrm{Rev}_{\mathcal{U}}(w) \;:=\; \inf_{Q\in\mathcal{U}} \; \mathbb{E}_{v\sim Q}\big[\mathrm{rev}(w;v)\big]. \tag{8}$$

The infimum reflects the designer's risk: a mechanism tuned to extract revenue in a narrow region of the training distribution may perform poorly

when the environment shifts. In a platform interpretation, (8) corresponds to a conservative revenue forecast under model uncertainty, and it provides a natural counterpart to robust regret when the platform seeks both incentive reliability and predictable monetization.

Robust objectives and what they protect against. A convenient way to combine incentives and revenue is to robustify the same Lagrangian loss used in baseline training. For a given ambiguity set $\mathcal{U}$, define

$$\mathcal{L}_{\mathcal{U}}(w) \;:=\; \sup_{Q\in\mathcal{U}} \; \mathbb{E}_{v\sim Q}\big[\,-\operatorname{rev}(w;v) + \lambda \operatorname{rgt}(w;v)\big].$$

Minimizing $\mathcal{L}_{\mathcal{U}}(w)$ asks for a mechanism that performs well under the *most adverse* distribution within the neighborhood: adverse both in the sense of low revenue and in the sense of high incentive violations. This captures a practical concern in learned mechanisms: the same off-distribution regions that create large regret are often also regions where payment normalization and allocation policies interact in unexpected ways, producing fragile revenue.

It is important to be clear about what robustness does *not* provide. These criteria do not yield pointwise DSIC guarantees (regret can still be high on individual valuation profiles), nor do they protect against arbitrary strategic models beyond the deviation oracle used to estimate regret. Robustness here is with respect to *type-distribution shift*, holding fixed the mechanism and the definition of regret. Put differently, the ambiguity set formalizes uncertainty about *which valuation profiles arise*, not about *how bidders compute deviations*. We view this as a useful separation: distribution shift is an empirical reality in platforms, while bounded rationality and strategic sophistication are better addressed through the choice and validation of the deviation oracle.

Why these ambiguity sets are natural for platforms. From an institutional perspective, one can read δ (or ρ) as a design knob encoding the platform's tolerance for misspecification. Small radii correspond to stable, well-understood markets; larger radii correspond to rapidly evolving environments or to settings where training data are known to be biased (e.g., early adopters differ from the eventual user base). Because d can be scaled and weighted, the Wasserstein model can also incorporate product-level knowledge: the platform may believe that bidders' singleton values drift substantially while complementarities remain relatively stable, or vice versa, and can encode this in the transportation cost.

Finally, these sets align with operational monitoring. Platforms routinely track shifts in observable features correlated with valuations (entry cohorts, categories, budgets). While we do not observe v directly in deployed sealed-bid settings, the same tools used for drift detection in covariates can motivate

a plausible δ or ρ ex ante. Our contribution is not to pin down the "correct" radius, but to provide a clean framework in which the economic tradeoff is explicit: enlarging the ambiguity set strengthens incentive robustness and revenue predictability under shift, but it forces the mechanism to hedge against a broader range of environments.

In the next section, we show how Wasserstein ambiguity sets lead to particularly tractable dual representations when the loss is Lipschitz, yielding a closed-form robustification term that can be optimized alongside the usual regret penalties.

4 Robust Objectives and Dual Representations

The distributionally robust training problem introduces a seemingly burdensome inner maximization over deployment distributions,

$$\mathcal{L}_\delta(w) \;=\; \sup_{Q:W_1(Q,\hat{P}_L)\le\delta} \mathbb{E}_{v\sim Q}\big[f_w(v)\big], \qquad f_w(v) \equiv -\mathrm{rev}(w;v) + \lambda\,\mathrm{rgt}(w;v).$$

Taken literally, this is an optimization over an infinite-dimensional object (a probability measure), and one might worry that robustness simply replaces a tractable empirical objective with an intractable saddle-point problem. The central practical message of this section is that, under mild smoothness, Wasserstein robustness collapses to a transparent regularization term: worst-case expectations over transportation balls are controlled by the Lipschitz modulus of the loss. This is the sense in which Wasserstein ambiguity sets provide not only an economically meaningful model of shift, but also a computationally convenient one.

Kantorovich–Rubinstein duality and a tractable reformulation. Let $(\mathcal{V}, d)$ be a Polish metric space and let $f_w : \mathcal{V} \to \mathbb{R}$ be measurable and integrable under $\hat{P}_L$. A standard dual representation (see, e.g., the Kantorovich–Rubinstein theorem and its DRO refinements) implies

$$\sup_{Q:W_1(Q,\hat{P}_L)\le\delta} \mathbb{E}_Q[f_w(v)] \;=\; \inf_{\gamma\ge 0}\left\{\gamma\,\delta + \mathbb{E}_{v\sim\hat{P}_L}\Big[\sup_{v'\in\mathcal{V}}\Big(f_w(v') - \gamma\,d(v',v)\Big)\Big]\right\}, \tag{9}$$

under standard regularity ensuring strong duality.[1] The expression (9) is already useful: it converts an optimization over distributions into an optimization over a scalar multiplier γ and a pointwise supremum over types v'. Economically, γ is the shadow price of transportation distance, and the inner supremum computes the best *shifted* type v' that trades off performance

[1] For our purposes it is enough that $\mathcal{V}$ is compact (as assumed) and f_w is bounded/continuous in v; these conditions rule out the pathologies that can create a duality gap.

under f_w against the transportation cost back to an empirical sample point v.

Closed-form robustification under Lipschitz losses. The preceding dual becomes particularly clean when f_w is Lipschitz. Write $\mathrm{Lip}(f_w)$ for the smallest L such that $|f_w(v) - f_w(\tilde{v})| \le L\, d(v, \tilde{v})$ for all $v, \tilde{v} \in \mathcal{V}$. If $\mathrm{Lip}(f_w) \le L_f$, then for any $\gamma \ge L_f$ we have

$$f_w(v') - \gamma d(v', v) \;\le\; f_w(v) + \big(L_f - \gamma\big) d(v', v) \;\le\; f_w(v),$$

so the inner supremum in (9) equals $f_w(v)$ (attained at $v' = v$). Plugging this into (9) yields the bound

$$\sup_{Q : W_1(Q, \hat{P}_L) \le \delta} \mathbb{E}_Q[f_w(v)] \;\le\; \mathbb{E}_{\hat{P}_L}[f_w(v)] + \delta\, L_f. \tag{10}$$

Moreover, when $L_f = \mathrm{Lip}(f_w)$ is known (or when the bound is tight), equality holds and the infimum is achieved at $\gamma = L_f$. Thus, in the Lipschitz regime, the distributionally robust objective becomes the empirical loss plus a linear "robustness tax" proportional to the shift budget δ and the sensitivity of the loss to perturbations in v.

This expression makes the tradeoff particularly transparent. Holding w fixed, the only way the worst-case expectation can be large is if either (i) the empirical mean performance is poor, or (ii) performance is highly sensitive to valuations (large Lipschitz constant), so that small transportation can move probability mass to regions where f_w is worse. In this sense, the Wasserstein model exactly operationalizes a common platform intuition: mechanisms that are locally stable in type space are less brittle to deployment drift.

Applying the bound to the revenue–regret Lagrangian. For our robust Lagrangian loss $f_w(v) = -\mathrm{rev}(w; v) + \lambda\, \mathrm{rgt}(w; v)$, a convenient sufficient condition is that both revenue and regret are Lipschitz in v (under the same metric d). In that case,

$$\mathrm{Lip}(f_w) \;\le\; \mathrm{Lip}(\mathrm{rev}(w; \cdot)) + \lambda\, \mathrm{Lip}(\mathrm{rgt}(w; \cdot)) \;\equiv\; L_{\mathrm{rev}}(w) + \lambda\, L_{\mathrm{rgt}}(w). \tag{11}$$

Substituting (11) into (10) gives an explicit surrogate for the inner supremum:

$$\mathcal{L}_\delta(w) \;\le\; \frac{1}{L} \sum_{\ell=1}^{L} \Big(-\mathrm{rev}(w; v^{(\ell)}) + \lambda\, \mathrm{rgt}(w; v^{(\ell)}) \Big) + \delta \Big(L_{\mathrm{rev}}(w) + \lambda\, L_{\mathrm{rgt}}(w) \Big).$$

This bound already clarifies how robustness "tilts" training. Even if the empirical objective strongly favors aggressive revenue extraction, the term $\delta\, L_{\mathrm{rev}}(w)$ penalizes mechanisms whose revenue is highly sensitive to valuation

perturbations, and the term $\delta\,\lambda\,L_{\mathrm{rgt}}(w)$ analogously penalizes mechanisms whose incentive violations spike under small shifts.

A practical subtlety is that the usefulness of this robustification depends on whether the Lipschitz constants are treated as mechanism-dependent quantities. If we merely impose a uniform bound $L_{\mathrm{rev}}(w) \le \overline{L}_{\mathrm{rev}}$ and $L_{\mathrm{rgt}}(w) \le \overline{L}_{\mathrm{rgt}}$ for all $w \in \mathcal{W}$, then the additive term $\delta(\overline{L}_{\mathrm{rev}} + \lambda\overline{L}_{\mathrm{rgt}})$ is constant in w and does not affect the optimizer. To obtain a nontrivial robustness–performance tradeoff in training, we therefore either (i) work with sharper, architecture-driven upper bounds that vary with w (e.g., via spectral norms of network layers), or (ii) use the more general dual form (9) and approximate the inner supremum over v' numerically, which effectively learns the "adversarial" perturbations that matter for a given w.

When do we need numerical inner maximization? There are two main cases where the clean Lipschitz robustification is not enough.

First, the relevant loss may fail to be globally Lipschitz (or we may be unwilling to certify a global bound that is not overly conservative). This can occur if the allocation rule is effectively discontinuous in reports (for instance, if one uses hard argmax operations without smoothing), or if the regret estimate uses a non-smooth deviation oracle. In such cases, the inequality (10) can still be written with a local or empirical Lipschitz proxy, but tightness is not guaranteed. A common response is to revert to (9): for each sampled $v^{(\ell)}$, one approximately solves

$$\sup_{v' \in \mathcal{V}} \Big(f_w(v') - \gamma\, d(v', v^{(\ell)}) \Big)$$

by gradient-based ascent in type space (projected onto $\mathcal{V}$), thereby directly constructing worst-case perturbations consistent with the transportation budget. This "adversarial types" computation is more expensive than adding a closed-form regularizer, but it remains a finite-dimensional optimization and aligns well with modern automatic differentiation.

Second, if one uses ambiguity sets other than Wasserstein balls—most notably f-divergence balls—the inner problem typically takes a different dual form and rarely collapses to an empirical mean plus a simple Lipschitz term. For example, for KL divergence one obtains an entropic risk functional,

$$\sup_{Q : D_{\mathrm{KL}}(Q\|P) \le \rho} \mathbb{E}_Q[f_w(v)] \;=\; \inf_{\eta > 0} \Big\{ \eta\,\rho + \eta \log\Big(\mathbb{E}_{v\sim P}\big[e^{f_w(v)/\eta} \big] \Big) \Big\},$$

and when $P = \hat{P}_L$ this becomes a one-dimensional optimization in η involving a log-sum-exp over samples. More generally, f-divergences lead to Fenchel–Legendre conjugate expressions that entail either a nontrivial multiplier search or an explicit reweighting of sample points. These formulations are computationally tractable, but they implement a different economic notion of robustness: they protect primarily against *reweighting* of observed

types rather than *support shift*. In our combinatorial setting, where unseen-but-nearby valuation profiles are a realistic concern, this distinction is consequential.

What the dual view buys us. The key benefit of the Wasserstein duality perspective is that it separates two modeling tasks. The ambiguity radius δ encodes the platform's tolerance for distributional drift, while the Lipschitz modulus of performance encodes the mechanism's intrinsic stability to such drift. When stability can be certified (or encouraged) through architectural constraints and capacity control, robust design becomes a disciplined form of regularization rather than an opaque adversarial optimization. The next section develops concrete sufficient conditions—bounded valuation domains, Lipschitz network components, and complexity control for the mechanism class—under which regret and revenue inherit the smoothness needed for these guarantees to be meaningful and non-vacuous.

5 Lipschitz and Capacity Control for Auction Networks

The dual viewpoint in Section 4 tells us that Wasserstein robustness becomes informative precisely when the mechanism's induced loss varies smoothly with types. We therefore need a set of *verifiable* sufficient conditions—stated at the level of the auction network architecture and parameter magnitudes—under which (i) the allocation and payment rules are Lipschitz in reports, and hence (ii) revenue and regret inherit Lipschitz continuity in the valuation profile. In parallel, to turn empirical robust regret into deployment-time guarantees, we need *capacity control*: a way to upper bound the statistical complexity of the mechanism class, along the lines of the RegretNet generalization arguments but adapted to our robust objective.

Bounded type space and the role of normalization. We work on a compact valuation domain $\mathcal{V} = \prod_i \mathcal{V}_i \subset [0, V_{\max}]^{nk}$, which is both economically natural (valuations are finite) and technically convenient (it prevents pathologies in duality and uniform convergence). The metric $d(\cdot, \cdot)$ matters: when d is an ℓ_1-type metric on the concatenated valuation vector, Lipschitz constants scale with the ambient dimension nk unless we normalize. In practice and in our bounds, we interpret d as either an *average* ℓ_1 metric (dividing by nk) or a weighted metric that downweights large bundles; this keeps L_{rev} and L_{rgt} comparable across settings with different m.

Lipschitz allocation networks via smooth relaxations and bounded weights. Consider an allocation network that maps reports $b \in \mathcal{V}$ to a

(possibly fractional) allocation tensor $z = g_w(b) \in [0,1]^{n\times k}$ satisfying feasibility. Many modern architectures implement feasibility by producing *scores* $s_{iS}(b)$ and then applying a differentiable normalization that enforces item and bidder constraints, e.g., softmax layers combined with iterative scaling, Sinkhorn-type operators, or differentiable projections onto a packing polytope.

A general sufficient condition is as follows. Suppose g_w can be written as a composition

$$g_w \;=\; \Pi \circ \phi_w,$$

where $\phi_w : \mathbb{R}^{nk} \to \mathbb{R}^{n\times k}$ is a feedforward network with activations that are κ-Lipschitz (ReLU is 1-Lipschitz; tanh is 1-Lipschitz), and Π is a post-processing map that enforces feasibility and is non-expansive under the relevant norm (for instance, Euclidean projection is 1-Lipschitz in ℓ_2; softmax is Lipschitz with a constant controlled by temperature). If each linear layer ℓ of ϕ_w has operator norm at most $\|W_\ell\|_{\mathrm{op}} \le B_\ell$, then the standard composition bound yields

$$\mathrm{Lip}(\phi_w) \;\le\; \kappa^D \prod_{\ell=1}^{D} B_\ell, \qquad \mathrm{Lip}(g_w) \;\le\; \mathrm{Lip}(\Pi)\,\mathrm{Lip}(\phi_w),$$

where D is depth. Thus, bounded spectral norms (or any operator-norm surrogate) and Lipschitz activations give an explicit Lipschitz bound for the allocation rule. This is not merely a proof device: spectral normalization, weight clipping, and gradient penalties are practical training tools that directly target these constants and thereby reduce the Wasserstein "robustness tax" from Section 4.

Lipschitz payments: direct networks or payment identities. Payment rules in learned mechanisms are typically implemented either (a) by a separate payment network $p_w(b)$ with nonnegativity enforced by softplus/reLU, or (b) via a differentiable approximation to a payment identity that guarantees incentive properties in restricted settings. For our purposes, we only require that the realized payment map be Lipschitz in b and bounded on $\mathcal{V}$.

A convenient sufficient condition mirrors the allocation case. If $p_w = \psi_w$ is a feedforward network with κ-Lipschitz activations and layer operator norms bounded by $\widetilde{B}_\ell$, then

$$\mathrm{Lip}(p_w) \;\le\; \kappa^{\widetilde{D}} \prod_{\ell=1}^{\widetilde{D}} \widetilde{B}_\ell.$$

If, additionally, we enforce bounded outputs (e.g., by a final sigmoid scaled by $p_{\max}$ or by explicit clipping), then $p_w(b) \in [0, p_{\max}]^n$ uniformly. Bounded

payments are economically interpretable as a no-excessive-charges policy and are technically useful because they make both revenue and regret uniformly bounded, a prerequisite for clean complexity bounds.

From Lipschitz (g_w, p_w) to Lipschitz revenue and regret. Given Lipschitzness of g_w and p_w in reports, revenue Lipschitzness is immediate:

$$\big|\mathrm{rev}(w;v) - \mathrm{rev}(w;\tilde{v})\big| = \Big|\sum_{i=1}^{n} p_i(v;w) - \sum_{i=1}^{n} p_i(\tilde{v};w)\Big| \;\le\; n\,\mathrm{Lip}(p_w)\,d(v,\tilde{v}),$$

up to the norm equivalence implicit in d. The more delicate object is regret, because it is defined through an inner maximization over misreports. Here compactness of $\mathcal{V}_i$ and Lipschitzness of utilities jointly in (v, b) do the work. Writing bidder i's utility as

$$u_i(v_i; b; w) = \sum_{S \in K} v_{iS} z_{iS}(b; w) - p_i(b; w),$$

we see that for fixed b the dependence on v_i is linear, while for fixed v_i the dependence on b is Lipschitz through g_w and p_w. Under bounded valuations ($v_{iS} \in [0, V_{\max}]$) and bounded feasibility ($\sum_S z_{iS} \le 1$), utility differences can be bounded as

$$|u_i(v_i;b;w) - u_i(\tilde{v}_i;\tilde{b};w)| \;\le\; V_{\max}\,\|g_w(b) - g_w(\tilde{b})\|_1 + \|p_w(b) - p_w(\tilde{b})\|_1 + \|v_i - \tilde{v}_i\|_1,$$

again up to normalization constants. Taking the supremum over $v_i' \in \mathcal{V}_i$ preserves Lipschitz continuity on compact domains, yielding precisely the type of statement summarized in Proposition 2: there exists an explicit $L_{\mathrm{rgt}}(w)$ depending on $\mathrm{Lip}(g_w)$, $\mathrm{Lip}(p_w)$, $V_{\max}$, and the choice of d such that

$$|\mathrm{rgt}(w;v) - \mathrm{rgt}(w;\tilde{v})| \le L_{\mathrm{rgt}}(w)\,d(v,\tilde{v}).$$

Operationally, this tells us what architectural features matter for robustness: anything that creates sharp kinks in allocation or payments (hard argmax, discontinuous tie-breaking, unregularized large weights) can make $L_{\mathrm{rgt}}(w)$ large, thereby increasing the required conservatism for a given shift budget δ.

Capacity control via covering numbers: adapting the RegretNet logic. Smoothness alone is not enough: we also need to control how much a learned mechanism can overfit the sample. The typical route, used in RegretNet and related work, is to upper bound the uniform deviation between population and empirical expectations by a complexity term involving covering numbers or Rademacher complexity.

For our mechanism class $\mathcal{M} = \{(g_w, p_w) : w \in \mathcal{W}\}$, we consider the induced scalar function classes

$$\mathcal{F}_{\mathrm{rev}} = \{v \mapsto \mathrm{rev}(w;v) : w \in \mathcal{W}\}, \qquad \mathcal{F}_{\mathrm{rgt}} = \{v \mapsto \mathrm{rgt}(w;v) : w \in \mathcal{W}\}.$$

If we restrict $\mathcal{W}$ so that all layer operator norms are bounded (spectral norm bounds) and outputs are bounded, standard neural-network covering arguments imply that for any $\epsilon > 0$,

$$\log N(\mathcal{F}_{\text{rev}}, \epsilon) \;\lesssim\; P \log\Big(\frac{C}{\epsilon}\Big),$$

where P is the number of parameters and C depends on the norm bounds, depth, and activation Lipschitz constants. An analogous bound holds for $\mathcal{F}_{\text{rgt}}$ provided we treat the regret computation as the supremum of a family of Lipschitz utility differences indexed by $v_i' \in \mathcal{V}_i$. Compactness of $\mathcal{V}_i$ allows us to discretize the deviation space with an ϵ-net, turning the supremum into a maximum over finitely many deviations at the cost of an additional covering factor for $\mathcal{V}_i$. This is exactly where bounded valuation domains enter the statistical story: without compactness, the deviation class would be too rich to control uniformly.

Two caveats are worth emphasizing. First, in implementations regret is typically *approximated* by a deviation oracle (e.g., projected gradient ascent over misreports). Our guarantees can incorporate this by adding an optimization error term, provided the oracle is uniformly accurate on $\mathcal{V}$; otherwise, one should interpret the result as a guarantee for *estimated* regret. Second, complexity control is only meaningful if we actually enforce norm bounds during training; absent such constraints, the effective capacity of the network class can be large enough that the generalization term is vacuous.

Implication for robust training. Putting these pieces together, we obtain a disciplined recipe: (i) choose architectures for g_w and p_w that are differentiable and feasibly normalized; (ii) impose spectral or weight-norm constraints (or penalties) to bound Lipschitz constants; and (iii) bound outputs to ensure uniform integrability. The resulting mechanism class has both the smoothness needed for Wasserstein dual control and the capacity control needed for high-probability out-of-sample guarantees. The next section states the main theorems that translate these primitives into deployment-time regret and revenue bounds under distribution shift, and draws out their interpretation as robust approximate DSIC.

6 Main Theorems: Deployment-Time Regret and Revenue Guarantees Under Shift

We now formalize the sense in which robust training delivers *deployment-time* incentive and revenue guarantees when the valuation distribution shifts away from the training environment. The key conceptual move is to separate (i) what we can *optimize* from data, namely a robust empirical objective over the Wasserstein ball around $\hat{P}_L$, from (ii) what we ultimately *care about*,

namely performance over an *unknown* deployment distribution Q that is only assumed to satisfy a shift budget $W_1(Q, P_\star) \le \delta$. Our results therefore take the form of high-probability inequalities that hold *uniformly* over a mechanism class and *simultaneously* for all admissible deployment distributions.

To streamline notation, define the population robust regret functional and its empirical analogue:

$$\mathrm{RR}_\delta(w) \;:=\; \sup_{Q: W_1(Q,P_\star)\le\delta} \mathbb{E}_{v\sim Q}\big[\mathrm{rgt}(w;v)\big], \qquad \widehat{\mathrm{RR}}_\delta(w) \;:=\; \sup_{Q: W_1(Q,\hat{P}_L)\le\delta} \mathbb{E}_{v\sim Q}\big[\mathrm{rgt}(w;v)\big].$$

We similarly define robust revenue functionals

$$\mathrm{REV}_\delta(w) \;:=\; \inf_{Q: W_1(Q,P_\star)\le\delta} \mathbb{E}_{v\sim Q}\big[\mathrm{rev}(w;v)\big], \qquad \widehat{\mathrm{REV}}_\delta(w) \;:=\; \inf_{Q: W_1(Q,\hat{P}_L)\le\delta} \mathbb{E}_{v\sim Q}\big[\mathrm{rev}(w;v)\big].$$

The supremum in RR_δ reflects worst-case incentive violations, while the infimum in REV_δ reflects worst-case revenue erosion. Both are natural if we interpret δ as an explicit policy choice describing how adversarial (or simply how different) deployment may be relative to training.

Theorem 1: robust generalization of expected regret. Our first theorem states that robust regret computed on the sample controls robust regret at deployment, up to a statistical complexity term of the familiar $\tilde{\mathcal{O}}(1/\sqrt{L})$ form.

Theorem 1 (Deployment-time robust regret bound). *Assume $\mathcal{V} \subset [0, V_{\max}]^{nk}$ is compact, the regret map $v \mapsto \mathrm{rgt}(w;v)$ is L_{rgt}-Lipschitz under d for all $w \in \mathcal{W}$, and $\mathrm{rgt}(w;v)$ is uniformly bounded. Let $S \sim P_\star^L$ and fix $\beta \in (0,1)$. Then, with probability at least $1-\beta$ over S, we have for all $w \in \mathcal{W}$*

$$\sup_{Q: W_1(Q,P_\star)\le\delta} \mathbb{E}_{v\sim Q}\big[\mathrm{rgt}(w;v)\big] \;\le\; \widehat{\mathrm{RR}}_\delta(w) \;+\; \mathcal{G}_L(\mathcal{M},\beta),$$

where $\mathcal{G}_L(\mathcal{M},\beta)$ is a uniform convergence term that can be taken of the form

$$\mathcal{G}_L(\mathcal{M},\beta) \;=\; \mathcal{O}\left(\sqrt{\frac{\log N(\mathcal{M},\epsilon) + \log(1/\beta)}{L}}\right),$$

with optimization over $\epsilon > 0$ as usual.

While the exact complexity term depends on the metric used to cover the induced function class and on how regret is approximated, the economic meaning is invariant: if $\widehat{\mathrm{RR}}_\delta(w)$ is small, then *no* deployment distribution within Wasserstein radius δ can generate large *expected* ex-post regret, except with small probability due to sampling noise. In this sense, robust training protects not only against overfitting to S but also against misspecification of the environment.

Interpretation: robust approximate DSIC. Theorem 1 implies a clean approximate incentive statement. Suppose we train a mechanism $\hat{w}$ such that $\widehat{\mathrm{RR}}_\delta(\hat{w}) \le \varepsilon$ (or, more strongly, a bidderwise variant $\widehat{\mathrm{RR}}_{\delta,i}(\hat{w}) \le \varepsilon$ for all i). Then with the same high probability,

$$\sup_{Q:W_1(Q,P_\star)\le\delta} \mathbb{E}_{v\sim Q}\big[\mathrm{rgt}(\hat{w};v)\big] \;\le\; \varepsilon + \mathcal{G}_L(\mathcal{M},\beta).$$

In particular, for any such deployment distribution Q, truthful reporting forms an $(\varepsilon+\mathcal{G}_L)$-*approximate dominant-strategy equilibrium in expectation*: each bidder can improve her expected utility by at most $\varepsilon + \mathcal{G}_L$ by deviating unilaterally from truthful play. This is the precise sense in which Wasserstein-robust training yields *robust approximate DSIC*. Practically, it means that incentive violations cannot suddenly become large merely because the market shifts within the modeled ambiguity set.

Theorem 2: robust revenue guarantees and the robustness tax. We now state the analogous result for revenue. Here we emphasize the *direction* of the guarantee: robustness protects against the *lower tail* of revenue, which is typically the relevant concern for a seller facing uncertain demand.

Theorem 2 (Deployment-time robust revenue bound). *Assume $v \mapsto \mathrm{rev}(w;v)$ is L_{rev}-Lipschitz under d for all $w \in \mathcal{W}$ and uniformly bounded. Then, with probability at least $1-\beta$ over $S \sim P_\star^L$, for all $w \in \mathcal{W}$,*

$$\inf_{Q:W_1(Q,P_\star)\le\delta} \mathbb{E}_{v\sim Q}\big[\mathrm{rev}(w;v)\big] \;\ge\; \widehat{\mathrm{REV}}_\delta(w) \;-\; \mathcal{H}_L(\mathcal{M},\beta),$$

where $\mathcal{H}_L(\mathcal{M},\beta)$ is a uniform convergence term analogous to $\mathcal{G}_L(\mathcal{M},\beta)$.

Combining this with the dual bound from Section 4 yields an immediately interpretable "robustness tax": for Lipschitz revenue, the worst-case revenue over a δ-ball cannot exceed the in-distribution revenue, and it can be lower by an amount proportional to δL_{rev}. This is not a defect of our analysis but an economic reality: if we insist on performing well in environments that are δ-far, we must hedge against types that are unfavorable for revenue.

From regret guarantees to revenue under strategic play. In principle, revenue guarantees are stated under truthful input v; however, the regret bound provides a bridge to strategic behavior. If a mechanism has small regret, then the gap between truthful utility and best-response utility is small, which makes truthful reporting approximately stable. In many applications, this stabilizes revenue as well: if bidders cannot profitably manipulate the mechanism, observed bids will not systematically drift into low-revenue regions induced by misreports. We emphasize the limitation: our regret notion is ex-post and unilateral, so translating it into a full equilibrium statement

(e.g., bounds under Bayes–Nash play) requires additional assumptions on bidder beliefs and learning dynamics. Still, as a design principle, minimizing robust regret is a direct way to prevent revenue collapse that arises *because* of strategic behavior amplified by distribution shift.

Quantile-regret corollaries: controlling tail incentive violations. Expected regret is a natural summary, but platforms often care about *frequency* of large deviations: "in what fraction of auctions is the mechanism meaningfully non-IC?" A simple corollary converts expected-regret control into a quantile bound that is valid uniformly over all Q in the ambiguity set.

Fix $\eta \in (0,1)$ and define the $(1-\eta)$-quantile of regret under Q by

$$q_{1-\eta}(w;Q) \;:=\; \inf\Big\{t \ge 0 :\; \mathbb{P}_{v\sim Q}\big(\mathrm{rgt}(w;v) \le t\big) \ge 1-\eta\Big\}.$$

If $\mathbb{E}_Q[\mathrm{rgt}(w;v)] \le \rho$, then Markov's inequality implies

$$\mathbb{P}_{v\sim Q}\big(\mathrm{rgt}(w;v) > t\big) \;\le\; \frac{\rho}{t} \quad\Rightarrow\quad q_{1-\eta}(w;Q) \;\le\; \frac{\rho}{\eta}.$$

Applying Theorem 1 with $\rho = \widehat{\mathrm{RR}}_\delta(w) + \mathcal{G}_L(\mathcal{M},\beta)$ yields, with probability at least $1-\beta$,

$$\sup_{Q:W_1(Q,P_\star)\le\delta} q_{1-\eta}(w;Q) \;\le\; \frac{\widehat{\mathrm{RR}}_\delta(w) + \mathcal{G}_L(\mathcal{M},\beta)}{\eta}.$$

Thus, if we train so that $\widehat{\mathrm{RR}}_\delta(w)$ is small, then for every admissible deployment distribution, at least a $1-\eta$ fraction of auction instances exhibit regret no larger than $(\widehat{\mathrm{RR}}_\delta(w)+\mathcal{G}_L)/\eta$. This is a coarse but transparent tail guarantee, and it highlights a practical lever: if a regulator or platform operator desires that "99% of instances have regret at most τ," one can target $\widehat{\mathrm{RR}}_\delta(w) \lesssim 0.01\,\tau$ (up to generalization).

What these theorems do *not* guarantee. Two caveats deserve attention before we turn to algorithms. First, all statements hinge on controlling Lipschitz constants; without explicit architectural or norm constraints, the constants can be so large that robustness bounds become vacuous. Second, robust regret is only as accurate as our regret estimator: if the deviation oracle fails to find near-best responses, then $\widehat{\mathrm{RR}}_\delta(w)$ can substantially understate true incentive problems. For that reason, the next section focuses on practical training procedures with *strong* deviation oracles and on ways to estimate (or upper bound) the relevant Lipschitz quantities in implementations with modern feasibility layers.

7 Algorithms: Robust Training with Strong Deviation Oracles

We now turn from guarantees to the practical question: how do we *train* a parameterized combinatorial auction mechanism so that it is both (i) high-revenue and (ii) hard to manipulate, *even when deployment types shift within a Wasserstein budget*? Our guiding design choice is to mirror the structure of the robust objective: an *outer* minimization over mechanism parameters, coupled with (at least) two *inner* maximizations—one over bidder deviations (to estimate regret) and, conceptually, one over distributions in the ambiguity set (to enforce robustness). The main algorithmic challenge is that both inner problems are nontrivial in combinatorial domains.

7.1 Training loop: outer updates with inner deviation maximization

Fix a sample set $S = \{v^{(\ell)}\}_{\ell=1}^{L}$ and consider minibatches $B \subseteq S$ in stochastic training. For a given mechanism parameter w and valuation profile v, recall the regret definition

$$\mathrm{rgt}_i(w; v) \;=\; \max_{b_i \in \mathcal{V}_i} u_i(v_i; (b_i, v_{-i}); w) - u_i(v_i; v; w)\,, \qquad \mathrm{rgt}(w; v) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{rgt}_i(w; v).$$

In practice, we approximate each bidder's best deviation via a *deviation oracle* that searches over b_i given (v_i, v_{-i}, w). The outer loss on a profile v is then

$$f_w(v) \;=\; -\mathrm{rev}(w; v) + \lambda\, \widehat{\mathrm{rgt}}(w; v),$$

where $\widehat{\mathrm{rgt}}$ denotes the regret estimate returned by the oracle (typically a lower bound on true regret, hence conservative for incentives). The outer step performs gradient descent on w using the differentiable computation graph for (g_w, p_w) as well as (when applicable) differentiable components of the oracle.

A useful mental model is that we are solving a saddle-point problem by alternating updates:

1. **Deviation step (inner max).** For each $v \in B$ and each bidder i, approximately solve

   $$b_i^\star(v; w) \;\approx\; \arg\max_{b_i \in \mathcal{V}_i} u_i(v_i; (b_i, v_{-i}); w)$$

   using gradient ascent and/or combinatorial search (described below). Record the resulting utility gain as $\widehat{\mathrm{rgt}}_i(w; v)$.

2. **Mechanism step (outer min).** Update w to decrease a robust surrogate of $\mathbb{E}[f_w(v)]$ over the minibatch, i.e., a sample average plus a robustness correction.

Because regret is defined by a maximization, a weak deviation oracle can make training appear successful while leaving exploitable incentive violations at deployment. For that reason, we intentionally bias computation toward a *strong* oracle: more ascent steps, multiple random restarts, and (when the domain permits) hybrid discrete–continuous local improvements. This increases training cost but is precisely what makes the eventual incentive claims meaningful.

7.2 Strong deviation oracles for combinatorial reports

A report b_i is itself a vector over bundles, often high-dimensional ($k = 2^m - 1$). Two practical constraints help: (i) $\mathcal{V}_i \subset [0, V_{\max}]^k$ is compact, and (ii) we can parameterize feasible reports by a differentiable map (e.g., elementwise clipping or a low-dimensional embedding) that keeps the search within $\mathcal{V}_i$.

Our default deviation oracle is *projected gradient ascent*:

$$b_i^{(t+1)} \;=\; \Pi_{\mathcal{V}_i}\Big(b_i^{(t)} + \eta_{\text{dev}} \nabla_{b_i}\, u_i\Big(v_i; (b_i^{(t)}, v_{-i}); w\Big)\Big)\,,$$

run for T_{dev} steps with R random restarts, returning the best b_i found. The projection $\Pi_{\mathcal{V}_i}$ is typically implemented by coordinatewise clipping when $\mathcal{V}_i$ is a box; for structured type spaces (e.g., additivity or submodularity constraints), $\Pi_{\mathcal{V}_i}$ can be replaced by a differentiable parametrization that enforces structure by construction. In either case, the oracle only requires that we can backpropagate through u_i, hence through g_w and p_w.

Two refinements materially strengthen the oracle in combinatorial settings:

- **Targeted coordinate search.** Gradient steps can be complemented by a small neighborhood search over economically salient coordinates (e.g., singleton bundles and the grand bundle), which often capture the most profitable manipulations when the mechanism is approximately itemwise.
- **Warm-starting and caching.** We cache each bidder's previous best deviation for similar v (or the same training point across epochs) and warm-start from it. This reduces variance and makes the inner maximization harder to "forget" as w changes.

When we report regret in experiments, we evaluate using an *even stronger* oracle than the one used in training (more steps, more restarts, additional heuristics). This separation is important: if evaluation uses the same oracle, one risks measuring the oracle rather than the mechanism.

7.3 Robust outer objectives: implementing the Wasserstein correction

The distributionally robust objective involves $\sup_{Q:W_1(Q,\hat{P}_L)\le\delta} \mathbb{E}_Q[f_w(v)]$, which is not directly optimized by naive minibatch averaging. We implement robustness in one of two ways, depending on whether we want a *certifiable* bound (via Lipschitz control) or a *constructive* worst-case stressor (via adversarial perturbations of types).

(i) Lipschitz-penalized surrogate. Using Proposition 1, a tractable surrogate is

$$\widehat{\mathcal{L}}_\delta(w) \;=\; \frac{1}{|B|}\sum_{v\in B} f_w(v) \;+\; \delta\,\widehat{L}_f(w), \qquad \widehat{L}_f(w) \;\ge\; \mathrm{Lip}(f_w).$$

Here $\widehat{L}_f(w)$ is an upper bound or estimate of the Lipschitz constant of $v \mapsto f_w(v)$ under d. Since $f_w(v) = -\mathrm{rev}(w;v) + \lambda\,\mathrm{rgt}(w;v)$, we can bound

$$\mathrm{Lip}(f_w) \;\le\; \mathrm{Lip}(\mathrm{rev}(w;\cdot)) \;+\; \lambda\,\mathrm{Lip}(\mathrm{rgt}(w;\cdot)).$$

We use this decomposition to regularize the mechanism toward smoother, shift-stable behavior: for fixed δ, a larger Lipschitz bound directly increases the robust objective, making overly sharp mechanisms unattractive.

(ii) Adversarial type perturbations (empirical worst-case). Alternatively, we approximate the Wasserstein adversary by constructing perturbed valuations v' near each training point v, solving

$$v' \;\approx\; \arg\max_{\tilde{v}\in\mathcal{V}} \;\big\{f_w(\tilde{v}) \;-\; \gamma\, d(\tilde{v}, v)\big\},$$

for a chosen dual parameter $\gamma > 0$ (motivated by the Kantorovich–Rubinstein dual). This yields "hard" local environments (nearby in d) that empirically enlarge regret or depress revenue, and is useful as a stress-test during training. Conceptually, this is analogous to adversarial training in supervised learning: we are not only fitting the observed v, but also the worst nearby v' deemed plausible by the transportation metric.

7.4 Estimating and upper-bounding Lipschitz constants

Robust training is only as meaningful as our control of $\widehat{L}_f(w)$. We therefore combine three complementary tools, trading off tightness and computability.

Architecture-based upper bounds. When g_w and p_w are implemented by neural networks with norm constraints, we can bound their Lipschitz constants by products of layer operator norms (e.g., spectral norms for linear maps, 1 for ReLU/softplus). Denoting these bounds by $\overline{L}_g(w)$ and $\overline{L}_p(w)$, Proposition 2 motivates a bound of the form

$$\widehat{L}_f(w) \;=\; \overline{L}_{\mathrm{rev}}(w) \;+\; \lambda\,\overline{L}_{\mathrm{rgt}}(w), \qquad \overline{L}_{\mathrm{rgt}}(w) \;\lesssim\; c_1\,\overline{L}_g(w) + c_2\,\overline{L}_p(w),$$

with constants c_1, c_2 determined by $V_{\max}$ and the chosen metric normalization. In practice, we enforce these bounds via spectral normalization and weight decay, which provides an interpretable "smoothness knob" for robustness.

Jacobian-based empirical estimates. Upper bounds based on layer norms can be loose. As a complementary diagnostic, we estimate local Lipschitz behavior on minibatches via Jacobian norms (e.g., $\|\nabla_v f_w(v)\|_*$ for the dual norm induced by d), and maintain an exponential moving average as a stability indicator. While not a certificate, this estimate is informative for tuning δ and λ: if $\|\nabla_v f_w\|$ spikes, the mechanism is likely fragile to small shifts.

Direct regularization for smoothness. Finally, we can regularize f_w directly by penalizing differences across nearby samples:

$$\Omega(w) \;=\; \mathbb{E}_{v,v'\sim\hat{P}_L}\left[\frac{|f_w(v) - f_w(v')|}{d(v,v') + \xi}\right],$$

with a small $\xi > 0$ for numerical stability. This encourages Lipschitz-like behavior without explicitly computing global constants, and empirically improves robustness when the metric d aligns with realistic market variation.

7.5 Feasibility layers: implementing g_w with CAFormer/CANet

All of the above presumes that gradients through g_w are meaningful and that allocations remain combinatorially feasible throughout training. We therefore implement g_w using feasibility layers inspired by CAFormer/CANet: the network produces a structured score representation over bidders and bundles (or over items with bundle reconstruction), followed by a differentiable layer that enforces the constraints

$$\sum_{i\in N}\sum_{S\ni j} z_{iS} \le 1 \;\; \forall j \in M, \qquad \sum_{S\in K} z_{iS} \le 1 \;\; \forall i \in N, \qquad 0 \le z_{iS} \le 1.$$

In practice, this can be realized by (i) softmax-based normalizations combined with itemwise capacity penalties, (ii) differentiable projections onto a relaxation of the feasible polytope, or (iii) attention-based architectures that

allocate item capacities across bidders in a permutation-invariant way. The key implementation point is that the feasibility layer must be stable under the inner deviation search: if b_i changes during oracle ascent, $g_w(b)$ must remain feasible and differentiable, otherwise regret gradients become noisy or misleading.

Payments p_w are implemented with an explicit nonnegativity constraint (e.g., softplus output) and, when desired, an ex-post IR enforcement step that caps payments by reported value of the allocated bundle. We emphasize a limitation: while such caps help satisfy IR mechanically, they can interact with deviation incentives in subtle ways, and must be audited with the strong deviation oracle.

Taken together, these components yield a training procedure that is economically interpretable: the deviation oracle plays the role of a strategic "auditor," the Wasserstein correction operationalizes shift robustness, and feasibility layers guarantee that every gradient step corresponds to a meaningful (albeit fractional) combinatorial allocation rule.

8 Experiments: Synthetic Shifts and Platform-Inspired Stress Tests

Our theoretical guarantees are only useful insofar as they translate into mechanisms that remain *hard to manipulate* when the environment changes. We therefore design experiments around a simple organizing principle: we train mechanisms on a baseline distribution $\hat{P}_L$ and then evaluate them on a family of deployment distributions Q that are *plausibly close* to training in the sense of $W_1(Q, \hat{P}_L) \leq \delta$, as well as on shifts that mimic common marketplace dynamics (seasonality, demand shocks, and changes in complementarity patterns). Across all experiments we report two objects: (i) seller revenue under truthful play, and (ii) bidder incentive compatibility measured by ex-post regret, computed using a deviation oracle that is *strictly stronger at evaluation than during training*. This asymmetry is deliberate: it operationalizes the economic idea that a mechanism should withstand strategic sophistication beyond what the designer explicitly anticipated.

8.1 Experimental setup and baselines

We consider standard combinatorial settings with m items and n bidders, with valuations bounded in $[0, V_{\max}]^k$ and type spaces compact as assumed above. Each mechanism outputs a feasible fractional allocation and nonnegative payments, and we evaluate on held-out profiles drawn from various deployment distributions. We compare three families of learned mechanisms:

RegretNet (non-robust). We train a standard regret-penalized model by minimizing the empirical objective $\frac{1}{L}\sum_{\ell}\big[-\mathrm{rev}(w;v^{(\ell)})+\lambda\,\widehat{\mathrm{rgt}}(w;v^{(\ell)})\big]$ without any Wasserstein correction. This baseline isolates the contribution of robustness to performance under shift.

CAFormer/CANet-style allocation architectures (non-robust). We also train architectures emphasizing combinatorial structure and feasibility (attention over items/bundles, structured scoring, and differentiable feasibility enforcement). These typically improve in-distribution revenue and feasibility stability, but they are not, by themselves, robust to distribution shift in the incentive sense.

Robust RegretNet (ours). We train using the robust surrogate induced by the Wasserstein ambiguity set, implemented either as a Lipschitz-corrected objective (via a bound $\widehat{L}_f(w)$) or as adversarial type perturbations aligned with the W_1 dual. We treat δ as a design parameter and examine how varying δ traces out a revenue–robustness frontier.

For all methods, we hold fixed the evaluation protocol: regret is computed with a stronger oracle (more ascent steps, more restarts, and hybrid local search over salient bundles). This ensures that any apparent incentive gains are attributable to the mechanism rather than to a weak adversary.

8.2 (i) Controlled Wasserstein shift: calibrating the ambiguity radius

We begin with a controlled setting where we can explicitly generate a continuum of deployment distributions Q_t whose distance to the training distribution increases smoothly. Concretely, we generate a baseline type distribution with a tunable degree of substitutability/complementarity, then create shifted distributions by transporting mass along interpretable directions in valuation space (e.g., increasing the marginal values of a subset of items, increasing complementarity for certain pairs, or increasing dispersion across bidders). We numerically verify that $W_1(Q_t,\hat{P}_L)$ grows approximately linearly in the shift magnitude, and we map each t to a corresponding effective δ.

Two empirical regularities emerge. First, the non-robust baselines (RegretNet and CAFormer-style models) exhibit what we view as a form of *incentive fragility*: regret remains low near the training distribution but increases sharply once the shift crosses a modest threshold. Importantly, this increase is often concentrated in a small subset of bidder profiles, consistent with the idea that manipulation opportunities may be rare in-sample but severe out-of-sample. Second, Robust RegretNet degrades more gracefully: as t increases, regret rises more slowly and the upper tail of the regret distribution is substantially controlled. This is precisely the practical content

of a Wasserstein-style guarantee: we are not promising that incentives are perfect everywhere, but that small shifts do not uncover entirely new regions of exploitability.

Revenue exhibits the expected tradeoff. For small δ, robust training is nearly indistinguishable from non-robust training in-distribution, while providing measurable out-of-distribution gains (higher revenue under strategic play due to reduced manipulation and lower worst-case regret). As δ increases, robust mechanisms become more conservative: in-distribution revenue falls, but the worst-case revenue across shifted Q_t improves relative to the non-robust baselines. This is a concrete manifestation of the revenue–robustness tradeoff highlighted in Proposition 4.

8.3 (ii) Covariate/context shift: stress-testing with latent market conditions

Many real platforms face not only changes in the distribution of valuations, but changes in the *process* generating them: product assortments, buyer mix, and correlation patterns evolve with context (seasonality, promotions, entry/exit). To emulate this, we introduce a latent context variable c that affects valuation generation (e.g., changing correlation across items, changing the frequency of near-unit-demand vs highly complementary bidders, or changing the distribution of bidder budgets). We train on contexts drawn from a baseline mixture and evaluate on altered mixtures that change the prevalence of each context.

This shift is qualitatively different from a simple location shift because it changes *which parts of the type space are common*. In our experiments, non-robust mechanisms often over-specialize to the dominant training context: they extract high revenue there but leave "holes" in incentive performance in the minority contexts. When the deployment mixture tilts toward these minority contexts, regret rises and realized revenue under strategic play deteriorates. Robust RegretNet partially immunizes against this failure mode. Intuitively, robustness forces the mechanism to perform acceptably not only on average, but also on nearby distributions that reweight contexts, thereby reducing dependence on fragile correlations.

A practical takeaway is that robustness can be interpreted as an *insurance policy* against mis-specified market segmentation. If the platform designer does not know which segments will dominate in the future, a moderate δ can reduce the cost of being wrong.

8.4 (iii) Adversarial reweighting: worst-case emphasis on "hard" profiles

Our third stress test makes the adversary explicit. Rather than perturbing valuations pointwise, we adversarially reweight the evaluation set toward

profiles that maximize either regret or the robust loss $f_w(v) = -\text{rev}(w; v) + \lambda\, \text{rgt}(w; v)$ subject to a transportation budget. This corresponds to the economic scenario in which the platform encounters a population that is systematically more sophisticated or more complementary than the training data suggests, but not arbitrarily so.

Empirically, adversarial reweighting reveals that incentive violations are often driven by economically interpretable configurations: (i) a bidder with strong value for the grand bundle facing competitors with strong singleton demand, or (ii) profiles where two bidders have near-ties on several bundles, making allocation discontinuities profitable to exploit. These are precisely the profiles a seller might see during demand spikes or when large buyers enter. Robust RegretNet reduces both the frequency and severity of these worst-case profiles by smoothing the response of allocation and payments to reported values, which in turn reduces the marginal gains from strategic misreports.

8.5 Evaluation with stronger deviation searches

Across all three shift families, we find that the ranking of mechanisms by regret is sensitive to the strength of the deviation oracle. When we intentionally weaken the oracle, most methods appear approximately DSIC; when we strengthen it, non-robust mechanisms often exhibit substantial hidden regret. Robust RegretNet is not immune to this issue, but it is more stable: strengthening the oracle increases measured regret by a smaller margin, suggesting that robustness and smoothness regularization reduce the prevalence of sharp local profitable deviations.

This observation has an important methodological implication for mechanism learning: reported regret should be interpreted as a lower bound whose tightness depends on the auditor. In the discussion below, we return to how a platform might operationalize this in deployment via monitoring and safe update rules.

9 Discussion and policy/engineering implications

Our experiments highlight a central practical point: incentive compatibility is not merely an in-sample property. Even when a learned mechanism exhibits low measured regret on held-out data from the training distribution, small and economically plausible shifts can uncover pockets of high exploitability. The purpose of the Wasserstein formulation is therefore less to "solve" misspecification than to make it *priced and tunable*: by choosing an ambiguity radius δ, the designer selects how much out-of-distribution insurance to purchase, and the theory clarifies the premium this insurance commands in expected revenue (Proposition 4). In this section we translate

that logic into design guidance for platforms, and we flag limitations that matter for deployment.

Choosing the ambiguity radius as a policy choice. From an economic perspective, δ plays the role of a robustness budget—how far the deployment environment may deviate from what was observed. In many platforms, the relevant uncertainty is not adversarial in the worst sense, but it is persistent: seasonality, product churn, and buyer entry/exit change valuation profiles in ways that are difficult to predict but rarely unbounded. A pragmatic calibration strategy is therefore *scenario-based*: the platform enumerates a set of stress scenarios (e.g., a demand spike for a subset of items, or an increase in complementarity driven by bundles becoming salient) and computes empirical estimates of their transportation distance to the baseline, $W_1(\hat{Q}, \hat{P}_L)$. One can then set δ to cover, say, the 90th percentile of historically observed shifts or the largest shift the platform deems operationally plausible. This turns robustness from an abstract minimax choice into a governance decision akin to setting a risk limit.

A complementary approach is to use the dual surrogate in Proposition 1 to interpret δ as a *marginal penalty* on sensitivity. If the robust objective is approximated by

$$\frac{1}{L}\sum_{\ell=1}^{L} f_w\big(v^{(\ell)}\big) \;+\; \delta\,\widehat{\mathrm{Lip}}(f_w), \qquad f_w(v) = -\mathrm{rev}(w;v) + \lambda\,\mathrm{rgt}(w;v),$$

then increasing δ is equivalent to demanding a smaller effective Lipschitz constant of the revenue–regret tradeoff. When platform operators have an interpretable "cost of volatility"—for instance, the operational cost of policy reversals when incentives break—it is natural to map that cost into a target bound on $\widehat{\mathrm{Lip}}(f_w)$ and choose δ accordingly. We emphasize, however, that $\widehat{\mathrm{Lip}}(f_w)$ is itself an object that must be estimated or upper bounded, and conservative estimation will push mechanisms toward more muted allocations and payments.

Interacting roles of δ and λ. In practice, δ and λ are not substitutes. The regret weight λ controls *how expensive* manipulation is made during training, while δ controls *where* we ask the mechanism to be safe. A useful heuristic is to first choose a target incentive level (e.g., an acceptable expected regret threshold ε under the baseline) by tuning λ, and then increase δ until regret remains acceptably small under stress tests. This mirrors common engineering practice: fix a performance target under nominal conditions, then broaden the operating envelope. The comparative statics in the global context provide guidance on the expected direction of changes, but the quantitative choice is ultimately empirical because both rev and rgt depend on architecture, regularization, and the deviation oracle.

Monitoring distribution shift in deployment. Robust training does not eliminate the need for monitoring; rather, it changes what monitoring is for. Under a Wasserstein model, what matters is not whether the empirical distribution has changed in some generic sense, but whether it has moved beyond the insured neighborhood. A platform can operationalize this by maintaining a rolling empirical distribution $\hat{Q}_t$ of recent valuation proxies (or features predictive of valuations) and periodically estimating $W_1(\hat{Q}_t, \hat{P}_L)$ under the same metric d used in training. When direct valuations are unobserved (as in many marketplaces), one can still monitor sufficient statistics tied to the model class: predicted values, bid distributions, or allocation-relevant embeddings. The metric choice becomes a policy lever here: weighting certain bundles or items more heavily in d encodes the platform's belief about which shifts are consequential for incentives.

We also recommend monitoring *strategic behavior indicators* alongside covariate drift. Regret is not directly observable, but sudden changes in bid shading patterns, jump discontinuities in allocation probabilities near common bid levels, or increased dispersion in realized payments conditional on similar bids may signal that bidders have discovered profitable deviations. In that sense, the regret framework provides an auditing lens: one does not only ask "did demand shift?" but also "did the shift create new manipulability?"

Safe updates and "trust-region" deployment. A recurring engineering problem is how to update mechanisms without creating incentive shocks. Our framework suggests two guardrails. First, updates should be *robustness-preserving*: when retraining on new data, one can keep the ambiguity radius at least as large as before (or increase it if monitoring indicates drift), ensuring that new mechanisms are safe on a neighborhood that includes both past and present conditions. Second, updates should be *local* in mechanism space. Concretely, one can add a trust-region penalty that discourages large changes in allocation or payment maps:

$$\mathbb{E}_{v\sim\hat{P}_{\text{mix}}}\left[\|g_w(v) - g_{w^{\text{old}}}(v)\| + \|p_w(v) - p_{w^{\text{old}}}(v)\|\right] \le \tau,$$

for a mixture $\hat{P}_{\text{mix}}$ of recent and historical data. Economically, this is a commitment device: it limits the platform's ability to inadvertently introduce new discontinuities that invite gaming. Operationally, it enables staged rollouts with rollback options, where τ plays the role of an allowable policy delta.

Limitations: deviation oracles and the meaning of "low regret." A first limitation is methodological but economically important: computed regret is only as strong as the deviation search. Our evaluation used stronger oracles than training, yet even that may miss profitable global deviations in high-dimensional type spaces. This implies that any regret guarantee should

be interpreted as *audited approximate DSIC*, not DSIC in the classical sense. For deployment, the implication is to treat the deviation oracle as part of compliance infrastructure. Platforms that face sophisticated bidders should invest in stronger auditing—more powerful optimization, multiple initialization schemes, and targeted searches over economically salient deviations (e.g., bundle misreports that mimic unit-demand or pure complementarity). A conservative practice is to maintain an "oracle gap" dashboard: the difference between regret measured by a fast oracle (used during routine checks) and a slower, stronger oracle (used periodically). A widening gap is a warning sign even if the fast-oracle regret remains low.

Limitations: nonconvex training and robustness estimation. A second limitation is optimization. The robust objective is nonconvex in w, and the Wasserstein surrogate can exacerbate gradient variance when implemented via adversarial perturbations. As a result, different random seeds may yield mechanisms on different points of the revenue–robustness frontier. In settings where policy stakes are high, we view this as an argument for ensemble-style governance: train multiple candidates, audit each with a strong oracle under a battery of shifts, and select the mechanism that minimizes a conservative upper bound on regret subject to acceptable revenue. Relatedly, our theory relies on Lipschitz control (Proposition 2), but bounding $\mathrm{Lip}(f_w)$ tightly for deep networks remains challenging. Overly loose bounds will induce excessive conservatism, while overly optimistic bounds risk under-insuring. Improving practical estimators of these constants is an important engineering task.

Next steps. Several extensions are natural. First, learning or adapting the metric d from domain knowledge (or from observed drift) could make Wasserstein neighborhoods align better with plausible economic changes, improving the meaning of δ. Second, extending robustness to richer bidder models—budget constraints, risk aversion, or correlated types—would bring the approach closer to platform realities, albeit with new challenges in oracle design and feasibility. Third, the dynamic setting is unavoidable in many applications: repeated participation and learning by bidders can turn small one-shot regret into large long-run revenue effects. A promising direction is to combine our one-shot robust incentives with online monitoring and constrained updates, yielding a mechanism that is not only robust to distribution shift, but also robust to bidder adaptation.

Taken together, we view Wasserstein-robust mechanism learning as a disciplined way to manage a familiar tradeoff in market design: aggressive revenue optimization exploits patterns in observed demand, while robust incentive performance hedges against the patterns changing. The appropriate balance is context-dependent, but the framework makes that balance

explicit, auditable, and adjustable as the marketplace evolves.