Automated Design of Affine Maximizer Mechanisms in Dynamic Settings

Michael J. Curry[*]
University of Zurich & ETH AI Center
curry@ifi.uzh.ch

Vinzenz Thoma[*]
ETH Zurich & ETH AI Center
vinzenz.thoma@ai.ethz.ch

Darshan Chakrabarti
Columbia University

Stephen McAleer
Carnegie Mellon University

Christian Kroer
Columbia University

Tuomas Sandholm
Carnegie Mellon University

Niao He
ETH Zurich

Sven Seuken
University of Zurich & ETH AI Center

Abstract

Dynamic mechanism design is a challenging extension to ordinary mechanism design in which the mechanism designer must make a sequence of decisions over time in the face of possibly untruthful reports of participating agents. Optimizing dynamic mechanisms for welfare is relatively well understood. However, there has been less work on optimizing for other goals (e.g. revenue), and without restrictive assumptions on valuations, it is remarkably challenging to characterize good mechanisms. Instead, we turn to *automated mechanism design* to find mechanisms with good performance in specific problem instances. We extend the class of affine maximizer mechanisms to MDPs where agents may untruthfully report their rewards. This extension results in a challenging bilevel optimization problem in which the upper problem involves choosing optimal mechanism parameters, and the lower problem involves solving the resulting MDP. Our approach can find truthful dynamic mechanisms that achieve strong performance on goals other than welfare, and can be applied to essentially any problem setting—without restrictions on valuations—for which RL can learn optimal policies.

1 Introduction

Dynamic mechanism design studies sequential decision-making problems, where decisions are based on the self-reported preferences of agents. A typical model is that the environment consists of a *Markov decision process (MDP)*, and the mechanism controls the process given reported utilities by the agents. This has important applications, such as ad auctions or more generally online pricing (e.g. Bergemann and Välimäki 11) but also problems of decentralised decision making in RL (e.g. Chang et al. 15).

Much work in dynamic mechanism design has focused on maximizing welfare [6, 42, 37] subject to *strategyproofness* (there should be no incentive for untruthful reports by agents). Some other work considers different goals, notably revenue [10, 30, 27, 28] but needs to make restrictive assumptions about the space of agent types. Work on dynamic mechanism design, for general goals and for broad spaces of agent types, is much more limited.

[*]Equal contribution.

16th European Workshop on Reinforcement Learning (EWRL 2023).

Dynamic mechanism design includes as a special case static mechanism design, and here the situation is similar. To maximize welfare while ensuring strategyproofness, one can use the celebrated and well-understood Vickrey-Clarke-Groves (VCG) mechanism [59, 16, 26]. For optimizing revenue, Myerson [40] completely settles the question under the restrictive assumption that agents' types are single-dimensional (essentially, that there is only one type of item up for sale); beyond this there has been little progress except for very specific problem instances [62]. But for static settings, *automated mechanism design (AMD)* [18, 49, 21] has been used: this is a data-driven search through some class of mechanisms in order to find one that performs well while satisfying the constraints of strategyproofness and individual rationality. Automated mechanism design has in some cases found the highest-performing mechanisms known so far, and can recover optimal mechanisms in special cases where they are known [25, 29, 54].

Given the successes of automated mechanism design for static problems, it is surprising that its use for dynamic problems is relatively underexplored.

1.1 Our Contributions

Our present paper develops automated dynamic mechanism design techniques which can be applied to a very broad range of problems. In particular, we consider mechanism design on general MDPs. Our model works with many loss functions (including revenue but also other domain-specific loss functions), not just the easier goal of welfare, and we do not assume one-dimensional agent types. Our assumptions are sufficiently general to capture essentially all of static multi-parameter mechanism design as a special case[2].

Optimal mechanism design over all possible mechanisms entails the very difficult problem of computing equilibria in imperfect information games, to understand whether or not any agent has any incentive to deviate from truthful reporting. Inspired by prior work for static mechanism design, we sidestep this issue by focusing on the class of affine maximizers (AMAs), which we define on MDPs.[3] These mechanisms are always strategyproof.

We justify this restriction in two ways. First, as mentioned, our problem assumptions capture multi-parameter mechanism design—but finding optimal mechanisms in this setting has proven extremely difficult, so it makes sense to search within a more tractable class of mechanisms. Second, our framing of the problem is broad enough that it captures problem instances where Roberts' theorem [48] applies, which states that under general agent type spaces, *only* affine maximizers can be strategyproof.

We frame the search for a high-performing **dynamic affine maximizer mechanism** as a **bilevel optimization problem**, where the outer problem consists of choosing weights and (possibly state-dependent) boosts—the AMA parameters—to minimise a given loss function. The inner problem consists of learning to control an MDP to maximise affinely-transformed social welfare (the definition of an affine maximizer), given the weights, boosts, and agents' type reports (see equation 1 below). The derivatives of this inner problem do not exist at many points: however, we show that for the important case of revenue, the *expected* loss over the distribution (with a continuous density) of agent valuations is differentiable.

We solve the inner problem via (possibly regularized) linear programming. We also propose a variety of ways to solve the outer problem: by grid search, by differentiating through the regularized LP, or by using zeroth-order methods to approximate the LP gradient. These latter two approaches explicitly or implicitly smooth the objective and avoid the problem of nonexistent derivatives. In experiments on several dynamic mechanism design settings, such as sequential auctions, task scheduling and navigating a gridworld, our approaches result in truthful mechanisms that outperform the VCG baseline.

[2]If we restrict the MDP to have a single state, then we recover ordinary mechanism design, with each possible action corresponding to an outcome.

[3]The acronym AMA refers to "affine maximizer auctions". We consider affine maximizer *mechanisms*, but we stick to the AMA acronym because it is widely used and to avoid confusion with AMM used to refer to "automated market makers".

2 Related Work

Maximizing Welfare in Dynamic Mechanism Design Athey and Segal [6] consider a dynamic mechanism design setting where agents update their beliefs over time, and where the goal is an efficient and budget-balanced outcome. Parkes [43] describes a dynamic mechanism design setting where the focus is on agents who may arrive and depart at different periods. Both of these approaches simply assume an optimal policy is available. More recently, Lyu et al. [37] presents a model for learning this policy in an offline RL setting; another work focuses on the online case [36]. Chang et al. [15] consider a dynamic VCG mechanism for decentralised RL where agents bid on the MDP transitions. Bergemann and Välimäki [10] presents a VCG-like "dynamic pivot mechanism".There are different modeling choices and goals in each of these approaches, but the common theme is that the allocation problem involves making decisions on some MDP after observing agent reports. All of these papers consider a dynamic analogue to the VCG mechanism, i.e. the mechanism designer acts according to the welfare-optimal policy and charges agents their externalities to ensure incentive compatibility. This is in contrast to our work, where we are concerned with goals beyond welfare.

Dynamic Mechanism Design for Goals Other Than Welfare There is some existing work in this direction as well, with a particular focus on revenue. Pavan et al. [45] and Kakade et al. [30] both consider cases where the private information about the value of the item is 1-dimensional. This allows for a Myerson-style analysis of the actual profit-maximizing mechanism. Other work considers optimal selling of items to agents who arrive and depart over time from the perspective of optimal stopping, but again considers single-parameter item valuations [27, 28, 31]. Bergemann and Välimäki [11] surveys other results that make similar assumptions for tractability. Pai and Vohra [44] considers a similar setting and finds the optimal Bayes-Nash incentive compatible mechanism. Still other work considers settings where the mechanism designer may update the mechanism online over time (e.g. by changing reserve prices) [53] and where bidders may even strategically attempt to manipulate the learning process [4]. Overall, these approaches are restrictive in terms of their assumed value model and frequently focus on analytic results, while our computational approach allows more general values and loss functions.

Preference Elicitation from Multiple Agents and Multistage Mechanisms Another line of work considers iterative preference elicitation [17, 50]: based on past agent reports, the mechanism can query what preference information it needs most. Existing approaches make use of an analogy to query learning [63, 13, 33], or leverage machine learning [56, 61, 14]. In this context, Sandholm et al. [52] use automated mechanism design to first find good mechanisms (for general type spaces) and then convert them into multistage mechanisms. While these approaches are typically focused on making a single final decision, but eliciting agent's preferences over multiple stages, our automated dynamic mechanism design approach is concerned with making a sequence of decisions over time, while eliciting preferences once.

Static Automated Mechanism Design Due to the difficulty of analytically finding optimal mechanisms, a number of works have instead attempted to treat static mechanism design as a computational optimization problem, starting with Conitzer and Sandholm [18] and Sandholm [49], and learn good mechanisms from samples, starting with Likhodedov and Sandholm [34]. One line of work makes use of static affine maximizers and achieves good performance in multi-item multi-bidder auctions [51, 21, 23]. There is also a line of learning theory research on choosing the AMA parameters given samples from the valuation distribution [9, 7, 8]. Another direction is to start with a potentially non-strategyproof mechanism and iteratively modify it to improve strategyproofness. This is known as *incremental mechanism design* [19]. One line of work in this direction makes use of rich function approximators to learn mechanisms. Duetting et al. [25] presents one influential direction, which uses neural networks to optimize revenue and a penalized loss to approximately enforce strategyproofness, with many followups [22, 20, 29, 47, 46]. Shen et al. [54] presents an alternative approach for single-bidder settings which can cope with broader classes of utility functions. As mentioned above, the success of these techniques in static settings is our motivation to develop such approaches for the dynamic case.

3 Preliminaries

Below, we describe our mechanism design problem. The nature of the problem and our mechanism design desiderata motivate our choice to restrict attention to affine maximizer mechanisms. We then describe in more detail how these mechanisms work in a dynamic setting.

3.1 Formal Model of Problem Setting

Environment and policy/allocation rule Consider some MDP $\mathcal{M} = (S, A, P)$ (with reward not yet specified), where S is a set of states, A is a set of actions, and P is a transition function. There are n agents, each with their own reward function $r_i : S \times A \to \mathbb{R}$ drawn from a distribution with density f_i. We emphasize that these agents are not themselves taking actions in the MDP—this is done by the mechanism. Their only choice will be which rewards to report to the mechanism. We assume the mechanism designer wants to minimize some loss function (which will often be the negative of some objective to be maximized) $\mathcal{L}$ in expectation over the f_i, which in general depends on the chosen policy π on the MDP, which corresponds to the allocation rule in traditional mechanism design.

3.2 Mechanism Design Desiderata

Incentive compatibility and payments In order to achieve its goal, the mechanism will need access to the true $r_i(s, a)$. However, in general, we should expect the agents to misreport their reward function if they think it will benefit them. Thus, we will allow for some side payments to be made based on the MDP's solution and the agents' reports, in order to ensure that there is no incentive to misreport, that is, making the mechanism *incentive compatible (IC)* or *strategyproof.* (Such payments only exist for some choices of π.) We assume agent utility is quasilinear, that is, positive payments just correspond to negative reward.

Individual rationality We also want to guarantee individual rationality (IR), meaning that agents should not be charged so much that they receive negative utility and would be better off not participating in the mechanism.

Remark. *The mechanism designer's goal may also relate to the payments. For example, a canonical mechanism design goal is to maximize revenue—in our setting, choose a policy π^* such that payments can be made as high as possible while still ensuring IC and IR.*

3.3 Background on Affine Maximizers

As motivated above our goal is to choose some π such that IC/IR side payments can be constructed, while also performing as well as possible on the mechanism designer's higher-level objective.

Unlike prior work (e.g., Kakade et al. 30, Bergemann and Välimäki 10), we do not make any assumptions about the structure of the reward functions. Our setting is therefore general enough to incorporate many hard problems such as optimal multi-item mechanism design, and is thus at least as hard as those. Therefore hoping to get the truly best-performing π, even only in an infinite-sample/asymptotic sense, is too much. Thus it is appropriate to restrict attention to a more tractable class of mechanisms.

Also, our problem setting is general enough to include situations where a result known as Roberts' theorem applies [48]. It states that under certain conditions (arbitrary rewards, at least 3 outcomes), the only allocation rules that can possibly have IC payments take the form of affine maximizers.

Below, we give the standard definition of affine maximizer mechanisms in terms of the allocation and payment rules, modified for our dynamic problem setting.

Definition 3.1 (Affine maximizers)**.** *Given so-called weights $w \in \mathbb{R}^n_+$ and boosts $b \in \mathbb{R}^{|S|\times|A|}$, a dynamic affine maximiser mechanism (AMA) takes reported reward functions $\boldsymbol{r} \in \mathbb{R}^{|S|\times|A|\times n}$ and returns a policy $\pi^*(w, b, \boldsymbol{r})$ on the MDP that maximizes the affine social welfare $\mathsf{asw}^{(\pi)}(w, b, \boldsymbol{r})$ where*

$$\mathsf{asw}^{(\pi)}(w, b, \boldsymbol{r}) = \mathbb{E}_\pi \left[\sum_{t=0}^{T} \left(\sum_{i=1}^{n} w_i r_i(s_t, a_t) \right) + b(s_t, a_t) \right]$$

Defining $\mathsf{asw}(w, b, \boldsymbol{r}) = \mathsf{asw}^{(\pi^*(w,b,\boldsymbol{r}))}(w, b, \boldsymbol{r})$ *as the maximum affine social welfare for reports* $\boldsymbol{r}$*, the resulting payment is then*

$$\boldsymbol{p}_i(w, b, \boldsymbol{r}) = \frac{1}{w_i}\left(\mathsf{asw}^{(-i)}(w, b, \boldsymbol{r}) - \left(\mathbb{E}_{\pi^*(w,b,\boldsymbol{r})}\left[\sum_{t=0}^{T}\left(\sum_{j\neq i} w_j r_j(s_t, a_t)\right) + b(s_t, a_t)\right]\right)\right)$$

where $\mathsf{asw}^{(-i)}(w, b, \boldsymbol{r})$ *defined as* $\max_\pi \mathbb{E}_\pi\left[\sum_{t=0}^{T}\left(\sum_{j\neq i} w_j r_j(s_t, a_t)\right) + b(s_t, a_t)\right]$ *is the maximum affine social welfare, when disregarding* i*.*

No matter the choice of weights and boosts, every resulting AMA is strategyproof ex-post — after learning the rewards of the other agents, reporting truthfully is a dominant strategy — and IR in expectation over the MDP trajectories. The proof can be found in Appendix A.

4 Dynamic Mechanism Design as Bilevel Optimisation

The problem of searching for a performant mechanism within the class of AMAs can be naturally formulated as stochastic bilevel optimization as follows[4]:

$$\min_{w,b} \mathbb{E}_{\boldsymbol{r}\sim f}[\mathcal{L}(\pi^*, w, b)] \text{ s.t. } \pi^* \in \arg\max_\pi \mathbb{E}_{s_t,a_t\sim\pi}\left[\sum_{t=0}^{T}\left(\sum_{i=1}^{n} w_i r_i(s_t, a_t)\right) + b(s_t, a_t)\right] \forall \boldsymbol{r} \quad (1)$$

Given the bilevel structure, we can think of the problem as a game between a leader and a follower.

- The leader knows only the joint distribution $f = \prod_i f_i$ from which the set of reward functions are drawn, and chooses weights w_i and boosts $b(s, a)$.
- For any draw of r_i and the weights and boosts, the follower acts optimally ("best responds") in the MDP according to the AMA objective.

To clarify (and to contrast with many models of mechanism design where the agents making reports are treated as followers): the leader and follower are only "notional". In reality, there is only one mechanism designer. We nevertheless speak in terms of a "leader" and "follower" because in order for strategyproofness to be attained, some component of the system must successfully maximize affine social welfare, which is a goal distinct from the true goal of the mechanism designer.

In general, the problem above—bilevel optimization, with stochasticity in the leader's objective—is quite difficult. Indeed, the case of revenue-maximisation in one-round auctions with multiple goods, which is a very special case of our much more general problem, remains essentially unsolved beyond a few very simple special cases [62]. Therefore finding a globally optimal solution to the above problem is too much to hope for, but we show that derivatives exist for important expected loss functions, which enables us to use gradient-based optimization techniques to find local optima.

We then consider three complementary methods for optimizing the AMA parameters: random grid search, zeroth-order methods to approximate the derivatives, and differentiation through a smoothed LP.

4.1 Existence of derivatives

So far, we have not made any assumptions about the loss function. A very natural desideratum would be that $\mathcal{L}(\pi^*(w, b, \boldsymbol{r}), w, b)$ is differentiable, so that we can perform stochastic gradient descent. As we show later, this does not hold for the loss functions we consider and is unlikely to hold in general. However, we prove that the relaxed condition that $E_{\boldsymbol{r}}[\mathcal{L}(\pi^*(w, b, \boldsymbol{r}), w, b)]$ is differentiable holds for revenue—arguably the most important loss function in mechanism design literature. Therefore, for expected revenue (and other loss functions whose expected value is differentiable) we can

[4]We constrain the policy to be in the *set* of best responses, because there is a null set of possible $\boldsymbol{r}$ where π^* is not unique. However, because it is a null set, the choice of π^* does not influence the expected value in the outer problem.

approach locally optimal solutions by zeroth-order methods or differentiating through a regularized LP formulation of the MDP.

The expected revenue of a dynamic AMA mechanism is given by

$$
\begin{aligned}
\mathsf{rev}(w,b) &= \mathbb{E}_{\boldsymbol{r}}\left[\sum_{i=1}^{n}\boldsymbol{p}_i(w,b,\boldsymbol{r})\right] \\
&= \mathbb{E}_{\boldsymbol{r}}\left[-\sum_{i=1}^{n}(\frac{1}{w_i}\mathsf{asw}(w,b,\boldsymbol{r})) + \mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r}) + \sum_{i=1}^{n}\frac{1}{w_i}\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r})\right]
\end{aligned}
$$

Here sw, the non-transformed social welfare, is just asw when weights are all 1 and boosts are all 0, i.e. $\mathsf{sw}^{(\pi)} = \mathbb{E}_\pi\left[\sum_{t=0}^{T}\sum_{i=1}^{n} r_i(s_t,a_t)\right]$.

First we show that this is differentiable almost everywhere.

Theorem 4.1. *The expected revenue in a dynamic AMA mechanism is continuous and differentiable almost everywhere.*

Proof Idea. The full proof is in Appendix B. In brief, it suffices to show asw and sw are locally Lipschitz continuous. Affine social welfare is the maximum over a set of functions which are linear in w and b, so bounding the change as a result of changing weights and boosts is straightforward.

Showing local Lipschitzness of expected social welfare is more involved. One can argue that, for any w, b, the set of rewards for which two alternative policies give the same affine social welfare, lie on a hyperplane. Changing the weights and boosts may change the optimal policy, and therefore may result in differing social welfare for some subset of the rewards. We would like to bound the probablility mass of the set of rewards where this can happen. Indeed, the rewards where the optimal policy may change, all lie in between the aforementioned hyperplanes. The distance between these depends linearly on the change in w and b, so the mass can be bounded—assuming the probability density decays sufficiently fast. ☐

4.2 Linear Programming Formulation

We now describe the linear-programming formulation for an MDP, which can be either infinite horizon or episodic with states partially ordered by time (that is, the time step is encoded in the state) [3]. In particular, we suppose the follower solves the following LP to find the optimal state-action occupancy measure $\nu_{\pi^*(w,b,\boldsymbol{r})}$,[5] which determines the revenue (or whichever loss function is chosen):

$$
\begin{aligned}
&\max \sum_{s\in S, a\in A}\left(\sum_i w_i r_i(s,a) + b(s,a)\right)\nu(s,a) \text{ s.t.} \\
&\sum_{a\in A}\nu(s,a) = \sum_{s',a'} P(s|s',a')\nu(s',a') + \mu_0(s) \quad \forall s\in S \\
&\nu(s,a) \geq 0 \quad \forall s\in S, a\in A
\end{aligned} \tag{2}
$$

As we argue in Appendix C, we have the exact form of $\nabla_{w,b}\mathsf{asw}(w,b,r)$ almost everywhere. Similarly, it holds that $\nabla_{w,b}\mathsf{sw}^{\pi^*(w,b,r)}(\boldsymbol{r}) = 0$ a.e. However, $\mathsf{sw}^{\pi^*(w,b,r)}(\boldsymbol{r})$ is discontinuous and the gradient thus not defined at all $\boldsymbol{r}$, where more than one policy is optimal. This prevents us from estimating $\nabla_{w,b}\mathsf{rev}(w,b)$ from samples as we cannot exchange the gradient and the expected value. Indeed it seems that variations of this problem are likely to arise for most loss functions. When small changes in the weights and boosts lead to jumps from one deterministic policy to another, it can generally be expected that the loss function will exhibit these discontinuities as well. For this reason we propose two techniques that allow us to calculate a smoothed approximation to the gradient: zeroth-order methods, and differentiating through a regularized version of the linear program. Using either of these approximate gradients we proceed as in Algorithm 1.

[5] This corresponds to the optimal policy by $\pi^*(a|s) = \frac{\nu_{\pi^*(w,b,\boldsymbol{r})}(s,a)}{\sum_a \nu_{\pi^*(w,b,\boldsymbol{r})}(s,a)}$.

Algorithm 1 Gradient-based Dynamic Mechanism Design

Input: MDP $\mathcal{M}$, number of agents n, loss $\mathcal{L}$

```
1: Initialize: w ∈ ℝ^n, b ∈ ℝ^{|S|×|A|}
2: (w_i, b_i)_{1≤i≤10} ← grid search(M, L)
   {Multiple starting points to avoid local minima}
3: for 1 ≤ i ≤ n do
4:    for 1 ≤ j ≤ m do
5:       (w_i, b_i) ← (w_i, b_i) + γ gradientstep(M, w_i, b_i) {Zeroth or first order gradient estimate}
6:    end for
7: end for
8: return arg min_{i∈{1,...,10}} E[L(w_i, b_i)]
```

4.2.1 Zeroth-order methods

For bilevel problems, Sow et al. [57] present a zeroth-order approach, which we adapt to our own setting. The key observation of their approach is that the derivative of the leader's objective is the sum of two partial derivatives. One of these, the derivative of the leader's objective with respect to their own solution, is usually easy to evaluate. The other requires differentiating through the follower's best-response map and inverting a Jacobian, which is challenging—and this second portion can be separately estimated using zeroth-order perturbations.

Zeroth-order methods, due to the use of random perturbations, also implicitly smooth the function [24], so that when using them there is no further need for regularization to ensure that derivatives can be estimated. [6]

4.2.2 Regularized linear program

As an alternative method, we propose regularizing our problem to get a smooth surrogate of the derivative. We add a small entropy regularizer to the follower's objective. [7] The result is now a convex program (see 13 in the appendix): When the objective is strongly convex, the solution map $\nu_r^* : (w, b) \mapsto \nu_{\pi^*(w,b,\boldsymbol{r})}$ is smooth. Therefore, derivatives of revenue can now be estimated from the gradients at sampled type profiles, calculated using reverse-mode automatic differentiation [2, 1].

Adding a small amount of regularisation does not change the follower's problem significantly. As has been shown by Weed [60], the distance between the solutions to the regularized and unregularized inner problem decays exponentially fast in the regularisation constant α, for sufficiently small α.

For expected revenue we claim this convergence translates to the outer problem, so that choosing a small α ensures the objective is not disturbed too much.

Theorem 4.2 (Pointwise convergence of regularised revenue)**.** *Let $\mathsf{rev}_\alpha(w, b)$ denote the revenue achieved with the optimal policy $\pi^\alpha(w, b, \boldsymbol{r})$ for the regularised problem (see equation 13 in Appendix E). Assume that $\mathbb{E}[\|\boldsymbol{r}\|_\infty]$ exists, then $\lim_{\alpha\to 0} \mathsf{rev}_\alpha(w, b) = \mathsf{rev}(w, b)$.*

The proof of the Theorem can be found in Appendix D.

5 Experiments

5.1 Methods

We optimize AMAs in dynamic mechanism design settings on tabular MDPs. We compare three different mechanism design settings with different reward distributions, across our three proposed methods for optimizing AMA parameters.

[6] They have been proposed in certain related (static) settings [12, 38] to cope with similar problems related to nonexistence of derivatives.

[7] A technique equivalent to such regularization has also been used to deal with a similar issue in first-order computation of equilibria of non-truthful static auctions [32].

Table 1: Results for optimizing auction revenue in a sequential sales setting (n agents, m sales) with symmetric uniformly-distributed types. Standard errors were < 0.007. Runtime was < 31 hours for grid search, < 0.2 hours for zeroth-order and < 1.1 hours for first order.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	2	0.0000	0.4902	**0.4939**	0.4893	N/A
3	2	0.4999	0.6079	**0.6777**	0.6755	35.56%
4	2	0.8000	0.7977	**0.8783**	0.8328	9.79%
5	2	0.9996	1.0050	**1.0078**	0.9967	0.83%
2	3	0.0000	**0.4715**	0.3825	0.4168	N/A
3	3	0.0000	0.6107	0.6446	**0.7240**	N/A
4	3	0.6007	0.7323	0.8875	**0.9104**	51.54%
5	3	0.9988	0.7142	1.0631	**1.0743**	7.56%

Implementation Details

Grid search As a benchmark, we implement a naïve grid search algorithm. We sample 10000 different weights and boosts from a Sobol sequence [55]. For each draw we evaluate its expected performance by solving the corresponding dual LP for 2000 different random reward draws.

Zeroth-order methods We estimate derivatives using 20 perturbations, sampled from a Gaussian distribution with standard deviation 0.05 per estimate on 20 sampled type profiles. We use a learning rate of 0.1.

Regularized LP. We compute derivatives with respect to social welfare using the regularized LP, with smoothing parameter 10^{-2} except where mentioned. We solve the regularized program using MOSEK [5] and use the DiffOpt package within JuMP [35] to differentiate. As a computational heuristic, we do not compute full derivatives for asw when calculating revenue. Instead, we simply compute exact derivatives in the form given in Appendix C. For each stochastic gradient step, we sample 20 type profiles and optimize with learning rate 10^{-2}.

In all cases, when evaluating the objective, we sample 10000 type profiles and do not use regularization. Thus at evaluation time, the LP solution is exactly correct, ensuring strategyproofness. Computational details and hyperparameters are described in appendix F.

5.2 Environments and Results

Sequential Sales We begin with a simple setting in which identical items are sold sequentially to unit-demand bidders. The states consist of a record of who has received the item; the allowed actions are to sell the item to some bidder, or to no one. The welfare-maximizing mechanism thus involves giving the items to the highest bidder, but by altering the boosts, revenue can be increased by sometimes withholding the item. We consider a distribution of type profiles drawn uniformly from $[0, 1]$, with results in Table 1.

We observe that optimizing the boosts can consistently improve performance compared to VCG, especially when there are no tight supply constraints. Intuitively, if there is a large supply of goods, VCG revenue should be low, as agents do not cause much externality on other agents. By setting boosts to effectively withhold goods (like a reserve price), revenue can be increased. We also consider a distribution where agent i's type is uniformly distributed on $[0, i]$. In this setting, we also allow the bidder weights to vary, with results in Table 2. Again, we observe improved performance by optimizing the AMA parameters.

In both settings, our gradient-based approaches outperform random grid search in terms of results and runtime, in particular for the larger settings, which makes sense given the high number of dimensions.

Dynamic truthful task scheduling The next setting we consider is a dynamic version of the classic truthful task scheduling problem [41]. In the static problem, workers report the time it takes them to

Table 2: Results for optimizing auction revenue in a sequential sales setting (n agents, m sales) with asymmetric uniformly-distributed types. Standard errors were < 0.004. Runtime was < 32 hours for grid search, < 0.2 hours for zeroth-order and < 1.1 hours for first order.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	2	0.0000	0.3327	0.3302	**0.3665**	N/A
3	2	0.2348	0.3217	**0.4123**	0.4116	75.55%
4	2	0.3089	0.3328	0.4021	**0.4508**	45.95%
5	2	0.3350	0.3355	0.4348	**0.4645**	38.65%
2	3	0.0000	**0.3208**	0.2544	0.3202	N/A
3	3	0.0000	0.3304	0.3556	**0.4354**	N/A
4	3	0.2276	0.3431	0.4263	**0.4751**	108.77%
5	3	0.3193	0.2713	0.3770	**0.4976**	55.85%

Table 3: Results for minimizing makespan in the dynamic truthful task scheduling setting (n agents, m sales) with symmetric uniformly-distributed types. Standard errors were < 0.02. Runtime was < 16 hours for grid search, < 0.1 hours for zeroth-order and < 4.1 hours for first order. AMA outperforms VCG because the makespan is smaller.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	4	1.0336	1.0326	**0.9132**	0.9288	-11.65%
2	5	1.0116	1.0142	**0.8820**	0.9286	-12.81%
3	4	0.6111	0.6196	**0.5967**	0.6184	-2.36%
3	5	0.5662	0.5632	**0.5429**	0.5538	-4.12%

complete certain tasks. A mechanism then has to assign the tasks and give payments[8] to incentivize truthful reports with the goal of minimizing the makespan of all jobs.

We formulate a dynamic version of the above problem. There are n workers and T tasks. Each round, one of the tasks arrives and has to be assigned. Each worker has a cost vector $\theta_i = (t_{i,1}, \dots, t_{i,T})$ distributed according to a prior f_i, which consists of the times they take to finish each task. Each round τ the mechanism designer takes an allocative action $\boldsymbol{x}_\tau \in \{0,1\}^n$, s.t. $\sum_{i=1}^n \boldsymbol{x}_{\tau,i} = 1$. At time τ, this causes agent i to receive reward $r_{i,\tau} = -\boldsymbol{x}_{\tau,i} t_{i,\tau}$.

The objective of the leader is to minimise total makespan. For this define $\tilde{t}_i$ as the time i has already worked on its jobs, when the last task has been assigned in round T. The leader's loss is then given by $\max_{i\in\{1,\dots,n\}} \left(\left(\sum_{\tau=1}^T \boldsymbol{x}_{\tau,i} t_{i,\tau} \right) - \tilde{t}_i \right)$.

As a benchmark, we consider a dynamic VCG mechanism that chooses the solution which minimizes the total work done by the agents—an objective which is not the same as minimizing makespan. (It can been shown, however, that VCG is an n-approximation of the optimal static mechanism [41].)

[8]This in contrast to the auction environment where the mechanism could charge payments, because here agents suffer costs for which they need to be compensated.

Table 4: Results for minimizing makespan in the dynamic truthful task scheduling setting (n agents, m sales) with asymmetric uniformly-distributed types. Standard errors were < 0.03. Runtime was < 17 hours for grid search, < 0.1 hours for zeroth-order and < 4 hours for first order. AMA outperforms VCG because the makespan is smaller.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	4	1.8312	1.8260	**1.5978**	1.6121	-12.75%
2	5	1.9651	1.9837	**1.6347**	1.6886	-16.81%
3	4	1.4299	1.4426	**1.3242**	1.3243	-7.39%
3	5	1.4644	1.4729	**1.3256**	1.3629	-9.48%

Table 5: Results for maximising revenue in the gridworld environment (n agents, $m \times m$ grid). Standard errors were < 0.05. Runtime was < 30 hours for grid search, < 0.1 hours for zeroth-order and < 0.1 hours for first order.

n	m	VCG	Grid	0-order	reg. LP	Best imp.
2	3	0.7547	1.0607	1.3486	**1.5464**	104.90%
3	3	1.2812	1.4348	1.5575	**1.9251**	50.25%
4	3	1.8683	1.8729	1.8853	**2.3009**	23.15%
5	3	2.3563	2.3689	1.9951	**2.5989**	10.30%
2	4	1.0402	1.0446	1.4935	**1.6134**	55.11%
3	4	1.4898	1.4904	1.6821	**2.0171**	35.40%
4	4	1.8610	1.8757	1.9427	**2.3413**	25.81%
5	4	2.2472	2.2133	2.0256	**2.5911**	15.30%

For a symmetric valuation distribution (uniform on $[0, 3]$), results are in Table 3. We also consider an asymmetric distribution with disutilities distributed on $[0, 3i]$ for bidder i, with results in Table 4. Across all environments, we see an improvement in makespan for the best AMAs. In both settings, gradient-based approaches outperform the naïve grid search in terms of results and runtime.

Navigating a grid with multiple tasks One of the most canonical environments in RL is the *gridworld*, where an agent deterministically navigates a two-dimensional grid with rewards for reaching certain states [58]. We consider the following variant: the mechanism moves ("up","down","left","right") in a grid with n agents observing the trajectories. It starts in state s_0. Each agent i draws a goal state $s_i \in S \setminus \{s_0\}$ and a reward $r_i \sim \mathcal{U}(0, 1)$, which they receive when the mechanism reaches s_i. Given (w, b), the mechanism finds a policy to maximize affine social welfare $\pi^* \in \arg\max_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \left(\sum_{i=1}^{n} w_i r_i(s_t, a_t) + b(s_t, a_t)\right)\right]$ where γ is the discount factor to account for the infinite time horizon. One way to interpret this is an auctioneer navigating an environment with different replenishing goods, which agents wish to collect. The agents now bid to influence trajectories, and the auctioneer tries to maximize revenue. The results are in Table 5 and show that we can increase revenue by at least 10% in every setting considered. Moreover, the gradient-based approaches are an order of magnitude faster and achieve better results than grid search. We further observed that the optimized boosts correspond to a preference of the mechanism for staying close to s_0. This can be roughly interpreted as a counterpart to a reserve price in an auction.

6 Conclusion

In this paper, we have proposed an approach for automated dynamic mechanism design. In contrast to earlier work, this formulation allows for a wide array of possible objectives (not just maximising social welfare) and works without strong restrictions on the type space. In principle, it captures essentially all problems of static mechanism design as a special case. By focusing on the class of AMAs, we can frame the problem as stochastic bilevel optimization, where the mechanism designer acting in the outer problem chooses parameters to maximize their objective in expectation over possible rewards and the inner problem consists of optimally solving the MDP.

For the most prominent objective in mechanism design—expected revenue—we have further proven differentiability, which allows for gradient-based optimisation approaches to converge to local optima. By restricting to the class of AMAs, all these mechanisms are guaranteed to be exactly IC and IR. To solve the bilevel problem, we have presented randomized grid search, as well as a zeroth and first order gradient-based algorithm to find well-performing mechanisms, which can beat the benchmark dynamic VCG mechanism across a broad range of environments. Our gradient-based methods also consistently outperform naïve grid search, which suffers from the curse of dimensionality.

The method we have presented is appropriate for any problem that can be formulated as controlling an MDP in the face of possibly untruthful preferences. This covers a wide range of interesting scenarios. In particular, the use of affine maximizers and the bilevel problem formulation are applicable to a broader range of settings, including those beyond the reach of tabular methods. Future work could apply deep RL for both the leader and follower, enabling scaling to significantly larger and more complicated problems, or apply our techniques to novel mechanism design settings.

References

[1] Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers. In *Neural Information Processing Systems (NeurIPS)*, 2019.

[2] Akshay Agrawal, Shane Barratt, Stephen Boyd, Enzo Busseti, and Walaa M. Moursi. Differentiating Through a Cone Program, May 2020.

[3] Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC press, 1999.

[4] Kareem Amin, Afshin Rostamizadeh, and Umar Syed. Repeated Contextual Auctions with Strategic Buyers. In *Neural Information Processing Systems (NeurIPS)*, 2014.

[5] MOSEK ApS. *The MOSEK optimization toolbox, version 10.0*, 2023.

[6] Susan Athey and Ilya Segal. An efficient dynamic mechanism. *Econometrica*, 81(6):2463–2485, 2013.

[7] Maria-Florina Balcan, Tuomas Sandholm, and Ellen Vitercik. A general theory of sample complexity for multi-item profit maximization. In *Economics and Computation (EC)*, 2018.

[8] Maria-Florina Balcan, Dan DeBlasio, Travis Dick, Carl Kingsford, Tuomas Sandholm, and Ellen Vitercik. How much data is sufficient to learn high-performing algorithms? Generalization guarantees for data-driven algorithm design. In *ACM Symposium on Theory of Computing (STOC)*, 2021.

[9] Maria-Florina F Balcan, Tuomas Sandholm, and Ellen Vitercik. Sample complexity of automated mechanism design. In *Neural Information Processing Systems (NeurIPS)*, 2016.

[10] Dirk Bergemann and Juuso Välimäki. The dynamic pivot mechanism. *Econometrica*, 78(2): 771–789, 2010.

[11] Dirk Bergemann and Juuso Välimäki. Dynamic Mechanism Design: An Introduction. *Journal of Economic Literature*, 57(2):235–274, June 2019.

[12] Martin Bichler, Maximilian Fichtl, Stefan Heidekrüger, Nils Kohring, and Paul Sutterer. Learning equilibria in symmetric auction games using artificial neural networks. *Nature Machine Intelligence*, 3(8):687–695, 2021.

[13] Avrim Blum, Jeffrey C. Jackson, Tuomas Sandholm, and Martin Zinkevich. Preference Elicitation and Query Learning. *Journal of Machine Learning Research*, 2004.

[14] Gianluca Brero, Benjamin Lubin, and Sven Seuken. Machine learning-powered iterative combinatorial auctions. *arXiv preprint arXiv:1911.08042*, 2019.

[15] Michael Chang, Sid Kaushik, S. Matthew Weinberg, Tom Griffiths, and Sergey Levine. Decentralized reinforcement learning: Global decision-making via local economic transactions. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1437–1447. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/chang20b.html`.

[16] Edward H Clarke. Multipart pricing of public goods. *Public Choice*, pages 17–33, 1971.

[17] Wolfram Conen and Tuomas Sandholm. Preference elicitation in combinatorial auctions: Extended abstract. In *Economics and Computation (EC)*, pages 256–259, 2001. More detailed description of algorithmic aspects in IJCAI-01 Workshop on Economic Agents, Models, and Mechanisms, pp. 71–80.

[18] Vincent Conitzer and Tuomas Sandholm. Complexity of mechanism design. In *Uncertainty in Artificial Intelligence (UAI)*, 2002.

[19] Vincent Conitzer and Tuomas Sandholm. Incremental mechanism design. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[20] Michael Curry, Ping-yeh Chiang, Tom Goldstein, and John P. Dickerson. Certifying strategyproof auction networks. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[21] Michael Curry, Tuomas Sandholm, and John Dickerson. Differentiable economics for randomized affine maximizer auctions. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

[22] Michael J Curry, Uro Lyi, Tom Goldstein, and John Dickerson. Learning revenue-maximizing auctions with differentiable matching. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

[23] Zhijian Duan, Haoran Sun, Yurong Chen, and Xiaotie Deng. A Scalable Neural Network for DSIC Affine Maximizer Auction Design, May 2023.

[24] John C Duchi, Peter L Bartlett, and Martin J Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

[25] Paul Duetting, Zhe Feng, Harikrishna Narasimhan, David C. Parkes, and Sai Srivatsa Ravindranath. Optimal auctions through deep learning. In *International Conference on Machine Learning (ICML)*, 2019.

[26] Theodore Groves. Incentives in teams. *Econometrica: Journal of the Econometric Society*, pages 617–631, 1973.

[27] MohammadTaghi Hajiaghayi, Robert Kleinberg, and David C Parkes. Adaptive limited-supply online auctions. In *Electronic Commerce (EC)*, 2004.

[28] MohammadTaghi Hajiaghayi, Robert Kleinberg, and Tuomas Sandholm. Automated online mechanism design and prophet inequalities. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2007.

[29] Dmitry Ivanov, Iskander Safiulin, Ksenia Balabaeva, and Igor Filippov. Optimal-er auctions through attention. *arXiv preprint arXiv:2202.13110*, 2022.

[30] Sham M Kakade, Ilan Lobel, and Hamid Nazerzadeh. Optimal dynamic mechanism design and the virtual-pivot mechanism. *Operations Research*, 61(4):837–854, 2013.

[31] Robert Kleinberg. A Multiple-Choice Secretary Algorithm with Applications to Online Auctions. In *Symposium on Discrete Algorithms (SODA)*, 2005.

[32] Nils Kohring, Fabian R. Pieroth, and Martin Bichler. Enabling First-Order Gradient-Based Learning for Equilibrium Computation in Markets. *arXiv preprint arXiv:2303.09500*, March 2023.

[33] Sebastien M. Lahaie and David C. Parkes. Applying learning algorithms to preference elicitation. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, EC '04, New York, NY, USA, May 2004. Association for Computing Machinery.

[34] Anton Likhodedov and Tuomas Sandholm. Methods for boosting revenue in combinatorial auctions. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2004.

[35] Miles Lubin, Oscar Dowson, Joaquim Dias Garcia, Joey Huchette, Benoît Legat, and Juan Pablo Vielma. Jump 1.0: Recent improvements to a modeling language for mathematical optimization. *Mathematical Programming Computation*, 2023. In press.

[36] Boxiang Lyu, Qinglin Meng, Shuang Qiu, Zhaoran Wang, Zhuoran Yang, and Michael I. Jordan. Learning dynamic mechanisms in unknown environments: A reinforcement learning approach, 2022. URL `https://arxiv.org/abs/2202.12797`.

[37] Boxiang Lyu, Zhaoran Wang, Mladen Kolar, and Zhuoran Yang. Pessimism meets VCG: Learning dynamic mechanism design via offline reinforcement learning. *arXiv preprint arXiv:2205.02450*, 2022.

[38] Carlos Martin and Tuomas Sandholm. Finding mixed-strategy equilibria of continuous-action games without gradients using randomized policy networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

[39] Paul Milgrom and Ilya Segal. Envelope Theorems for Arbitrary Choice Sets. *Econometrica*, 70 (2):583–601, 2002.

[40] Roger B Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

[41] Noam Nisan and Amir Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35(1-2):166–196, apr 2001.

[42] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007. doi: 10.1017/CBO9780511800481.

[43] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic Game Theory*, chapter 16. Cambridge University Press, 2007. doi: 10.1017/CBO9780511800481.

[44] Mallesh Pai and Rakesh V. Vohra. Optimal dynamic auctions. Working Paper 1461, Discussion Paper, 2008.

[45] Alessandro Pavan, Ilya Segal, and Juuso Toikka. Dynamic mechanism design: A Myersonian approach. *Econometrica*, 82(2):601–653, 2014.

[46] Jad Rahme, Samy Jelassi, Joan Bruna, and S Matthew Weinberg. A permutation-equivariant neural network architecture for auction design. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[47] Jad Rahme, Samy Jelassi, and S Matthew Weinberg. Auction learning as a two-player game. In *International Conference on Learning Representations (ICLR)*, 2021.

[48] Kevin Roberts. The characterization of implementable choice rules. *Aggregation and Revelation of Preferences*, 12(2):321–348, 1979.

[49] Tuomas Sandholm. Automated mechanism design: A new application area for search algorithms. In *International Conference on Principles and Practice of Constraint Programming*, 2003.

[50] Tuomas Sandholm and Craig Boutilier. Preference elicitation in combinatorial auctions. In Peter Cramton, Yoav Shoham, and Richard Steinberg, editors, *Combinatorial Auctions*, pages 233–263. MIT Press, 2006. Chapter 10.

[51] Tuomas Sandholm and Anton Likhodedov. Automated design of revenue-maximizing combinatorial auctions. *Operations Research*, 63(5):1000–1025, 2015.

[52] Tuomas W Sandholm, Vincent Conitzer, and Craig Boutilier. Automated Design of Multistage Mechanisms. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[53] Weiran Shen, Binghui Peng, Hanpeng Liu, Michael Zhang, Ruohan Qian, Yan Hong, Zhi Guo, Zongyao Ding, Pengjun Lu, and Pingzhong Tang. Reinforcement mechanism design, with applications to dynamic pricing in sponsored search auctions. *CoRR*, abs/1711.10279, 2017.

[54] Weiran Shen, Pingzhong Tang, and Song Zuo. Automated mechanism design via neural networks. In *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2019.

[55] I.M Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(67)90144-9. URL `https://www.sciencedirect.com/science/article/pii/0041555367901449`.

[56] Ermis Soumalias, Behnoosh Zamanlooy, Jakob Weissteiner, and Sven Seuken. Machine Learning-powered Course Allocation, 2023.

[57] Daouda Sow, Kaiyi Ji, and Yingbin Liang. On the convergence theory for hessian-free bilevel algorithms. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Neural Information Processing Systems (NeurIPS)*, 2022.

[58] Richard S. Sutton. *Reinforcement learning : : an introduction*. Adaptive computation and machine learning series. The MIT Press, Cambridge, Massachusetts, second edition. edition, 2018. ISBN 0-262-35270-2.

[59] William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961.

[60] Jonathan Weed. An explicit analysis of the entropic penalty in linear programming. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 1841–1855. PMLR, 06–09 Jul 2018. URL `https://proceedings.mlr.press/v75/weed18a.html`.

[61] Jakob Weissteiner, Jakob Heiss, Julien Siems, and Sven Seuken. Bayesian Optimization-based Combinatorial Assignment. *arXiv preprint arXiv:2208.14698*, 2023.

[62] Andrew Chi-Chih Yao. Dominant-strategy versus bayesian multi-item auctions: Maximum revenue determination and comparison. In *Economics and Computation (EC)*, 2017.

[63] Martin Zinkevich, Avrim Blum, and Tuomas Sandholm. On polynomial-time preference elicitation with value queries. In *Economics and Computation (EC)*, pages 176–185, 2003.

A Affine maximizers are incentive compatible

Theorem A.1. *For any choice of fixed weights and boosts, affine maximizers are incentive compatible.*

Proof. (Following the standard structure for truthfulness of VCG in the static case.) Consider player i with true reward function r_i and reported rewards $\tilde{r}_i$, where other players have rewards r_{-i}. Recall that the payments charged for each agent can be written as

$$\begin{aligned} p_i(\boldsymbol{r}) &= \frac{1}{w_i}\left(\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r}) - \left(\mathbb{E}_{\pi^*(w,b,\boldsymbol{r})}\left[\sum_{t=0}^{T}\left(\sum_{j\neq i} w_j r_j(s_t,a_t)\right) + b(s_t,a_t)\right]\right)\right) \\ &= \frac{1}{w_i}\left(\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r}) - \mathsf{asw}(w,b,\boldsymbol{r}) + \sum_{t=0}^{T} w_i \mathbb{E}_{\pi^*(w,b,\boldsymbol{r})}[r_i(s_t,a_t)]\right) \end{aligned}$$

where $\boldsymbol{r} = (r_i, r_{-i})$. The true expected utility at the time they report, as a function of their true reward function and a possible misreport $\tilde{r}_i$, is

$$U_i(r_i,\hat{r}_i) = \mathbb{E}_{\pi^*(w,b,\tilde{r}_i,r_{-i})}\left[\sum_{t=0}^{T} r_i(s_t,a_t) - \frac{1}{w_i}\left(\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r}) - \left(\sum_{t=0}^{T}\left(\sum_{j\neq i} w_j r_j(s_t,a_t)\right) + b(s_t,a_t)\right)\right)\right]$$

where the expectation is over the randomness in the MDP (note that it does *not* need to be over the randomness in opponent types – IC should hold in dominant strategies considering the opponents).

In choosing a misreport $\tilde{r}_i$, player i thus faces a maximization problem:

$$\begin{aligned} \arg\max_{\tilde{r}_i} U_i(r_i,\tilde{r}_i) &= \arg\max_{\tilde{r}_i} \mathbb{E}_{\pi^*(w,b,\tilde{r}_i,r_{-i})}\left[\sum_{t=0}^{T} r_i(s_t,a_t) - \frac{1}{w_i}\left(\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r})\right.\right. \\ &\qquad \left.\left. - \left(\sum_{t=0}^{T}\left(\sum_{j\neq i} w_j r_j(s_t,a_t)\right) + b(s_t,a_t)\right)\right)\right] \\ &= \arg\max_{\tilde{r}_i} \mathbb{E}_{\pi^*(w,b,\tilde{r}_i,r_{-i})}\left[\sum_{t=0}^{T} r_i(s_t,a_t) + \frac{1}{w_i}\left(\sum_{t=0}^{T}\sum_{j\neq i} w_j r_j(s_t,a_t) + b(s_t,a_t)\right)\right] - c \\ &= \arg\max_{\tilde{r}_i} \mathbb{E}_{\pi^*(w,b,\tilde{r}_i,r_{-i})}\left[\sum_{t=0}^{T} w_i r_i(s_t,a_t) + \sum_{t=0}^{T}\sum_{j\neq i} w_j r_j(s_t,a_t) + b(s_t,a_t)\right] \end{aligned}$$

where c is a constant. This is exactly the objective that the mechanism attempts to maximize if the bidder reports truthfully, so the best choice they can make is to do so. □

A.1 IR and IC guarantees

In mechanism design, there is often a distinction between:

- *Ex post* – properties that hold after all types have been reported and decisions have been made
- *Ex interim* – properties that hold after a bidder has observed their own type, but before seeing other types.
- *Ex ante* – properties that hold before types have been observed.

As far as the prior distribution of agent types is concerned, all our guarantees are *ex post* – i.e. we ensure dominant-strategy incentive compatibility and ex-post IR.

In our problem, we have an additional source of randomness, the inherent randomness in the MDP itself. We thus refer to "in expectation" to refer to properties that hold when averaging over the randomness in the MDP, but ex post in the types.

Given the above choices of terminology, and given a correct policy and value estimate, our chosen mechanism and payment rules can guarantee in-expectation incentive compatibility and individual rationality.

B Proofs of continuity and differentiability a.e. for revenue loss function.

In this proof, we identify boosts and agents' rewards with vectors in $\mathbb{R}^{|S|\times|A|}$ and weights with vectors in $\mathbb{R}^n_+$. Moreover, we will assume that f is a continuous density function, according to which $\boldsymbol{r}$ is distributed and that it is sufficiently well-behaved, such that $\mathbb{E}_{\boldsymbol{r}}[\|\boldsymbol{r}\|_\infty]$ exists.

Recall that expected revenue is defined as

$$\begin{aligned}\mathsf{rev}(w,b) &= \mathbb{E}_{\boldsymbol{r}}\left[\sum_{i=1}^n p_i(\boldsymbol{r})\right]\\ &= \mathbb{E}_{\boldsymbol{r}}\left[-\left(\sum_{i=1}^n(\frac{1}{w_i}\mathsf{asw}(w,b,\boldsymbol{r})\right)+\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r})+\sum_{i=1}^n\frac{1}{w_i}\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r})\right]\end{aligned}$$

In order to prove revenue is continuous and almost everywhere differentiable, it suffices to show that $\mathbb{E}_{\boldsymbol{r}}\left[\mathsf{asw}(w,b,\boldsymbol{r})\right], \mathbb{E}_{\boldsymbol{r}}\left[\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r})\right]$ and $\mathbb{E}_{\boldsymbol{r}}\left[\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r})\right]$ are locally Lipschitz continuous in w,b. Differentiability almost everywhere then follows from Rademacher's theorem. Further note that the proofs for $\mathbb{E}_{\boldsymbol{r}}\left[\mathsf{asw}(w,b,\boldsymbol{r})\right]$ and $\mathbb{E}_{\boldsymbol{r}}\left[\mathsf{asw}^{(-i)}(w,b,\boldsymbol{r})\right]$ are identical, since one is equivalent to the other, when removing one agent. We thus restrict ourselves to proving $\mathbb{E}_{\boldsymbol{r}}\left[\mathsf{asw}(w,b,\boldsymbol{r})\right]$ and $\mathbb{E}_{\boldsymbol{r}}\left[\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r})\right]$ are locally Lipschitz continuous.

B.1 Affine social welfare is Locally Lipschitz continuous

Here we only show continuity in w, since the proof for b works analogous. Fix $\boldsymbol{r}\in\mathbb{R}^{|S|\times|A|\times n}$ and let $w,\hat{w}\in\mathbb{R}^n_+$, s.t. $\exists!\, j: w_j\neq\hat{w}_j$. We need to show that $\exists K_{\boldsymbol{r}}$, s.t.

$$\begin{aligned}|\left(\sum_{i=1}^n\sum_{t=0}^T w_i\mathbb{E}_{s_t,a_t\sim\pi}[r_i(s_t,a_t)]\right)+\sum_{t=0}^T\mathbb{E}_{s_t,a_t\sim\pi}[b(s_t,a_t)]-\left(\sum_{i=1,i\neq j}^n\sum_{t=0}^T w_i\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_i(s_t,a_t)]\right)\\ -\sum_{t=0}^T\hat{w}_j\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)]-\sum_{t=0}^T\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[b(s_t,a_t)]|\leq K_{\boldsymbol{r}}|w_j-\hat{w}_j|\end{aligned}\tag{3}$$

where $\pi=\pi^*(w,b,\boldsymbol{r})$ and $\hat{\pi}=\pi^*(\hat{w},b,\boldsymbol{r})$. To perform our proof we need the following inequalities:

By optimality of π

$$\begin{aligned}&\sum_{i=1,i\neq j}^n(\sum_{t=0}^T w_i\mathbb{E}_{s_t,a_t\sim\pi}[r_i(s_t,a_t)])+\sum_{t=0}^T w_j\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)]+\sum_{t=0}^T\mathbb{E}_{s_t,a_t\sim\pi}[b(s_t,a_t)]\\ \geq&\sum_{i=1,i\neq j}^n(\sum_{t=0}^T w_i\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_i(s_t,a_t)])+\sum_{t=0}^T w_j\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)]+\sum_{t=0}^T\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[b(s_t,a_t)]\end{aligned}\tag{4}$$

By optimality of $\hat{\pi}$

$$\begin{aligned}&\sum_{i=1,i\neq j}^n(\sum_{t=0}^T w_i\mathbb{E}_{s_t,a_t\sim\pi}[r_i(s_t,a_t)])+\sum_{t=0}^T \hat{w}_j\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)]+\sum_{t=0}^T\mathbb{E}_{s_t,a_t\sim\pi}[b(s_t,a_t)]\\ \leq&\sum_{i=1,i\neq j}^n(\sum_{t=0}^T w_i\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_i(s_t,a_t)])+\sum_{t=0}^T \hat{w}_j\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)]+\sum_{t=0}^T\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[b(s_t,a_t)]\end{aligned}\tag{5}$$

By assumption on $w, \hat{w}_j$

$$\sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\pi}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} w_j\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\pi}[b(s_t,a_t)]$$
$$-\left(\sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\pi}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} \hat{w}_j\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\pi}[b(s_t,a_t)]\right)$$
$$\leq |w_j - \hat{w}_j|\sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)] \tag{6}$$

$$\sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} w_j\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[b(s_t,a_t)]$$
$$-\left(\sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} \hat{w}_j\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[b(s_t,a_t)]\right)$$
$$\leq |w_j - \hat{w}_j|\sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)] \tag{7}$$

Now we have all the ingredients to show local Lipschitz continuity wrt to w_i. Let $K_{\boldsymbol{r}} = \|\boldsymbol{r}\|_\infty T \geq \max(\sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)], \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)])$. Then we have

$$\sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\pi}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} w_j\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\pi}[b(s_t,a_t)] + K_{\boldsymbol{r}}|w_j - \hat{w}_j|$$
$$\underset{Eq.(4)}{\geq} \sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} w_j\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[b(s_t,a_t)] + K_{\boldsymbol{r}}|w_j - \hat{w}_j|$$
$$\underset{Eq.(7)}{\geq} \sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} \hat{w}_j\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\hat{\pi}}[b(s_t,a_t)]$$
$$\underset{Eq.(5)}{\geq} \sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\pi}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} \hat{w}_j\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\pi}[b(s_t,a_t)]$$
$$\underset{Eq.(6)}{\geq} \sum_{i=1,i\neq j}^{n}(\sum_{t=0}^{T} w_i\mathbb{E}_{s_t,a_t\sim\pi}[r_i(s_t,a_t)]) + \sum_{t=0}^{T} w_j\mathbb{E}_{s_t,a_t\sim\pi}[r_j(s_t,a_t)] + \sum_{t=0}^{T}\mathbb{E}_{s_t,a_t\sim\pi}[b(s_t,a_t)] - K_{\boldsymbol{r}}|w_j - \hat{w}_j|$$

To conclude this proof, we note

$$\mathbb{E}_{\boldsymbol{r}}[\mathsf{asw}(w,b,\boldsymbol{r}) - \mathsf{asw}(\hat{w},b,\boldsymbol{r})] \leq T\mathbb{E}_{\boldsymbol{r}\sim f}[\|\boldsymbol{r}\|_\infty]|w_j - \hat{w}_j|$$

and $\mathbb{E}_{\boldsymbol{r}\sim f}[\|\boldsymbol{r}\|_\infty]$ is finite by assumption. Differentiability almost everywhere now follows from Rademacher's theorem.

B.2 Expected social welfare is Locally Lipschitz continuous

Unlike for affine social welfare, we cannot argue that social welfare is Lipschitz continuous. Indeed changing the weights and boosts only slightly can cause a completely different policy to become optimal, leading to a discontinuous jump or drop in social welfare. Indeed this behaviour is generally expected for any loss function. However, as we will show, when taking the expected value, these discontinuities get smoothed out, guaranteeing local Lipschitz continuity and thereby differentiability almost surely.

We will restrict to proving Lipschitz continuity of $\mathbb{E}_{\boldsymbol{r}}\left[\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r})\right]$ with respect to b, as the proof with respect to w works similar.

In the proofs below we identify the reported reward functions of all agents with vectors $\boldsymbol{r} \in \mathbb{R}^d$, where $d = |S|\times|A|\times n$.

Our first observation is that all vectors $\boldsymbol{r}$ for which two policies π_1, π_2 give the same affine social welfare lie on a hyperplane. Indeed, let ν_i denote the induced state-action occupancy measure of policy π_i. Then

$$\sum_{s,a} \nu_1(s,a)(w_i r_i(s,a) + b(s,a)) = \sum_{s,a} \nu_2(s,a)(w_i r_i(s,a) + b(s,a))$$

is equivalent to

$$\sum_{s,a,i} (\nu_1(s,a) - \nu_2(s,a)) w_i r_i(s,a) = \sum_{s,a} (\nu_2(s,a) - \nu_1(s,a))(b(s,a))$$

Let $\boldsymbol{\nu}_1 = (w_1\nu_1(s_1,a_1), w_2\nu_1(s_1,a_1), \ldots, w_n\nu_1(s_1,a_1), w_1\nu_1(s_2,a_1), \ldots, w_n\nu_1(s_{|S|}, a_{|A|})$. Then the above is equivalent to the following hyperplane:

$$\mathcal{H}_{12}(w,b) = \{\boldsymbol{r} : (\boldsymbol{\nu}_1 - \boldsymbol{\nu}_2)^T \boldsymbol{r} = \sum_{s,a} (\nu_2(s,a) - \nu_1(s,a))(b(s,a))\}$$

Let $b, \tilde{b} \in \mathbb{R}^{|S|\times|A|}$, s.t. $\exists!\,(s',a') : b(s',a') \neq \tilde{b}(s',a')$. $\tilde{b}$ gives us another hyperplane of equivalence between π_1, π_2

$$\mathcal{H}_{12}(w,\tilde{b}) = \{\boldsymbol{r} : (\boldsymbol{\nu_1} - \boldsymbol{\nu_2})^T \boldsymbol{r} = \sum_{s,a} (\nu_2(s,a) - \nu_1(s,a))(\tilde{b}(s,a))\}$$

Note that $\mathcal{H}_{12}(w,\tilde{b})$ and $\mathcal{H}_{12}(w,b)$ are parallel with a distance $\frac{|(\nu_2(s',a') - \nu_1(s',a'))(b(s',a') - \tilde{b}(s',a'))|}{\|(\boldsymbol{\nu_1} - \boldsymbol{\nu_2})\|}$.

Moreover, for any given parameters (w,b) there are only $|\Pi|^2$ planes of equivalence, where $|\Pi|$ is the number of deterministic policies.

When we change $b(s',a')$ to $\tilde{b}(s',a')$, then there can be rewards $\boldsymbol{r}$, for which $\pi^*(w,\tilde{b},r) \neq \pi^*(w,b,r)$. For these rewards it follows that in general the social welfare $\mathsf{sw}^{(\pi^*(w,\tilde{b},\boldsymbol{r}))}(\boldsymbol{r})$ is different from $\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r})$. Using the hyperplanes defined above, we know the set of all $\boldsymbol{r}$, where $\mathsf{sw}^{(\pi^*(w,\tilde{b},\boldsymbol{r}))}(\boldsymbol{r}) \neq \mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r})$ is contained in the set

$$U(w,b,\tilde{b}) = \bigcup_{\pi_j,\pi_i \in \Pi} U(w,b,\tilde{b})_{ij}$$

where

$$\begin{aligned} U(w,b,\tilde{b})_{ij} = \{\boldsymbol{r} : & |(\boldsymbol{\nu_i} - \boldsymbol{\nu_j})^T \boldsymbol{r} - \sum_{s,a} (\nu_j(s,a) - \nu_i(s,a))(b(s,a))| \leq \\ & |(\nu_j(s',a') - \nu_i(s',a'))(\tilde{b}(s',a') - b(s',a'))|\} \end{aligned}$$

are the polytopes induced by the hyperplanes $\mathcal{H}_{ij}(w,\tilde{b})$ and $\mathcal{H}_{ij}(w,b)$.

With this in mind let us make a first naive analysis of the difference of expected social welfare, when changing b.

$$
\begin{aligned}
&|\mathbb{E}_{\boldsymbol{r}}[\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r})] - \mathbb{E}_{\boldsymbol{r}}[\mathsf{sw}^{(\pi^*(w,\tilde{b},\boldsymbol{r}))}(\boldsymbol{r})]| && (8)\\
&\leq E_{\boldsymbol{r}}[|\mathsf{sw}^{(\pi^*(w,b,\boldsymbol{r}))}(\boldsymbol{r}) - \mathsf{sw}^{(\pi^*(w,\tilde{b},\boldsymbol{r}))}(\boldsymbol{r})|] && (9)\\
&\leq 2Tn \int_{U(w,b,\tilde{b})} \|\boldsymbol{r}\|_\infty f(\boldsymbol{r}) d\boldsymbol{r} && (10)\\
&\leq \sum_{\pi_i,\pi_j\in|\Pi|} 2Tn \int_{U(w,b,\tilde{b})_{ij}} \|\boldsymbol{r}\|_\infty f(\boldsymbol{r}) d\boldsymbol{r} && (11)\\
& && (12)
\end{aligned}
$$

where for the second inequality we use $\mathsf{sw}^{(\pi)}(\boldsymbol{r}) = \sum_{i=1}^n (\sum_{t=0}^T \mathbb{E}_{s_t,a_t\sim\pi} r_i(s_t,a_t)) \leq Tn\|\boldsymbol{r}\|_\infty$. We want to show that the above can be bounded by $L|b(s',a') - \tilde{b}(s',a')|$ for some L. For this we need to get a better understanding of $\int_{U(w,b,\tilde{b})_{ij}} \|\boldsymbol{r}\|_\infty f(\boldsymbol{r}) d\boldsymbol{r}$.

For the sake of simplicity, we assume now the probability density f has compact support on R^d, i.e. there exists a K such that $\forall \boldsymbol{r} : \|\boldsymbol{r}\|_2 \geq K \implies f(r) = 0$.[9] Since the hyperplanes $\mathcal{H}_{ij}(w,\tilde{b})$ and $\mathcal{H}_{ij}(w,b)$ have dimension $d-1$ and are parallel with distance $\frac{|(\nu_2(s',a')-\nu_1(s',a'))(b(s',a')-\tilde{b}(s',a'))|}{\|(\boldsymbol{\nu_1}-\boldsymbol{\nu_2})\|}$, we can bound the integral by multiplying an upper bound of the volume of $U(w,b,\tilde{b})_{ij}$ with the maximum possible value of $\|\boldsymbol{r}\|_\infty f(\boldsymbol{r})$.

$$
\begin{aligned}
\int_{U(w,b,\tilde{b})_{ij}} \|\boldsymbol{r}\|_\infty f(\boldsymbol{r}) d\boldsymbol{r} \leq (2K)^{d-1} \frac{|(\nu_2(s',a') - \nu_1(s',a'))(b(s',a') - \tilde{b}(s',a'))|}{\|(\boldsymbol{\nu_1} - \boldsymbol{\nu_2})\|} K \max_{\boldsymbol{r}} f(\boldsymbol{r})\\
= L|(b(s',a') - \tilde{b}(s',a'))|
\end{aligned}
$$

for $L = (2K)^d \frac{|(\nu_2(s',a')-\nu_1(s',a'))|}{\|(\boldsymbol{\nu}_1-\boldsymbol{\nu}_2)\|} \max_{\boldsymbol{r}} f(\boldsymbol{r})$, which proves Lipschitz continuity and thereby differentiability almost surely.

C Gradients of affine social welfare

We note that for $\mathsf{asw}(w,b,r)$ the gradients can be computed in a straight-forward manner.[10] For this, rewrite asw using the state-action occupancy measure ν_π. [11] Let Π be the finite set of deterministic policies.[12] We have:

$$
\mathsf{asw}(w,b,r) = \max_{\pi\in\Pi} \sum_{s,a} \nu_\pi(s,a)(\sum_{i=1}^n w_i r_i(s,a) + b(s,a))
$$

As we are taking the maximum over a finite set of functions which are differentiable in w, b, we can apply a multi-dimensional version of the envelope theorem [39] to get

$$
\nabla_{w,b}\mathsf{asw}(w,b,r) = \nabla_{w,b} \sum_{s,a} \nu_{\pi^*(w,b,r)}(s,a)(\sum_{i=1}^n w_i r_i(s,a) + b(s,a)) \quad a.e.
$$

and hence almost everywhere

$$
\nabla_{w_i}\mathsf{asw}(w,b,r) = \sum_{s,a} \nu_{\pi^*(w,b,r)}(s,a) r_i(s,a) = \mathbb{E}_{\pi^*(w,b,r)}[\sum_{t=0}^T r_i(s,a)]
$$

[9]This assumption is not necessary. As long as f decays sufficiently quickly for large $\boldsymbol{r}$, the proof still goes through with some minor adjustments. However, we make the assumption here for streamlining our exposition and highlighting the parts of our proof, which are non-standard.

[10]The analysis holds equivalently for $\mathsf{asw}_{-i}(w,b,r)$

[11]In general finite horizon MDPs this would be defined as $\nu_\pi(s,a) = \sum_{t=1}^T \mathbb{P}_\pi(s_t = s, a_t = a)$. In our experiments we always assume the states contain the current timestep such that this simplifies to $\nu_\pi(s,a) = \mathbb{P}_\pi(s_t = s, a_t = a)$

[12]The optimal follower policy will always be deterministic.

$$\nabla_{b(s,a)}\mathsf{asw}(w,b,r) = \nu_{\pi^*(w,b,r)}(s,a)$$

Boosting a state increases asw in relation to how often the state is visited under the optimal policy. Similarly increasing the weight of an agent changes asw in proportion to the expected sum of rewards the agent gets. In fact, ν_{π^*} is the solution to the linear programming formulation of the MDP.

D Convergence of regularised revenue

Proof. Fix $w, b, \boldsymbol{r}$ and denote by $\mathsf{rev}_\alpha(w,b,\boldsymbol{r})$, $\mathsf{rev}(w,b,\boldsymbol{r})$ the regularised and unregularised revenue for this specific choice of variables. We first show

$$\lim_{\alpha\to 0}\mathsf{rev}_\alpha(w,b,\boldsymbol{r}) = \mathsf{rev}(w,b,\boldsymbol{r})$$

Note that since revenue is a linear combination of asw,asw_{-i} and sw, it suffices WLOG to show pointwise convergence of these terms. Here we restrict to proving this for asw.

Let ν^*, ν^α be the corresponding state-action measures to π^*, π^α—the optimal policies for the unregularised and regularised LP—and define $R(s,a) = \sum_{i=1}^n r_i(s,a) + b(s,a)$. Then using Corollary 9 of Weed [60], for sufficiently small α:

$$\begin{aligned}
&|\mathsf{asw}(w,b,\boldsymbol{r},\pi^*(w,b,\boldsymbol{r})) - \mathsf{asw}(w,b,\boldsymbol{r},\pi^\alpha(w,b,\boldsymbol{r}))| \\
&= \left|\sum_{s,a}(\nu^*(s,a)-\nu^\alpha(s,a))\left(\sum_{i=1}^n r_i(s,a)+b(s,a)\right)\right| \\
&= |\langle \nu^*-\nu^\alpha, R\rangle| \\
&\le \|\nu^*-\nu^\alpha\|_1\|R\|_\infty \\
&\le 2T\exp\left(-\frac{\Delta(\boldsymbol{r})}{\alpha R_1}+\frac{R_1+R_H}{R_1}\right)(\|w\|_\infty\|\boldsymbol{r}\|_\infty+\|b\|_\infty)
\end{aligned}$$

where R_1 is the l_1 radius of all feasible solutions, R_H is the entropic radius, and Δ the suboptimality gap [60]. For any fixed w, b we can now prove the statement of our theorem, by using the dominated convergence theorem. Using the same argument as before we restrict to doing so for asw. Note that

$$\mathsf{asw}(w,b,\boldsymbol{r},\pi^\alpha(w,b,\boldsymbol{r})) \le 2Tn(\|w\|_\infty\|\boldsymbol{r}\|_\infty+\|b\|_\infty)$$

which by assumption is integrable and thus by dominated convergence it holds that

$$\forall w,b: \quad \lim_{\alpha\to 0}\mathsf{rev}_\alpha(w,b) = \mathsf{rev}(w,b)$$

□

E Regularized MDP Linear Program

Below we give the regularized form of the MDP linear program – it is now a convex (exponential cone) program.

$$\begin{aligned}
&\max\sum_{s\in S,a\in A}\left(\sum_i w_i r_i(s,a)+b(s,a)\right)\nu(s,a)+ \\
&\quad\alpha\sum_{s\in S,a\in A}\nu(s,a)\log\nu(s,a)\quad \text{s.t.} \\
&\sum_{a\in A}\nu(s,a) = \sum_{s',a'}P(s|s',a')\nu(s',a')+\mu_0(s)\quad \forall s\in S \\
&\nu(s,a)\ge 0\quad \forall s\in S, a\in A
\end{aligned}\tag{13}$$

F Computational Details and Hyperparameters

The grid search experiments and all experiments in the gridworld environment were run concurrently on a server with 256 cores and 250GB of RAM, while restricting the number of threads in MOSEK to 4. Other experiments were run on cluster nodes with 4 cores and 1GB or 2GB of RAM per core, except that the task scheduling regularized LP jobs with 3 agents were run with 16 cores and 64GB memory. During development we experimented with up to 1000 sampled valuation profiles, up to 2000 perturbations, learning rates ranging from 0.001 to 0.1, and regularization strengths up to 0.1; we quickly settled on the chosen hyperparameters and did not do a more exhaustive search due to computational constraints. For distributions where the bidder valuations are symmetric, we optimize only boosts, fixing the weights to 1.