LEARNING REVENUE-MAXIMIZING AUCTIONS WITH NEURAL AFFINE MAXIMIZER

Yunxuan Ma
CFCS, School of Computer Science
Peking University
{yunxuanma}@pku.edu.cn

ABSTRACT

Learning truthful, revenue-maximizing auctions is a central challenge in automated mechanism design and differentiable economics. Existing learning approaches that guarantee truthfulness typically discretize the outcome space into a finite menu, which tends to favor deterministic but suboptimal auctions. In this work, we propose *Neural Affine Maximizer* (NAM), a discretization-free approach for learning truthful auctions. NAM guarantees truthfulness by building on affine maximizer auctions (AMAs) while replacing the conventional finite menu with a boosting function over the outcome space. NAM then parameterizes the boosting function with neural networks and derives unbiased gradient estimators to enable first-order optimization. Experiments show that NAM consistently improves revenue over state-of-the-art baselines. In the 2-bidder 2-item setting, NAM discovers a randomized, truthful auction with a 2.4% revenue improvement over known optimal deterministic, truthful auctions. In larger-scale settings up to 10 buyers or 30 goods, NAM continues to achieve revenue gains over existing approaches with comparable computation costs. Our codes are available at https://github.com/YunxuanMaPKU/NAM.

1 INTRODUCTION

Designing a *truthful* (DSIC, strategy-proof) auction that achieves high expected revenue is a central problem in mechanism design. While the single-item case admits an analytical optimal solution (Myerson, 1981), the multi-bidder, multi-item setting remains notoriously challenging (Sandholm & Likhodedov, 2015; Likhodedov & Sandholm, 2004), with optimal solutions known only in special cases (Manelli & Vincent, 2006; Pavlov, 2011; Giannakopoulos & Koutsoupias, 2014; Daskalakis et al., 2015; Yao, 2017). This gap motivates data-driven approaches for learning truthful mechanisms that improve revenue over standard mechanisms.

Recently, differentiable economics—a technical paradigm (Dütting et al., 2024; Duan et al., 2023a; Wang et al., 2024a; Ma et al., 2025a; Ravindranath et al., 2021; Zheng et al., 2022)— treats economic solution design as an end-to-end learning problem with neural networks and gradient-based optimization. Within this paradigm, notable progress has been made in auction learning (for few examples, Dütting et al. (2024); Curry et al. (2023); Duan et al. (2023b); Wang et al. (2024b); Ma et al. (2025b)).

However, existing learning methods for truthful, multi-bidder auctions either rely on *discretization* of the outcome space (Wang et al., 2024b; Curry et al., 2023; Duan et al., 2023b) or apply only to restrictive settings (*e.g.*, goods with unlimited supply (Ma et al., 2025b)). The discretization step leads to inefficiency in two main aspects: First, the number of deterministic outcomes grows combinatorially with the number of goods and buyers, leading to representation and computational burdens for large auctions (Ma et al., 2025b). Second, the set of randomized outcomes is far larger than the set of deterministic outcomes, leading discretization-based approaches prone to learn deterministic but suboptimal auctions (Duan et al., 2023b).

In this paper, we propose **Neural Affine Maximizer** (NAM), a discretization-free algorithm for learning truthful auctions. NAM guarantees its truthfulness by building upon the Affine Maximizer Auctions (AMAs), a truthful yet still expressive auction class (Roberts, 1979) widely used in auction

learning (Curry et al., 2023; Duan et al., 2023b; Sandholm & Likhodedov, 2015). Unlike existing approaches that discretize AMA directly, NAM replaces the discrete menu representation with a *continuous* boosting function $\lambda(x)$ over the outcome space. Learning an optimal boosting function is nontrivial, as existing techniques for the discrete case (Curry et al., 2023; Duan et al., 2023b; Wang et al., 2024b) do not directly generalize to the continuous case.

Our experiments show that NAM consistently outperforms existing discretization-based approaches across auction settings from small to large scale, with up to 10 buyers or 30 goods. Surprisingly, in the 2-buyer, 2-item setting with independent $U([0,1])$ valuations, where no closed-form optimal auction is known, NAM discovers a randomized truthful auction with a 2.4% revenue improvement (0.8995) over the best previously known optimal deterministic truthful auction (0.878 in (Wang et al., 2024b)). [1]

2 PRELIMINARY

Auction Model We study a standard multi-buyer, multi-good auction with one seller (the auctioneer), n buyers, and m goods. Buyer $i \in [n]$ is characterized by a private valuation vector $t_i \in \mathcal{T}_i \subseteq \mathbb{R}^m$, where the coordinate t_{ij} represents her value for good j. Let $\mathcal{T} \coloneqq \times_{i\in[n]}\mathcal{T}_i$ denote the set of type profiles.

The auctioneer implements a *direct mechanism*: given a reported type profile $\boldsymbol{t}' = \{t'_i\}_{i\in[n]} \in \mathcal{T}$, it returns (i) an outcome $\boldsymbol{x} \in \mathcal{X}$ and (ii) a monetary transfer vector $\boldsymbol{p} = (p_1, \ldots, p_n) \in \mathbb{R}^n$. We take the feasible outcome space to be $\mathcal{X} \coloneqq (\Delta_n)^m$, where $\Delta_n \coloneqq \{y \in \mathbb{R}^n_+ : \sum_{i\in[n]} y_i \le 1\}$ is the n-simplex; Δ_n represents that each good cannot be allocated more than once among n buyers. $\boldsymbol{x}$'s i,j-th element, x_{ij}, represents the probability that good j is allocated to buyer i, and p_i represents the monetary transfer from buyer i to the auctioneer. We write $x_i = (x_{i1}, \ldots, x_{im}) \in \mathcal{X}_i \coloneqq [0,1]^m$ for buyer i's allocation.

Following Dütting et al. (2024); Curry et al. (2023); Duan et al. (2023b), buyer i has an additive, quasi-linear utility on her allocation x_i, monetary transfer p_i and her type t_i, i.e., $u_i(x_i, p_i; t_i) = \langle x_i, t_i\rangle - p_i$. The auctioneer is revenue-maximizing, i.e., her utility is $\sum_i p_i$, which equals the auctioneer's revenue from the auction. Because types are private information to the auctioneer, she only knows a prior distribution $\mathcal{F} \in \Delta(\mathcal{T})$ over type profiles and seeks to maximize her expected revenue under $\mathcal{F}$. Overall, an auction instance is specified by the tuple $\mathcal{A} = (n, m, \mathcal{F}, \mathcal{T})$, and the auctioneer's goal is to design a truthful (IC or strategyproof) and individually rational (IR) direct mechanism that maximizes her expected revenue.

Definition 2.1 (Direct Mechanisms)**.** A direct mechanism $M^d = \left(\boldsymbol{x}^d, \boldsymbol{p}^d\right)$ consists of an allocation rule $\boldsymbol{x}^d : \mathcal{T} \to \mathcal{X}$ and a payment rule $\boldsymbol{p}^d : \mathcal{T} \to \mathbb{R}^n$. Given a reported type profile $\boldsymbol{t}'$, a direct mechanism M^d implements the outcome $\boldsymbol{x}^d(\boldsymbol{t}') \in \mathcal{X}$ and monetary transfer $\boldsymbol{p}^d(\boldsymbol{t}') \in \mathbb{R}^n$. For each buyer i with true type t_i, she realizes her allocation $x^d_i(\boldsymbol{t}')$ and the monetary transfer $p^d_i(\boldsymbol{t}')$, thus yielding utility $u_i(x^d_i(\boldsymbol{t}'), p^d_i(\boldsymbol{t}'); t_i)$.

Definition 2.2 (Truthful and IR Direct Mechanisms)**.** A direct mechanism $M^d = (\boldsymbol{x}^d, \boldsymbol{p}^d)$ is truthful if reporting the true type is a dominant strategy for each buyer i, regardless of the reported types $\boldsymbol{t}'_{-i}$ of all other buyers. M^d is IR if each buyer i weakly prefers participating in the auction to nonparticipation ($x_i = \mathbf{0}, p_i = 0$, representing no allocation and no payment) in every case. The properties of truthfulness and IR can be expressed by the following two conditions for all $i \in [n]$, buyer i's type $t_i \in \mathcal{T}_i$, and buyer $-i$'s reported types $\boldsymbol{t}'_{-i} \in \mathcal{T}_{-i}$:

$$u_i(x^d_i(t_i, \boldsymbol{t}'_{-i}), p^d_i(t_i, \boldsymbol{t}'_{-i}); t_i) \ge 0, \qquad \text{(IR)}$$

$$u_i(x^d_i(t_i, \boldsymbol{t}'_{-i}), p^d_i(t_i, \boldsymbol{t}'_{-i}); t_i) \ge u_i(x^d_i(t'_i, \boldsymbol{t}'_{-i}), p^d_i(t'_i, \boldsymbol{t}'_{-i}); t_i). \qquad \forall t'_i \in \mathcal{T}_i \quad \text{(IC)}$$

From now on, we refer to truthful and IR direct mechanisms simply as truthful mechanisms when the context is clear. Denote $\mathcal{M}^d$ as the set of all direct mechanisms. The auctioneer's goal for an auction problem $\mathcal{A}$ can be stated as solving Program (1).

$$\max_{(\boldsymbol{x}^d, \boldsymbol{p}^d) = M^d \in \mathcal{M}^d} \ \mathbb{E}_{\boldsymbol{t}\sim\mathcal{F}}\left[\sum_{i\in[n]} p^d_i(\boldsymbol{t})\right] \qquad \text{s.t. (IC), (IR).} \tag{1}$$

[1] Further related works are deferred to Appendix A.

Affine Maximizer Auctions (AMA) An Affine Maximizer Auction is represented by $M^{\text{AMA}} = \{(\boldsymbol{x}^k, \lambda^k)\}_{k\in[M]}$ where $M \in \mathbb{N}$, $\boldsymbol{x}^k \in \mathcal{X}$, and $\lambda^k \in \mathbb{R}$ for all $k \in [M]$. Here, $\boldsymbol{x}^k$ denotes a candidate outcome and λ^k denotes the boosting value *w.r.t.* $\boldsymbol{x}^k$.

An AMA M^{AMA} can behave like a direct mechanism, *i.e.*, given $\boldsymbol{t} \in \mathcal{T}$ as input, M^{AMA} can output an outcome $\boldsymbol{x} \in \mathcal{X}$ and a monetary transfer $\boldsymbol{p} \in \mathbb{R}^n$. Rigorously, we define an "equivalence" relation between direct mechanisms and AMA mechanisms. Specifically, fix an AMA $M^{\text{AMA}} = \{(\boldsymbol{x}^k, \lambda^k)\}_{k\in[M]}$. There always exists an equivalent direct mechanism $M^d = (\boldsymbol{x}^d, \boldsymbol{p}^d)$ that computes the outcome $\boldsymbol{x}^d(\boldsymbol{t})$ and monetary transfer $\boldsymbol{p}^d(\boldsymbol{t})$ through the following steps given $\boldsymbol{t} \in \mathcal{T}$ as input:

$$\begin{aligned}
&\text{ASW}(k;\boldsymbol{t}) \coloneqq \sum_i \langle t_i, x_i^k\rangle + \lambda^k, && \text{ASW}_{-i}(k;\boldsymbol{t}) \coloneqq \sum_{j\neq i} \langle t_i, x_i^k\rangle + \lambda^k,\\
&k^{-i,*}(\boldsymbol{t}) \in \arg\max_k \text{ASW}_{-i}(k;\boldsymbol{t}), && k^*(\boldsymbol{t}) \in \arg\max_k \text{ASW}(k;\boldsymbol{t}),\\
&\boldsymbol{x}^d(\boldsymbol{t}) = \boldsymbol{x}^{k^*(\boldsymbol{t})}, && p_i^d(\boldsymbol{t}) = \text{ASW}_{-i}(k^{-i,*}(\boldsymbol{t});\boldsymbol{t}) - \text{ASW}_{-i}(k^*(\boldsymbol{t});\boldsymbol{t})
\end{aligned} \tag{2}$$

where $\text{ASW}(k;\boldsymbol{t})$ denotes the affine social welfare of outcome $\boldsymbol{x}^k$. Note that the equivalent direct mechanism M^d for a given AMA M^{AMA} may be non-unique because of the $\arg\max$ operations.

Remark 2.3. *It is important to note that a more general form of AMA has an additional parameter set $\{w_i\}_{i\in[n]}, w_i > 0$, where w_i represents the weight of buyer i. The affine social welfare is then redefined as $\text{ASW}(k;\boldsymbol{t}) \coloneqq \sum_{i\in[n]} w_i \cdot \langle t_i, x_i^k\rangle + \lambda^k$ and $p_i^d(\boldsymbol{t}) = w_i^{-1}\left(\text{ASW}_{-i}(k^{-i,*}(\boldsymbol{t});\boldsymbol{t}) - \text{ASW}_{-i}(k^*(\boldsymbol{t});\boldsymbol{t})\right)$.*
In this work, we simply let $w_i \equiv 1$ for two main reasons: a) learning additional w_is seems orthogonal to our approach; and b) when buyers are symmetric (as in our experiments), it is natural to impose symmetric weights as an inductive bias. It will be straightforward to incorporate w_is as learnable parameters.

A well-known result about AMA is that if M^{AMA} and M^d are equivalent, then M^d is truthful.

Proposition 2.4. (Truthfulness of AMA). *Let M^{AMA} be an AMA mechanism and M^d be a direct mechanism equivalent to M^{AMA}, then M^d is truthful.*

For completeness, proof of Proposition 2.4 (as well as other theoretical statements in this paper) is presented in Appendix B.

3 Neural Affine Maximizer

This section introduces our novel concept of Neural Affine Maximizer (NAM), that defines an AMA functionally rather than enumeratively as in Section 2.

3.1 Definition

An NAM is represented by a continuous boosting function $\lambda^{\text{NAM}} : \mathcal{X} \to \mathbb{R}$. Similar to an AMA, an NAM can also be viewed as a direct mechanism. More precisely, fix an NAM λ^{NAM}, there is an equivalent direct mechanism $M^d = (\boldsymbol{x}^d, \boldsymbol{p}^d)$ that computes the outcome $\boldsymbol{x}^d(\boldsymbol{t})$ and monetary transfer $\boldsymbol{p}^d(\boldsymbol{t})$ via the following steps for a given type profile $\boldsymbol{t}$:

$$\begin{aligned}
&\boldsymbol{t}[-i] \coloneqq (t_1, ..., t_{i-1}, \mathbf{0}_m, t_{i+1}, ..., t_n), && \text{ASW}(\boldsymbol{x};\boldsymbol{t}) \coloneqq \sum_{i\in[n]} \langle t_i, x_i\rangle + \lambda^{\text{NAM}}(\boldsymbol{x}),\\
&\boldsymbol{x}^d(\boldsymbol{t}) \in \arg\max_{\boldsymbol{x}\in\mathcal{X}} \text{ASW}(\boldsymbol{x};\boldsymbol{t}), && \boldsymbol{x}^{-i}(\boldsymbol{t}) \in \arg\max_{\boldsymbol{x}\in\mathcal{X}} \text{ASW}(\boldsymbol{x};\boldsymbol{t}[-i]),\\
&p_i^d(\boldsymbol{t}) = \text{ASW}(\boldsymbol{x}^{-i}(\boldsymbol{t});\boldsymbol{t}[-i]) - \text{ASW}(\boldsymbol{x}^d(\boldsymbol{t});\boldsymbol{t}[-i])
\end{aligned} \tag{3}$$

where $\mathbf{0}_m$ represents an m-dimensional zero vector. Note that the $\arg\max$ is well defined by continuity of $\lambda^{\text{NAM}}(\boldsymbol{x})$ (consequently $\text{ASW}(\boldsymbol{x};\boldsymbol{t})$) and compactness of $\mathcal{X}$.

3.2 Connection between NAM and AMA

We establish the truthfulness of NAM in Proposition 3.1.

Proposition 3.1. (Truthfulness of NAM) *Let λ^{NAM} be a Neural Affine Maximizer and M^d be a direct mechanism that is equivalent to λ^{NAM}. Then M^d is truthful.*

Proposition 3.1 is quite straightforward if one regards NAMs as AMAs with uncountably many candidate outcomes, together with a proper topological refinement of the corresponding boosting values. Next, we establish the equivalence between NAMs and AMAs in Theorem 3.2.

Theorem 3.2. *Fix $M^{AMA} = \{(\boldsymbol{x}^k, p^k)\}_{k\in[M]}$ to be an arbitrary AMA mechanism. Then, there exists an NAM $\lambda^{NAM} : \mathcal{X} \to \mathbb{R}$ such that:*

(a) λ^{NAM} is concave on $\mathcal{X}$.

(b) If direct mechanism M^d is equivalent to M^{AMA}, then M^d is also equivalent to λ^{NAM}.

Theorem 3.2 theoretically guarantees that, compared with AMAs, NAMs never lose expressive power, even when λ^{NAM} is restricted to be concave. We note that Theorem 3.2 also holds without condition (a). However, condition (a) is imposed for the computational tractability of NAM, which we elaborate on in Section 4.3.

4 Learning Neural Affine Maximizers

This section introduces how we learn a Neural Affine Maximizer for a given auction model $\mathcal{A}$. At a high level, we first parameterize an NAM with a neural network that has universal approximation property with respect to convex functions. Thereafter, the network parameters are trained via first-order optimization.

4.1 Parameterization of Neural Affine Maximizers

We use Input Convex Neural Network (ICNN, (Amos et al., 2017)) to parameterize the boosting function in an NAM. The architecture of a K-layer ICNN is visualized in Figure 1.

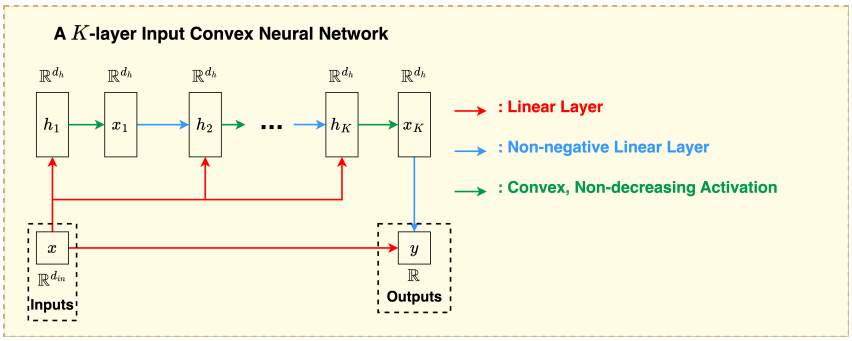


Figure 1: The architecture of an ICNN. Best viewed in color (same for the remaining figures).

Definition 4.1. A K-layer ICNN parameterizes a convex function $f : \mathbb{R}^{d_x} \to \mathbb{R}$ with parameters: $\{W_k^c\}_{1\le k\le K-1}$, $\{W_k^r\}_{0\le k\le K-1}$, $\{b_k\}_{0\le k\le K-1}$, $w^c \in \mathbb{R}_+^{d_h}$, $w^r \in \mathbb{R}^{d_x}$, $b \in \mathbb{R}$, $\{\sigma_k\}_{k\in[K]}$, where d_h is the hidden dimension, $W_k^c \in \mathbb{R}_+^{d_h\times d_h}$, $\sigma_k : \mathbb{R} \to \mathbb{R}$ is a convex, continuous and non-decreasing activation function (*e.g.*, leaky ReLU), and $W_k^r \in \mathbb{R}^{d_h\times d_x}$, $b_k \in \mathbb{R}^{d_h}$.

Given input $x \in \mathbb{R}^{d_x}$, the computation flow of ICNN to output $y = f(x) \in \mathbb{R}$ is illustrated in Algorithm 1.

Amos et al. (2017) has shown that the function represented by ICNN is always convex, but to our best knowledge, subsequent works fail to give a universal approximation guarantee on ICNN. We fill this blank by showing that ICNN is indeed a universal approximator of convex functions.

Algorithm 1 Computation of ICNN

Require: $W_k^c \geq 0, k = 1, 2, ..., K-1, w^c \geq 0$
input: $x \in \mathbb{R}^{d_x}$
$x^0 \leftarrow x$
$h^1 \leftarrow W_0^r \cdot x + b_0$
$x^1 \leftarrow \sigma_1(h^1)$, with activation function applied element-wise.
for $k = 1, ..., K-1$ **do**
  $h^{k+1} \leftarrow W_k^c \cdot x^k + W_k^r \cdot x + b_k$
  $x^{k+1} \leftarrow \sigma_{k+1}(h^{k+1})$, with activation function applied element-wise.
end for
$y = \langle w^c, x^K \rangle + \langle w^r, x \rangle + b$
return: y

Theorem 4.2. *Fix $\mathcal{X} \subseteq \mathbb{R}^{d_x}$ be a convex, compact subset of d_x-dimensional Euclidean space. Then, ICNN is a universal approximator of all functions $f(x) : \mathcal{X} \to \mathbb{R}$ that are convex and continuous on x.*

Together with Theorem 3.2, Theorem 4.2 justifies the expressive power of our approach: by using ICNN as the architecture of NAM, our approach can approximately represent the optimal (unknown) AMA with arbitrary precision.

As the boosting function in NAM is required to be concave, we apply the negative sign on the ICNN output. From now on, we denote θ as the network parameter, $\lambda_\theta^{\text{NAM}}(\cdot)$ as the parameterized NAM and $\text{ASW}(\boldsymbol{x}; \boldsymbol{t}, \theta) \coloneqq \sum_{i\in[n]} \langle x_i, t_i \rangle + \lambda_\theta^{\text{NAM}}(\boldsymbol{x})$ as the parameterized affine social welfare function.

4.2 Training of Neural Affine Maximizers

We now describe how the parameters of a Neural Affine Maximizer are learned via first-order optimization. Consider a θ-parameterized NAM $\lambda_\theta^{\text{NAM}}$. According to Section 3.1, every outcome computed by the mechanism—both $\boldsymbol{x}^d(\boldsymbol{t})$ and $\boldsymbol{x}^{-i}(\boldsymbol{t})$—must maximize the affine social welfare:

$$\boldsymbol{x}^{-i}(\boldsymbol{t}) \in \arg\max_{\boldsymbol{a}\in\mathcal{X}} \text{ASW}(\boldsymbol{a}; \boldsymbol{t}[-i], \theta), \quad \forall i \in [n], \qquad \boldsymbol{x}^d(\boldsymbol{t}) \in \arg\max_{\boldsymbol{a}\in\mathcal{X}} \text{ASW}(\boldsymbol{a}; \boldsymbol{t}, \theta) \tag{4}$$

To write these $n+1$ constraints compactly, we introduce the extended type profile $\tilde{\boldsymbol{t}} = (\boldsymbol{t}[-1], ..., \boldsymbol{t}[-n], \boldsymbol{t}) \in \mathcal{T}^{n+1}$—which is uniquely determined by $\boldsymbol{t}$ with $\tilde{\boldsymbol{t}}_{n+1} = \boldsymbol{t}$ and $\tilde{\boldsymbol{t}}_i = \boldsymbol{t}[-i]$ for $i \in [n]$—and the extended outcome profile $\tilde{\boldsymbol{x}}(\boldsymbol{t}) = (\boldsymbol{x}^{-1}(\boldsymbol{t}), ..., \boldsymbol{x}^{-n}(\boldsymbol{t}), \boldsymbol{x}^d(\boldsymbol{t})) \in \mathcal{X}^{n+1}$, where $\tilde{\boldsymbol{x}}_{n+1}(\boldsymbol{t}) = \boldsymbol{x}^d(\boldsymbol{t})$ and $\tilde{\boldsymbol{x}}_i(\boldsymbol{t}) = \boldsymbol{x}^{-i}(\boldsymbol{t})$. Under this notation, Equation (4) reduces to:

$$\tilde{\boldsymbol{x}}_i(\boldsymbol{t}) \in \arg\max_{\boldsymbol{a}\in\mathcal{X}} \text{ASW}(\boldsymbol{a}; \tilde{\boldsymbol{t}}_i, \theta), \qquad \forall i \in [n+1] \tag{5}$$

Since the payment rule $\boldsymbol{p}^d(\boldsymbol{t})$ in Equation (3) is fully determined by the extended outcomes $\tilde{\boldsymbol{x}}(\boldsymbol{t})$, the type profile $\boldsymbol{t}$, and the parameter θ, we can express the auctioneer's utility as $u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})$ without explicit reference to $\boldsymbol{p}$. The learning task is then captured by the following constrained optimization:

$$\max_{\tilde{\boldsymbol{x}}(\boldsymbol{t}), \theta} \quad \mathbb{E}_{\boldsymbol{t}\sim\mathcal{F}}[u_0(\tilde{\boldsymbol{x}}(\boldsymbol{t}), \theta; \boldsymbol{t})] \quad \text{s.t.} \quad \tilde{\boldsymbol{x}}_i(\boldsymbol{t}) \in \arg\max_{\boldsymbol{a}\in\mathcal{X}} \text{ASW}(\boldsymbol{a}; \tilde{\boldsymbol{t}}_i, \theta), \quad \forall i \in [n+1] \tag{6}$$

The remainder of this subsection develops a tractable procedure for solving Program (6).

Step 1: Sample Averaging. Because the expectation over $\mathcal{F}$ is generally intractable, we draw M i.i.d. type profiles $\mathcal{T}^M = \{\boldsymbol{t}^1, ..., \boldsymbol{t}^M\}$ from $\mathcal{F}$ and replace the expectation with a sample average, yielding:

$$\max_{\{\tilde{\boldsymbol{x}}^k \in \mathcal{X}^{n+1}\}_{k\in[M]}, \theta} \quad \frac{1}{M} \sum_{k\in[M]} u_0(\tilde{\boldsymbol{x}}^k, \theta; \boldsymbol{t}^k) \quad \text{s.t.} \quad \tilde{\boldsymbol{x}}_i^k \in \arg\max_{\boldsymbol{a}\in\mathcal{X}} \text{ASW}(\boldsymbol{a}; \tilde{\boldsymbol{t}}_i^k, \theta), \quad \forall i \in [n+1]. \tag{7}$$

Step 2: Softening the Constraints. The $\arg\max$ constraints in Program (7) are non-differentiable, precluding direct gradient-based optimization. We address this by replacing each hard $\arg\max$ with a Gibbs distribution that concentrates around the maximizer. Concretely, define

$$q^\beta(\boldsymbol{a}|\boldsymbol{t},\theta) = \frac{\exp(\beta \cdot \mathrm{ASW}(\boldsymbol{a};\boldsymbol{t},\theta))}{Z^\beta(\boldsymbol{t},\theta)}, \quad Z^\beta(\boldsymbol{t},\theta) = \int_{\boldsymbol{a}\in\mathcal{X}} \exp(\beta \cdot \mathrm{ASW}(\boldsymbol{a};\boldsymbol{t},\theta))\mathrm{d}\boldsymbol{a} \tag{8}$$

where $\beta > 0$ is a temperature hyperparameter controlling how sharply q^β is centered at the maximizer of ASW, and $Z^\beta(\boldsymbol{t},\theta)$ is the normalizing constant. Substituting this relaxation into Program (7) removes the argmax constraints and yields an unconstrained objective:

$$\max_\theta \qquad \mathrm{OBJ}^\beta(\theta) = \frac{1}{M}\sum_{k\in[M]} \mathbb{E}_{\tilde{\boldsymbol{x}}_i^k \overset{\text{i.d.}}{\sim} q^\beta(\cdot|\tilde{\boldsymbol{t}}_i^k,\theta)}[u_0(\tilde{\boldsymbol{x}}^k,\theta;\boldsymbol{t}^k)] \tag{9}$$

where $\overset{\text{i.d.}}{\sim}$ denotes that the samples are independently distributed.

Step 3: Covariance Trick. Following conventions in neural network optimization (Amari, 1993), we aim at computing an unbiased estimator of $\nabla_\theta \mathrm{OBJ}^\beta(\theta)$, namely the first-order gradient of the objective. Given this estimator, θ can be automatically optimized through first-order algorithms (*e.g.*, Adam) in libraries like Pytorch.

A difficulty here is that the differentiation operator ∇_θ and the expectation operator $\mathbb{E}[\cdot]$ cannot be interchanged directly, as $\mathbb{E}_{\tilde{\boldsymbol{x}}_i^k \overset{\text{i.d.}}{\sim} q^\beta(\cdot|\tilde{\boldsymbol{t}}_i^k,\theta)}$ explicitly depends on θ. This makes the gradient computation of Equation (9) non-trivial. Ma et al. (2025b) proposes "covariance trick" to resolve this difficulty. Inspired by Ma et al. (2025b), we derive the "covariance trick" for NAM in Proposition 4.3.

Proposition 4.3. *Following identity holds universally,*

$$\nabla_\theta \mathbb{E}_{\tilde{\boldsymbol{x}}_i}[u_0(\tilde{\boldsymbol{x}},\theta;\boldsymbol{t})] = \mathbb{E}_{\tilde{\boldsymbol{x}}_i}\left[\nabla u_0(\tilde{\boldsymbol{x}},\theta;\boldsymbol{t})\right] + \beta \cdot \mathrm{Cov}_{\tilde{\boldsymbol{x}}_i}\left[u_0(\tilde{\boldsymbol{x}},\theta;\boldsymbol{t}), \nabla_\theta \sum_{i\in[n+1]} \mathrm{ASW}(\tilde{\boldsymbol{x}}_i;\tilde{\boldsymbol{t}}_i,\theta)\right] \tag{10}$$

where $\tilde{\boldsymbol{x}}_i \overset{\text{i.d.}}{\sim} q^\beta(\cdot|\tilde{\boldsymbol{t}}_i,\theta)$.

Note that the left-hand side of Proposition 4.3 is the exact gradient of Equation (9) but is not directly computable. Meanwhile, the right-hand side can be unbiasedly estimated, given samples from $q^\beta(\cdot|\tilde{\boldsymbol{t}}_i,\theta)$ for each $i \in [n+1]$. Based on Proposition 4.3, we construct the loss function below:

$$\begin{aligned} L^\beta(\theta;\mathcal{T}^M) = \frac{1}{2M}\sum_{k\in[M]} &\left[\left(u_0(\tilde{\boldsymbol{y}}^k,\theta;\boldsymbol{t}^k) + u_0(\tilde{\boldsymbol{z}}^k,\theta;\boldsymbol{t}^k)\right)\right. \\ \left.+\beta \cdot sg\left(u_0(\tilde{\boldsymbol{y}}^k,\theta;\boldsymbol{t}^k) - u_0(\tilde{\boldsymbol{z}}^k,\theta;\boldsymbol{t}^k)\right) \cdot \right. &\left. \sum_{i\in[n+1]} \left(\mathrm{ASW}(\tilde{\boldsymbol{y}}_i^k;\boldsymbol{t}_i^k,\theta) - \mathrm{ASW}(\tilde{\boldsymbol{z}}_i^k;\boldsymbol{t}_i^k,\theta)\right),\right. \end{aligned} \tag{11}$$

Here $sg(\cdot)$ is the stop-gradient operator, defined by $\nabla_\theta(sg(f(\theta))\cdot g(\theta)) = f(\theta)\cdot\nabla_\theta g(\theta)$, eliminating the gradient propagation through $f(\theta)$. This operator is implementable in libraries like PyTorch.

If we can generate $\tilde{\boldsymbol{y}}_i^k, \tilde{\boldsymbol{z}}_i^k \overset{\text{i.d.}}{\sim} q^\beta(\cdot|\tilde{\boldsymbol{t}}_i^k,\theta)$, then $\nabla_\theta L^\beta(\theta;\mathcal{T}^M)$ is an unbiased estimator of $\nabla_\theta \mathrm{OBJ}^\beta(\theta)$. The next step explains how to generate $\tilde{\boldsymbol{y}}_i^k, \tilde{\boldsymbol{z}}_i^k \overset{\text{i.d.}}{\sim} q^\beta(\cdot|\tilde{\boldsymbol{t}}_i^k,\theta)$ during the full training process.

Step 4: Continuous Sampling within Simplex Constraints The last step focuses on how to sample $\tilde{\boldsymbol{y}}_i^k \sim q^\beta(\cdot|\tilde{\boldsymbol{t}}_i^k,\theta)$ efficiently for each $k \in [M], i \in [n+1]$. The treatment of $\tilde{\boldsymbol{z}}_i^k$ is similar.

Inspired by Ma et al. (2025b), we construct a discrete-time Langevin dynamics (Roberts & Tweedie, 1996) to approximately generate such samples. Extending their techniques to our setting is non-trivial, because the outcome space in our setting has simplex constraints ($\boldsymbol{x} \geq 0$ and $\sum_i x_{ij} \leq 1, \forall j$).

Denote θ^{last} and θ^{new} as the last-iteration and current network parameters. Assume that we already have an approximate sample $\tilde{\boldsymbol{y}}_i^{last}$ from $q^\beta(\cdot;\tilde{\boldsymbol{t}}_i,\theta^{last})$ that stays within the simplex constraints. Fix

the iteration time T, we sequentially construct $\tilde{\boldsymbol{y}}_i^1, ..., \tilde{\boldsymbol{y}}_i^T$ with initial point $\tilde{\boldsymbol{y}}_i^0 = \tilde{\boldsymbol{y}}_i^{last}$ as follows:

$$\tilde{\boldsymbol{y}}_i^t = \tilde{\boldsymbol{y}}_i^{t-1} + \eta_i \nabla_{x_i} u_i(\tilde{\boldsymbol{y}}_i^{t-1}; \tilde{\boldsymbol{t}}_i, \theta^{new}) + \sqrt{2\eta_i {\beta_i}^{-1}}\epsilon_i^t,\ \epsilon_i^t \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I), \tag{12}$$

where $\eta_i > 0$ is the step size.

Note that $\tilde{\boldsymbol{y}}_i^t$ does not necessarily stay within the simplex constraints. Therefore, we apply Skorokhod reflection (Skorokhod, 1961) on the boundary to make sure the constraints hold at all times. Specifically, let $x = \tilde{\boldsymbol{y}}_i^{t-1}$ be the initial point, $d = \tilde{\boldsymbol{y}}_i^t - \tilde{\boldsymbol{y}}_i^{t-1}$ be the update direction, and assume that x stays within the simplex. We accept $x + d$ iff $x + d$ stays within the simplex. Let $\{\langle a_j, \cdot\rangle \geq b_j\}_{j\in[n+1]}$ be the $n+1$ half-spaces of the simplex. Let $x(s) = x + sd$, for $s \in [0, 1]$, be the trajectory, and let t_0 be the minimum time at which $x(s)$ hits the boundary, *i.e.*,

$$t_0 = \min_j \frac{b_j - \langle a_j, x\rangle}{\langle a_j, d\rangle} \tag{13}$$

We then let $x \leftarrow x + t_0 d$, $d \leftarrow (1 - t_0)d$, $d \leftarrow d - 2\langle n_j, d\rangle n_j$, where n_j is the normal vector of j'th half-space and j is the index to achieve the min value in Equation (13). These updates make a "reflection" of the update vector $x \to x + d$ *w.r.t.* the jth half-space. We repeatedly update (x, d) until $x + d$ is accepted. Note that the reflection in a convex polyhedron terminates in finitely many steps (Burago et al., 1998); we empirically find that all reflections terminate within $n(n + 1)$ iterations.

4.3 Inference of Neural Affine Maximizers

Given a trained NAM $\lambda_\theta^{\text{NAM}}$, we can evaluate it as if it was a direct mechanism, *i.e.*, by computing the outcomes $(\boldsymbol{x}, \boldsymbol{p})$ for a given type profile $\boldsymbol{t}$ through the steps described in Equation (3). Most steps can be computed naturally. The only step that requires additional attention is computing the $\arg\max$ points of $\text{ASW}(\boldsymbol{x}; \boldsymbol{t}, \theta)$. In NAM, $\text{ASW}(\boldsymbol{x}; \boldsymbol{t}, \theta)$ is guaranteed to be concave due to the architectural convexity of ICNN. Therefore, with convex optimization tools (Boyd & Vandenberghe, 2004), these $\arg\max$ points can be efficiently computed in polynomial time.

5 Experiments

To evaluate the effectiveness of NAM, we conduct experiments in this section. In all these experiments, we have $\mathcal{T}_i = [0, 1]^m$. Therefore, an auction $\mathcal{A}$ is represented by $(n, m, \mathcal{F})$. All experiments are executed on a single NVIDIA A100 GPU.

5.1 Baselines and Settings

We consider the following state-of-the-art approaches that guarantee the truthfulness of learned auctions as baselines: LotteryAMA (Curry et al., 2023), AMenuNet (Duan et al., 2023b), and GemNet (Wang et al., 2024b). We implement LotteryAMA in our experiments and report the maximum of the value obtained in our experiments and the value reported in the original paper (when applicable). For AMenuNet, we incorporate their open-source implementation into our experiments. For GemNet, we report the value from its original paper when the same setting appears, as GemNet is not open-sourced. The computational complexity of NAM, LotteryAMA, and AMenuNet is set to $O(n^3m^2)$ to make the comparison fair [2].

We also implement three classical baselines, respectively:

- VCG (Vickrey, 1961; Clarke, 1971; Groves, 1973): Arguably the most classical mechanism which has strong versatility.
- Itemwise: Run second-price auction on each good independently with a parameterized reserve price. The optimal reserve price is solved through grid search. Note that when buyers' type distributions are i.i.d. (which is satisfied with uniform distribution), this baseline is identical with Itemwise-Myerson, a widely-used baseline in auctions (Curry et al., 2023; Dütting et al., 2024; Wang et al., 2024b).

[2] See more details in Appendix D

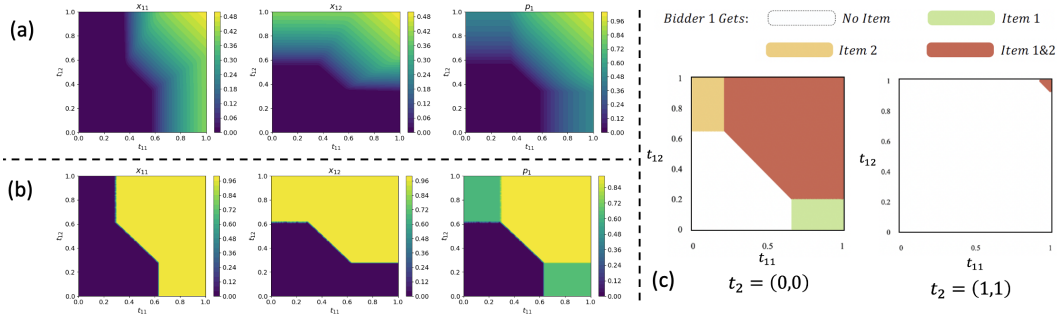


Figure 2: NAM and GemNet's learned allocation for first buyer in $U_{2,2}$, when $t_2 = (1, 1)$ or $t_2 = (0, 0)$. (a): NAM's result for $t_2 = (1, 1)$; (b): NAM's result for $t_2 = (0, 0)$; (c) GemNet's result for both t_2s (reproduced from Wang et al. (2024b)).

- Bundle: Bundle all goods as one good and run second-price auction on this bundling with a parameterized reserve price. The optimal reserve price is solved through grid search. This baseline is also widely used in auctions (Curry et al., 2023; Ma et al., 2025b).

We denote $D_{n,m}$ as the setting with n buyers, m goods and type profile distribution D. We incorporate the following choices of D:

- $D = U$: $t_{ij} \overset{\text{i.i.d.}}{\sim} U([0, 1])$. This is arguably the most standard setting in auctions.
- $D = S$: Generate t_i uniformly from the set X, where $X = \{e_i\}_{i\in[m]} \cup \{\frac{\mathbf{1}}{\sqrt{m}}\}$, e_i is the all-0 m-dimensional vector except the i' coordinate equals to 1, and $\mathbf{1}$ is the all-1 vector. This distribution is inspired by the "spherical distribution" in Curry et al. (2023).
- $D = B$: $t_{ij} \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(p = 0.2)$ [3]. This distribution is inspired by Ma et al. (2025b).
- $D = R$: Generate $b_i, g_j \overset{\text{i.i.d.}}{\sim} U([-1, 1]^{10})$ and let $t_{ij} = \sigma(\langle b_i, g_j\rangle)$, where $\sigma(x) = (1 + \exp(-x))^{-1}$ is the sigmoid function. This distribution is inspired by Duan et al. (2023b), where (b_i, g_j) are explained as buyer and good representations.

5.2 Experiment 1: The Case of $U_{2,2}$

In the first experiment, we consider arguably the simplest nontrivial case $U_{2,2}$. We visualize the learned allocation rule and payment rule *w.r.t.* the first buyer when the second buyer's type is $t_2 \in \{(0, 0), (1, 1)\}$ and compare the result with GemNet in Figure 2. Specifically, the auction learned by GemNet in $U_{2,2}$ is deterministic. The auction found by NAM has a similar structure to GemNet when $t_2 = (0, 0)$. Surprisingly, when $t_2 = (1, 1)$, NAM allocates goods to buyer 1 with low but non-zero probability. As a contrast, GemNet does not allocate goods to buyer 1 at all. Notably, NAM finds a truthful auction with revenue 0.8995, achieving 2.4% improvement over GemNet (0.878, which, to our knowledge, reaches the largest revenue among all existing truthful auctions prior to our work). This comparison suggests that randomization can indeed improve revenue over deterministic auctions even in the arguably simplest nontrivial setting $U_{2,2}$. We conjecture that the optimal auction for $U_{2,2}$ setting is randomized, though this is still an open problem. We also visualize the learned auction over a grid of values of t_2 in Appendix C.

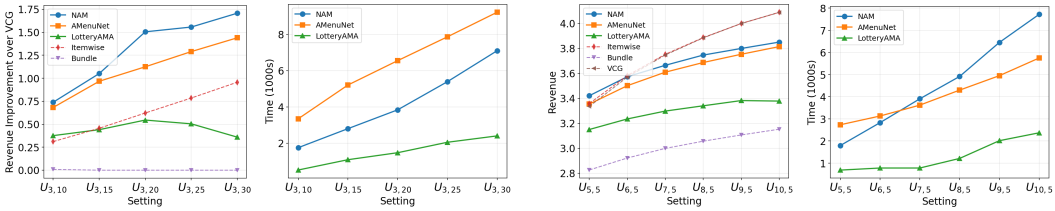
5.3 Experiment 2: Performance on Uniform Distribution

Next, we consider general $U_{n,m}$ settings up to 5 buyers or 10 goods, which have been widely studied (Wang et al., 2024b; Duan et al., 2023b; Curry et al., 2023). We note that most prior works consider settings with at most 5 buyers or 10 goods. Table 1 shows the auctioneer's expected revenue learned by different approaches on these settings. Notably, NAM gives the optimal revenue on all

[3] A Bernoulli random variable with parameter p has prob. p to be 1 and prob. $1 - p$ to be 0.

Table 1: Auctioneer's expected revenue in experiment with n buyers, m goods and i.i.d., $U([0,1])$ valuations. Four valid digits are reported. The ROR for LotteryAMA and AMenuNet $< 1\%$ for all listed settings. The maximum value is underlined.

Method & Settings	$U_{2,2}$	$U_{2,5}$	$U_{3,3}$	$U_{3,5}$	$U_{3,10}$	$U_{5,5}$	$U_{5,10}$
NAM	<u>0.8995</u>	<u>2.387</u>	<u>1.695</u>	2.833	<u>5.739</u>	<u>3.421</u>	<u>6.841</u>
ROR of NAM	0.16	0.27	0.35	0.43	0.47	0.51	0.46
LotteryAMA	0.8773	2.320	1.630	2.712	5.415	3.150	5.689
AMenuNet	0.8618	2.277	1.632	2.801	5.682	3.392	6.521
GemNet	0.8780	2.310	1.675	<u>3.124</u>	-	-	-
Itemwise	0.833	2.083	1.593	2.663	5.311	3.358	6.713
Bundle	0.839	2.186	1.508	2.508	5.009	2.826	5.454
VCG	0.667	1.667	1.500	2.500	5.000	3.335	6.669



(a) (b) (c) (d)

Figure 3: (a)(b): Revenue and Training Time comparison on $U_{3,m}$ settings with $m \in \{10, 15, 20, 25, 30\}$. The revenues are subtracted by VCG revenues for better visualization. (c)(d): Revenue and Training Time comparison on $U_{n,5}$ settings with $n \in \{5, 6, 7, 8, 9, 10\}$.

settings except one. This comparison demonstrates the effectiveness of our proposed Neural Affine Maximizer approach.

To understand why NAM might be effective in these settings, we design an indicator, named "randomized outcome ratio" (ROR), to measure the "randomness" of learned auctions. The ROR is defined as:

$$\text{ROR} := \frac{n}{(n-1)m} \sum_{j\in[m]} \mathbb{E}_{\boldsymbol{t}\sim\mathcal{F}}[1 - \max\{\max_i x_{i,j}, 1 - \sum_{i\in[n]} x_{i,j}\}] \tag{14}$$

Note that ROR lies in $[0, 1]$ and reaches 0 iff the auction always gives deterministic allocation and reaches 1 iff the auction always gives "the most nondeterministic" allocation that allocates each good to each buyer with probability $1/(n+1)$. The ROR results are also reported in Table 1. Note that LotteryAMA and AMenuNet always find approximately deterministic AMAs (ROR< 0.01).

Given the visualization results in Section 5.2, we hypothesize that the optimal auctions in $U_{n,m}$ settings use infinitely-many outcomes. Due to the discretization nature, both LotteryAMA and AMenuNet are insufficiently expressive to represent the AMAs with infinitely-many outcomes. As a contrast, NAM directly learns the boosting function over full outcome space, including both deterministic and randomized outcomes. This distinction explains why NAM is able to find randomized auctions, while LotteryAMA and AMenuNet fail.

5.4 Experiment 3: Scaling Performance

We conduct experiments with up to 10 buyers or 30 goods to evaluate the scaling performance of NAM. Figure 3 visualizes revenue and training time for NAM, LotteryAMA, and AMenuNet. Notably, as the number of goods increases, all approaches learn auctions that outperform VCG in revenue; in contrast, as the number of buyers increases, they struggle to learn auctions that compete with Itemwise Myerson or VCG. We conjecture that an intrinsic barrier may hinder AMAs from learning near-optimal auctions in the latter regime, and leave a formal understanding to future studies.

Table 2: Auctioneer's expected revenue in experiment with n buyers, m goods and diverse distributions. The ROR for LotteryAMA and AMenuNet $< 1\%$ for all listed settings, excepts ROR$= 0.02$ for AMenuNet in $S_{3,5}$. * represents the known optimal value. The maximum value is underlined.

Method & Settings	$S_{2,2}$	$S_{3,5}$	$B_{2,2}$	$B_{3,5}$	$R_{2,2}$	$R_{3,5}$
NAM	<u>1.416</u>	<u>2.405</u>	0.6971	2.331	<u>0.9122</u>	2.672
ROR of NAM	0.32	0.17	0.04	0.07	0.33	0.53
LotteryAMA	0.9652	1.614	0.5000	0.995	0.8458	2.624
AMenuNet	1.265	2.385	0.6692	1.692	0.8794	<u>2.674</u>
Itemwise	1.320	2.111	<u>0.7166</u>*	<u>2.447</u>*	0.8366	2.538
Bundle	1.047	1.087	0.5900	1.199	0.8694	2.498
VCG	0.6957	0.7796	0.0813	0.5116	0.7699	2.498

5.5 Experiment 4: Performance on Other Distributions

We also consider experiments on diverse distributions to verify the robustness of NAM. In all experiments, the performance of NAM either surpasses or is competitive with LotteryAMA and AMenuNet. Interestingly, in $B_{n,m}$ settings where optimal auctions are provably deterministic (Ma et al., 2025b), NAM finds near-deterministic, near-optimal auctions (with low ROR), while LotteryAMA and AMenuNet find deterministic but suboptimal auctions. This suggests that in addition to limited expressiveness, discretization-based approaches like LotteryAMA and AMenuNet are also prone to get stuck in local optima.

6 Conclusions and Discussions

This paper proposes NAM, a discretization-free approach to learn truthful auctions. Compared with existing approaches, NAM finds truthful auctions with larger revenue in many scenarios. For future work, one direction is to design permutation-equivariant and convex neural networks to replace ICNN and potentially accelerate training (Qin et al., 2022; Ma et al., 2025b; Han et al., 2022). Another is to apply similar techniques to discover solutions in broader applications of differentiable economics.

NAM possesses some features that differ from existing approaches. First, training time scales linearly with the number of samples M, which induces a trade-off: a small M is prone to overfitting, while a large M makes training costly. We find empirically that $M = 2^{14}$ strikes a good balance. Second, NAM incurs substantially larger inference time than baselines, as it requires solving convex programs to compute outcomes and monetary transfers. In our largest settings ($U_{3,30}$ and $U_{10,5}$), NAM takes at most 40s to process a batch of 2^{14} samples. We consider this inference cost acceptable in practice.

REFERENCES

Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5 (4-5):185–196, 1993.

Brandon Amos, Lei Xu, and J Zico Kolter. Input convex neural networks. In *International conference on machine learning*, pp. 146–155. PMLR, 2017.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Jerome Bracken and James T McGill. Mathematical programs with optimization problems in the constraints. *Operations Research*, 21(1):37–44, 1973.

Dmitry Burago, Sergei Ferleger, and Alexey Kononenko. Uniform estimates on the number of collisions in semi-dispersing billiards. *Annals of Mathematics*, 147(3):695–708, 1998.

Giuseppe C Calafiore, Stephane Gaubert, and Corrado Possieri. Log-sum-exp neural networks and posynomial models for convex and log-log-convex data. *IEEE transactions on neural networks and learning systems*, 31(3):827–838, 2019.

Lesi Chen, Yaohua Ma, and Jingzhao Zhang. Near-optimal nonconvex-strongly-convex bilevel optimization with fully first-order oracles. *Journal of Machine Learning Research*, 26:1–56, 2025.

Edward H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, 1971. doi: 10.1007/BF01726210.

Matthew Curry, Tuomas Sandholm, and David C. Parkes. Differentiable economics for randomized affine maximizer auctions. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI-23)*, 2023. doi: 10.24963/ijcai.2023/293. URL `https://www.ijcai.org/proceedings/2023/0293.pdf`.

Michael Curry, Ping-Yeh Chiang, Tom Goldstein, and John Dickerson. Certifying strategyproof auction networks. *Advances in Neural Information Processing Systems*, 33:4987–4998, 2020.

Michael J Curry, Uro Lyi, Tom Goldstein, and John P Dickerson. Learning revenue-maximizing auctions with differentiable matching. In *International Conference on Artificial Intelligence and Statistics*, pp. 6062–6073. PMLR, 2022.

Constantinos Daskalakis, Alan Deckelbaum, and Christos Tzamos. Strong duality for a multiple-good monopolist. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pp. 449–450, 2015.

Stephan Dempe. *Foundations of bilevel programming*. Springer Science & Business Media, 2002.

Justin Domke. Generic methods for optimization-based modeling. In *Artificial Intelligence and Statistics*, pp. 318–326. PMLR, 2012.

Zhijian Duan, Jingwu Tang, Yutong Yin, Zhe Feng, Xiang Yan, Manzil Zaheer, and Xiaotie Deng. A context-integrated transformer-based neural network for auction design. In *International Conference on Machine Learning*, pp. 5609–5626. PMLR, 2022.

Zhijian Duan, Wenhan Huang, Dinghuai Zhang, Yali Du, Jun Wang, Yaodong Yang, and Xiaotie Deng. Is nash equilibrium approximator learnable? In *22nd International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023*, pp. 233–241, 2023a.

Zhijian Duan, Haoran Sun, Yurong Chen, and Xiaotie Deng. A scalable neural network for DSIC affine maximizer auction design. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, 2023b. URL `https://dblp.org/rec/conf/nips/0001S0D23`.

Zhijian Duan, Haoran Sun, Yichong Xia, Siqiang Wang, Zhilin Zhang, Chuan Yu, Jian Xu, Bo Zheng, and Xiaotie Deng. Automated deterministic auction design with objective decomposition. *arXiv preprint arXiv:2402.11904*, 2024.

Paul Dütting, Zhe Feng, Harikrishna Narasimhan, David C Parkes, and Sai Srivatsa Ravindranath. Optimal auctions through deep learning: Advances in differentiable economics. *Journal of the ACM*, 71(1):1–53, 2024.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pp. 1126–1135. PMLR, 2017.

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, pp. 1568–1577. PMLR, 2018.

Saeed Ghadimi and Mengdi Wang. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*, 2018.

Yiannis Giannakopoulos and Elias Koutsoupias. Duality and optimality of auctions for uniform distributions. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pp. 259–276, 2014.

Noah Golowich, Harikrishna Narasimhan, and David C. Parkes. Deep learning for multi-facility location mechanism design. IJCAI'18. AAAI Press, 2018. ISBN 9780999241127.

Theodore Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973. doi: 10.2307/1914085.

Jiequn Han, Yingzhou Li, Lin Lin, Jianfeng Lu, Jiefu Zhang, and Linfeng Zhang. Universal approximation of symmetric and anti-symmetric functions. *Communications in Mathematical Sciences*, 20(5):1397–1408, 2022.

Minhui Huang, Xuxing Chen, Kaiyi Ji, Shiqian Ma, and Lifeng Lai. Efficiently escaping saddle points in bilevel optimization. *Journal of Machine Learning Research*, 26:1–61, 2025.

Dmitry Ivanov, Iskander Safiulin, Igor Filippov, and Ksenia Balabaeva. Optimal-er auctions through attention. *Advances in Neural Information Processing Systems*, 35:34734–34747, 2022.

Anton Likhodedov and Tuomas Sandholm. Methods for boosting revenue in combinatorial auctions. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, 2004. URL `https://f.aaai.org/Library/AAAI/2004/aaai04-037.php`.

Yunxuan Ma, Yide Bian, Hao Xu, Weitao Yang, Jingshu Zhao, Zhijian Duan, Feng Wang, and Xiaotie Deng. Large-scale contextual market equilibrium computation through deep learning. In *International Workshop on Frontiers in Algorithmics*, pp. 356–371. Springer, 2025a.

Yunxuan Ma, Siqiang Wang, Zhijian Duan, Yukun Cheng, and Xiaotie Deng. Learning truthful mechanisms without discretization. *arXiv preprint arXiv:2506.22911*, 2025b.

Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *International Conference on Machine Learning*, pp. 2113–2122. PMLR, 2015.

Alejandro M Manelli and Daniel R Vincent. Bundling as an optimal selling mechanism for a multiple-good monopolist. *Journal of Economic Theory*, 127(1):1–35, 2006.

Peter McMullen. The maximum numbers of faces of a convex polytope. *Mathematika*, 17(2): 179–184, 1970.

Roger B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981. doi: 10.1287/moor.6.1.58.

Harikrishna Narasimhan, Shivani Brinda Agarwal, and David C Parkes. Automated mechanism design without money via machine learning. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, 2016.

Gregory Pavlov. Optimal mechanism for selling two goods. *The BE Journal of Theoretical Economics*, 11(1):000010220219351704 1664, 2011.

Neehar Peri, Michael Curry, Samuel Dooley, and John Dickerson. Preferencenet: Encoding human preferences in auction design with deep learning. *Advances in Neural Information Processing Systems*, 34:17532–17542, 2021.

Tian Qin, Fengxiang He, Dingfeng Shi, Wenbing Huang, and Dacheng Tao. Benefits of permutation-equivariance in auction mechanisms. *Advances in Neural Information Processing Systems*, 35: 18131–18142, 2022.

Jad Rahme, Samy Jelassi, Joan Bruna, and S Matthew Weinberg. A permutation-equivariant neural network architecture for auction design. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 5664–5672, 2021a.

Jad Rahme, Samy Jelassi, and S. Matthew Weinberg. Auction learning as a two-player game. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=YHdeAO61l6T.

Sai Srivatsa Ravindranath, Zhe Feng, Shira Li, Jonathan Ma, Scott D Kominers, and David C Parkes. Deep learning for two-sided matching. *arXiv preprint arXiv:2107.03427*, 2021.

Gareth O Roberts and Richard L Tweedie. Exponential convergence of langevin distributions and their discrete approximations. 1996.

John Roberts. The characterization of implementable choice rules. In Jean-Jacques Laffont (ed.), *Aggregation and Revelation of Preferences*, pp. 321–349. North-Holland, 1979.

Tuomas Sandholm and Anton Likhodedov. Automated design of revenue-maximizing combinatorial auctions. *Operations Research*, 63(5):1000–1025, 2015. doi: 10.1287/opre.2015.1398.

Weiran Shen, Pingzhong Tang, and Song Zuo. Automated mechanism design via neural networks. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19*, pp. 215–223. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

Anatoliy V Skorokhod. Stochastic equations for diffusion processes in a bounded region. *Theory of Probability & Its Applications*, 6(3):264–274, 1961.

William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961. doi: 10.1111/j.1540-6261.1961.tb02789.x.

Tonghan Wang, Paul Duetting, Dmitry Ivanov, Inbal Talgam-Cohen, and David C Parkes. Deep contract design via discontinuous networks. *Advances in Neural Information Processing Systems*, 36, 2024a.

Tonghan Wang, Yanchen Jiang, and David C. Parkes. Gemnet: Menu-based, strategy-proof multi-bidder auctions through deep learning. *CoRR*, abs/2406.07428, 2024b. URL https://arxiv.org/abs/2406.07428.

Xavier Warin. The groupmax neural network approximation of convex functions. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

Andrew Chi-Chih Yao. Dominant-strategy versus bayesian multi-item auctions: Maximum revenue determination and comparison. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pp. 3–20, 2017.

Peiran Yu, Junyi Li, and Heng Huang. Hessian free efficient single loop iterative differentiation methods for bi-level optimization problems. *Transactions on Machine Learning Research*, 2025.

Stephan Zheng, Alexander Trott, Sunil Srinivasa, David C Parkes, and Richard Socher. The ai economist: Taxation policy design via two-level deep multiagent reinforcement learning. *Science advances*, 8(18):eabk2607, 2022.

A Further Related Works

Differentiable Economics. Differentiable Economics is a technical paradigm that aims to solve economic problems with differentiable functions (*e.g.*, neural networks) and gradient-based optimization. Within this field, much progress has been made in the context of social choice (Narasimhan et al., 2016), facility location (Golowich et al., 2018), matching market (Ravindranath et al., 2021), contract design (Wang et al., 2024a), market equilibrium (Ma et al., 2025a), and economic tax design (Zheng et al., 2022).

Automated Mechanism Design (AMD) is arguably one of the most widely studied topics in this area. The integration of differentiable economics into AMD was first studied by Dütting et al. (2024), followed by Curry et al. (2020); Peri et al. (2021); Rahme et al. (2021b;a); Curry et al. (2022); Duan et al. (2022); Ivanov et al. (2022). However, these approaches are not guaranteed to output truthful auctions. On the truthful-learning side, Shen et al. (2019) initiates the learning of truthful single-buyer auctions, while Curry et al. (2023); Duan et al. (2023b; 2024); Wang et al. (2024b); Ma et al. (2025b) extend it to truthful multi-buyer auctions. This work also lies in the field of differentiable economics.

Bi-level Optimization. AMD can be viewed as a bi-level optimization problem (Dempe, 2002) from a deeper perspective, where the outer level (auctioneer) optimizes revenue subject to the rationality constraints of the inner level (*e.g.*, regret minimization as in Dütting et al. (2024) or affine welfare maximization as in Curry et al. (2023)). Bi-level optimization dates back to Bracken & McGill (1973) and has recently attracted attention in machine learning, with applications to hyper-parameter optimization (Franceschi et al., 2018; Maclaurin et al., 2015) and meta-learning (Finn et al., 2017).

While bi-level algorithms generally require the strict convexity of the inner program and second-order derivatives to tackle the bi-level structure (see, for example, Domke (2012); Maclaurin et al. (2015); Ghadimi & Wang (2018); Huang et al. (2025); Chen et al. (2025); Yu et al. (2025)), this work only requires the (weak) convexity of the inner program (Theorem 4.2) and uses the *covariance trick* (Proposition 4.3) to tackle the bi-level structure, eliminating the need to compute second-order information. We believe this work provides a new insight for bi-level optimization.

Convex Neural Networks. Max-of-Affine (MoA) functions and Log-sum-exp (LSE) functions are two classical convex function classes, with universal approximation established by Calafiore et al. (2019). However, these function designs have incomplete network architectures and suffer from the curse of dimensionality, with a counterexample provided in McMullen (1970).

Amos et al. (2017) proposes Input Convex Neural Network (ICNN), a network-based convex function approximator, yet subsequent works are unable to determine whether it is a universal approximator to our knowledge. Later, Warin (2023) propose GroupMax Network and show its universal approximation property. Surprisingly, we find that ICNN generally outperforms GroupMax Network in our task. This work connects to this line of literature by utilizing ICNN as architectural design, simultaneously proving the universal approximation property of ICNN.

B Omitted Proofs

B.1 Proof of Proposition 2.4

Proposition 2.4. (Truthfulness of AMA). *Let M^{AMA} be an AMA mechanism and M^d be a direct mechanism equivalent to M^{AMA}, then M^d is truthful.*

Proof. Recall that $M^{\text{AMA}} = \{\boldsymbol{x}^k, \lambda^k\}_{k\in[M]}$ and $M^d = (\boldsymbol{x}^d, \boldsymbol{p}^d)$.

Consider the utility of player i by reporting t_i', when her true valuation is t_i and other players report $\boldsymbol{t}_{-i}$:

$$\begin{aligned} &u_i(x_i^d(t_i';\boldsymbol{t}_{-i}), p_i^d(t_i';\boldsymbol{t}_{-i}); t_i) = \langle x_i^d(t_i';\boldsymbol{t}_{-i}), t_i\rangle - p_i^d(t_i';\boldsymbol{t}_{-i}) \\ =&\langle x_i^{k^*(t_i',\boldsymbol{t}_{-i})}, t_i\rangle + \text{ASW}_{-i}(k^*(t_i',\boldsymbol{t}_{-i}); (t_i',\boldsymbol{t}_{-i})) - \text{ASW}_{-i}(k^{-i,*}(t_i',\boldsymbol{t}_{-i}); (t_i',\boldsymbol{t}_{-i})) \end{aligned} \tag{15}$$

If player i reports truthfully, then her utility is

$$\begin{aligned} & u_i(x_i^d(\boldsymbol{t}), p_i^d(\boldsymbol{t}); t_i) \\ =& \mathrm{ASW}(k^*(\boldsymbol{t}); \boldsymbol{t}) - \mathrm{ASW}_{-i}(k^{-i,*}(\boldsymbol{t}); (\boldsymbol{t})) \\ =& \max_k \mathrm{ASW}(k; \boldsymbol{t}) - \max_k \mathrm{ASW}_{-i}(k; \boldsymbol{t}) \end{aligned} \tag{16}$$

Since $\langle x_i^k, t_i \rangle \geq 0$ always hold, we have $\mathrm{ASW}(k; \boldsymbol{t}) \geq \mathrm{ASW}_{-i}(k; \boldsymbol{t})$ holds for all k, therefore

$$u_i(x_i^d(\boldsymbol{t}), p_i^d(\boldsymbol{t}); t_i) = \max_k \mathrm{ASW}(k; \boldsymbol{t}) - \max_k \mathrm{ASW}_{-i}(k; \boldsymbol{t}) \geq 0 \tag{17}$$

This means that M^d is IR.

Notice that $\mathrm{ASW}_{-i}(t_i', \boldsymbol{t}_{-i})$ do not rely on t_i' and t_i by definition, therefore,

$$\begin{aligned} & u_i(x_i^d(t_i'; \boldsymbol{t}_{-i}), p_i^d(t_i'; \boldsymbol{t}_{-i}); t_i) = \langle x_i^{k^*(t_i', \boldsymbol{t}_{-i})}, t_i \rangle + \mathrm{ASW}_{-i}(k^*(t_i', \boldsymbol{t}_{-i}); \boldsymbol{t}) - f(\boldsymbol{t}_{-i}) \\ =& \mathrm{ASW}(k^*(t_i', \boldsymbol{t}_{-i}); \boldsymbol{t}) - f(\boldsymbol{t}_{-i}) \end{aligned} \tag{18}$$

By definition of maximization,

$$\mathrm{ASW}(k^*(t_i', \boldsymbol{t}_{-i}); \boldsymbol{t}) \leq \max_k \mathrm{ASW}(k; \boldsymbol{t}) = \mathrm{ASW}(k^*(\boldsymbol{t}); \boldsymbol{t}) \tag{19}$$

Note that after removing the term $f(\boldsymbol{t}_{-i})$ that is constant for player i, RHS of Equation (19) is the utility of truthful reporting for player i, while LHS of Equation (19) is the utility of reporting t_i' for player i. Therefore, by reporting $t_i' = t_i$, player i achieves her maximum utility. Therefore, M^d is truthful.

□

B.2 Proof of Proposition 3.1

Proposition 3.1. (Truthfulness of NAM) *Let λ^{NAM} be a Neural Affine Maximizer and M^d be a direct mechanism that is equivalent to λ^{NAM}. Then M^d is truthful.*

Proof. Recall that an NAM is a continuous function $\lambda^{\mathrm{NAM}} : \mathcal{X} \to \mathbb{R}$ and $M^d = (\boldsymbol{x}^d, \boldsymbol{p}^d)$.

Consider the utility of player i by reporting t_i', when her true valuation is t_i and other players report $\boldsymbol{t}_{-i}$:

$$\begin{aligned} & u_i(x_i^d(t_i'; \boldsymbol{t}_{-i}), p_i^d(t_i'; \boldsymbol{t}_{-i}); t_i) = \langle x_i^d(t_i'; \boldsymbol{t}_{-i}), t_i \rangle - p_i^d(t_i'; \boldsymbol{t}_{-i}) \\ =& \langle x_i^d(t_i'; \boldsymbol{t}_{-i}), t_i \rangle - \mathrm{ASW}(\boldsymbol{x}^{-i}(t_i', \boldsymbol{t}_{-i}); \boldsymbol{t}[-i]) + \mathrm{ASW}(\boldsymbol{x}^d(t_i', \boldsymbol{t}_{-i}); \boldsymbol{t}[-i]) \end{aligned} \tag{20}$$

By reporting truthfully and note that $\mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t}[-i]) + \langle x_i, t_i \rangle = \mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t})$, player i's utility becomes,

$$\mathrm{ASW}(\boldsymbol{x}^d(\boldsymbol{t}); \boldsymbol{t}) - \mathrm{ASW}(\boldsymbol{x}^{-i}(\boldsymbol{t}); \boldsymbol{t}[-i]) = \max_{\boldsymbol{x}} \mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t}) - \max_{\boldsymbol{x}} \mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t}[-i]) \tag{21}$$

As $\mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t}) \geq \mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t}[-i])$ holds universally, we have $\max_{\boldsymbol{x}} \mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t}) \geq \max_{\boldsymbol{x}} \mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t}[-i])$ holds universally as well. Therefore, NAM is IR.

Next, by observing that $\mathrm{ASW}(\boldsymbol{x}^{-i}(t_i', \boldsymbol{t}_{-i}); \boldsymbol{t}[-i]) = \mathrm{ASW}(\boldsymbol{x}^{-i}(t_i', \boldsymbol{t}_{-i}); (t_i', \boldsymbol{t}_{-i})[-i]) = \max_{\boldsymbol{x}} \mathrm{ASW}(\boldsymbol{x}, (t_i', \boldsymbol{t}_{-i})[-i]) = \max_{\boldsymbol{x}} \mathrm{ASW}(\boldsymbol{x}, \boldsymbol{t}[-i])$, we know this term do not rely on t_i' and t_i. We then denote this term as $f(\boldsymbol{t}_{-i})$.

Then, player i's utility of reporting t_i' is:

$$\begin{aligned} & \langle x_i^d(t_i'; \boldsymbol{t}_{-i}), t_i \rangle + \mathrm{ASW}(\boldsymbol{x}^d(t_i', \boldsymbol{t}_{-i}); \boldsymbol{t}[-i]) - f(\boldsymbol{t}_{-i}) \\ =& \mathrm{ASW}(\boldsymbol{x}^d(t_i'; \boldsymbol{t}_{-i}); \boldsymbol{t}) - f(\boldsymbol{t}_{-i}) \leq \max_{\boldsymbol{x}} \mathrm{ASW}(\boldsymbol{x}; \boldsymbol{t}) - f(\boldsymbol{t}_{-i}) = \mathrm{ASW}(\boldsymbol{x}^d(\boldsymbol{t}); \boldsymbol{t}) - f(\boldsymbol{t}_{-i}) \end{aligned} \tag{22}$$

where RHS of Equation (22) is the utility of truthful reporting for player i. Therefore, NAM λ^{NAM} is truthful. □

B.3 PROOF OF THEOREM 3.2

Theorem 3.2. *Fix $M^{AMA} = \{(\boldsymbol{x}^k, p^k)\}_{k\in[M]}$ to be an arbitrary AMA mechanism. Then, there exists an NAM $\lambda^{NAM} : \mathcal{X} \to \mathbb{R}$ such that:*

(a) λ^{NAM} is concave on $\mathcal{X}$.

(b) If direct mechanism M^d is equivalent to M^{AMA}, then M^d is also equivalent to λ^{NAM}.

Proof. Our proof consists of three step: (a) We construct an extended AMA $\widetilde{M}^{\text{AMA}}$ that is equivalent with M^{AMA} and its outcome candidates be sufficiently "broad"; (b) Based on $\widetilde{M}^{\text{AMA}}$, we construct the λ^{NAM} and verify the concavity and continuity; (c) We show that if M^d is equivalent with $\widetilde{M}^{\text{AMA}}$, then M^d is equivalent with λ^{NAM}.

Step (a). We first construct an extended AMA whose outcome candidates is sufficiently "broad".

Specifically, let $(y^1, ..., y^L)$ be all vertices of $\mathcal{X}$ and $C \geq 0$ as a sufficiently large number. We consider the extended AMA $\widetilde{M}^{\text{AMA}} = \{(\boldsymbol{x}^k, p^k)\}_{k\in[M+L]}$ where $\boldsymbol{x}^{M+j} = y^j, p^{M+j} = -C$ for $j \in [L]$. The C is chosen such that for all direct mechanism M^d that is equivalent with M^{AMA}, M^d is also equivalent with $\widetilde{M}^{\text{AMA}}$.

We show that such C always exists. Let $k \in [M]$ be arbitrary index. Consider the following inequality:

$$\begin{aligned}
&\forall j \in [L], \boldsymbol{t} \in \mathcal{T}, \quad \text{ASW}(k; \boldsymbol{t}) > \text{ASW}(M + j; \boldsymbol{t}) \Leftrightarrow \forall j \in [L], \boldsymbol{t} \in \mathcal{T}, \quad \sum_i \langle t_i, x_i^k \rangle + \lambda^k > \sum_i \langle t_i, y_i^j \rangle - C \\
\Leftrightarrow &\forall j \in [L], \boldsymbol{t} \in \mathcal{T}, \quad C > \sum_i \langle t_i, y_i^j \rangle - \sum_i \langle t_i, x_i^k \rangle - \lambda^k
\end{aligned} \tag{23}$$

Since RHS is linear on $\boldsymbol{t}$, $\mathcal{T}$ is compact and $[L]$ is finite, we know that there is $C^* > 0$ such that inequality (23) holds.

Define $\tilde{k}^*(\boldsymbol{t}) = \arg\max_{k\in[M+L]} \text{ASW}(k; \boldsymbol{t})$. Inequality (23) indicates $k^*(\boldsymbol{t}) = \tilde{t}^*(\boldsymbol{t})$ for all $\boldsymbol{t}$. Similarly, there is also $C_i^* > 0$ such that $k^{-i,*}(\boldsymbol{t}) = \tilde{t}^{-i,*}(\boldsymbol{t})$ for all $\boldsymbol{t}$. Let C be the maximum value of $C^*, \{C_i^*\}_{i\in[n]}$, we know that by choosing such value of C, the computation flow of M^{AMA} in Equation (2) is identical of the computation flow of $\widetilde{M}^{\text{AMA}}$. Therefore, M^d is equivalent with M^{AMA} indicates that M^d is equivalent with $\widetilde{M}^{\text{AMA}}$.

Step (b). We *w.l.o.g.* assumes that the outcome candidates of M^{AMA} cover all vertices of $\mathcal{X}$.

We define

$$\begin{aligned}
A(\boldsymbol{x}) &= \{\{\alpha^k\}_{k\in[M]} : \sum_{k\in[M]} \alpha^k \boldsymbol{x}^k = \boldsymbol{x}; \sum_{k\in[M]} \alpha^k = 1, \forall k \in [M], \alpha^k \geq 0\} \\
\lambda^{\text{NAM}}(\boldsymbol{x}) &= \sup_{\{\alpha^k\}_{k\in[M]} \in A(\boldsymbol{x})} \sum_{k\in[M]} \alpha^k \lambda^k
\end{aligned} \tag{24}$$

.

Here, $\boldsymbol{x}$ is a convex combination of $\{\boldsymbol{x}^k\}_{k\in[M]}$ with coefficient $\{\alpha^k\}_{k\in[M]}$. As $\{\boldsymbol{x}^k\}_{k\in[M]}$ covers all vertices of $\mathcal{X}$, we know that such convex combination always exists, therefore $A(\boldsymbol{x})$ is non-empty and the supreme is well-defined. Moreover, $A(\boldsymbol{x})$ is a close set. Therefore the supreme value can always be achieved at some $\{\alpha^k\}_{k\in[M]} \in A$.

We next show the concavity and continuity of λ^{NAM}. To show this we only need to prove the concavity and continuity of $A(\boldsymbol{x})$ (as a set-valued function). The concavity and continuity of λ^{NAM} can then be derived since it is the maximum value of a linear function ($\sum \alpha^k \lambda^k$) on $A(\boldsymbol{x})$.

First, let $\boldsymbol{x}_i, i = 1, 2$ be two outcomes and $\gamma\boldsymbol{x}_1 + (1-\gamma)\boldsymbol{x}_2, \gamma \in (0,1)$ be its convex combination. Let the maximum value of $\lambda^{\text{NAM}}(\boldsymbol{x}_i)$ is achieved on $\{\alpha_i^k\}_{k\in[K]}$. Define $\beta^k = \gamma\alpha_1^k + (1-\gamma)$. It's clear to see that $\{\beta^k\}_{k\in[K]} \in A(\gamma\boldsymbol{x}_1 + (1-\gamma)\boldsymbol{x}_2)$, showing that $A(\boldsymbol{x})$ is a concave set-valued function.

Next, let $\{\boldsymbol{x}_n\}_{n\in\mathbb{N}}$ be a sequence converging at $\boldsymbol{x}$ and $\alpha_n \coloneqq \{\alpha_n^k\}_{k\in[M]}$ be the corresponding coefficients of $\boldsymbol{x}_n$. Since $\alpha_n \in \Delta_M$ and Δ_M is compact, we know that there is a converging subsequent $(\alpha_{i_n}) \to \alpha_* = \{\alpha_*^k\}_{k\in[M]}$. As $\sum_{k\in[M]} \alpha_{i_n}^k \boldsymbol{x}^k = \boldsymbol{x}_{i_n}$, taking limitation we know that $\sum_{k\in[M]} \alpha_*^k \boldsymbol{x}^k = \boldsymbol{x}$. This means that $\alpha_* \in A(\boldsymbol{x})$ and therefore $A(\boldsymbol{x})$ is continuous.

Step (c). The last step is to show that M^d is equivalent with λ^{NAM} given M^d be equivalent with M^{AMA}. To see so, we need to verify the two inclusions and one equality in Equation (3), which is listed below:

$$\begin{aligned}
\boldsymbol{t}[-i] \coloneqq& (t_1, ..., t_{i-1}, \mathbf{0}_m, t_{i+1}, ..., t_n) \\
\text{ASW}(\boldsymbol{x}; \boldsymbol{t}) \coloneqq& \sum_{i\in[n]} \langle t_i, x_i\rangle + \lambda^{\text{NAM}}(\boldsymbol{x}) \\
\boldsymbol{x}^d(\boldsymbol{t}) \in& \arg\max_{\boldsymbol{x}\in\mathcal{X}} \text{ASW}(\boldsymbol{x}; \boldsymbol{t}) \\
\boldsymbol{x}^{-i}(\boldsymbol{t}) \in& \arg\max_{\boldsymbol{x}\in\mathcal{X}} \text{ASW}(\boldsymbol{x}; \boldsymbol{t}[-i]) \\
p_i^d(\boldsymbol{t}) =& \text{ASW}(\boldsymbol{x}^{-i}(\boldsymbol{t}); \boldsymbol{t}[-i]) - \text{ASW}(\boldsymbol{x}^d(\boldsymbol{t}); \boldsymbol{t}[-i])
\end{aligned} \tag{25}$$

We first show that the first inclusion holds ($\boldsymbol{x}^d(\boldsymbol{t}) \in \arg\max_{\boldsymbol{x}\in\mathcal{X}} \text{ASW}(\boldsymbol{x}; \boldsymbol{t})$). Since a direct mechanism does not directly have $\boldsymbol{x}^{-i}(\boldsymbol{t})$, we would like to define that $\boldsymbol{x}^{-i}(\boldsymbol{t}) = \boldsymbol{x}^{k^{-i,*}(\boldsymbol{t})}$. Then the second inclusion is similar to prove and is thus omitted.

Fix $\boldsymbol{t}$. Assume the contrary that $\boldsymbol{x}^d(\boldsymbol{t})$ is not the maximum point of $\text{ASW}(\cdot; \boldsymbol{t})$. Then, there is $\boldsymbol{x}^*$ such that $\text{ASW}(\boldsymbol{x}^*; \boldsymbol{t}) > \text{ASW}(\boldsymbol{x}^d(\boldsymbol{t}); \boldsymbol{t})$. By definition of λ^{NAM}, there exists $\alpha \in A(\boldsymbol{x}^*)$ such that $\sum_k \alpha^k \boldsymbol{x}^k = \boldsymbol{x}^*$ and $\sum_k \alpha^k \lambda^k = \lambda^{\text{NAM}}(\boldsymbol{x}^*)$. We have,

$$\begin{aligned}
\text{ASW}(\boldsymbol{x}^*; \boldsymbol{t}) &= \sum_{i\in[n]} \langle t_i, x_i^*\rangle + \lambda^{\text{NAM}}(\boldsymbol{x}^*) \\
&= \sum_{i\in[n]} \sum_{k\in[M]} \alpha^k \langle t_i, x_i^k\rangle + \sum_{k\in[M]} \alpha^k \lambda^{\text{NAM}}(\boldsymbol{x}^k) \\
&= \sum_{k\in[M]} \alpha^k \text{ASW}(\boldsymbol{x}^k; \boldsymbol{t})
\end{aligned} \tag{26}$$

Therefore, there is $k \in [M]$ such that $\text{ASW}(\boldsymbol{x}^k; \boldsymbol{t}) \geq \text{ASW}(\boldsymbol{x}^*; \boldsymbol{t}) > \text{ASW}(\boldsymbol{x}^d(\boldsymbol{t}); \boldsymbol{t}) = \max_{k'\in[M]} \text{ASW}(\boldsymbol{x}^{k'}; \boldsymbol{t})$, which leads a contradiction. The last equation is due to the definition of equivalence between M^d and M^{AMA} in Equation (2).

Lastly, we verify the equation $p_i^d(\boldsymbol{t}) = \text{ASW}(\boldsymbol{x}^{-i}(\boldsymbol{t}); \boldsymbol{t}[-i]) - \text{ASW}(\boldsymbol{x}^d(\boldsymbol{t}); \boldsymbol{t}[-i])$. Note that

$$p_i^d(\boldsymbol{t}) \coloneqq \text{ASW}_{-i}(k^{-i,*}(\boldsymbol{t}); \boldsymbol{t}) - \text{ASW}_{-i}(k^*(\boldsymbol{t}); \boldsymbol{t}) \tag{27}$$

We will prove independently that $\text{ASW}_{-i}(k^{-i,*}(\boldsymbol{t}); \boldsymbol{t}) = \text{ASW}(\boldsymbol{x}^{-i}(\boldsymbol{t}); \boldsymbol{t}[-i])$ and $\text{ASW}_{-i}(k^*(\boldsymbol{t}); \boldsymbol{t}) = \text{ASW}(\boldsymbol{x}^d(\boldsymbol{t}); \boldsymbol{t}[-i])$.

To show this, we first show that $\lambda^{k^*(\boldsymbol{t})} = \lambda^{\text{NAM}}(\boldsymbol{x}^d(\boldsymbol{t}))$

Since $\boldsymbol{x}^d(\boldsymbol{t}) = \boldsymbol{x}^{k^*(\boldsymbol{t})}$, it's straight-forward that $\lambda^{\text{NAM}}(\boldsymbol{x}^d(\boldsymbol{t})) \geq \lambda^{k^*(\boldsymbol{t})}$ by definition of λ^{NAM} and supreme. If fix some $\boldsymbol{t}$ and $\lambda^{\text{NAM}}(\boldsymbol{x}^d(\boldsymbol{t})) > \lambda^{k^*(\boldsymbol{t})}$, we could find some $\alpha \in A(\boldsymbol{t})$ such that,

$$\sum_k \alpha^k \lambda^k = \lambda^{\text{NAM}}(\boldsymbol{x}^d(\boldsymbol{t})), \qquad \sum_k \alpha^k \boldsymbol{x}^k = \boldsymbol{x}^d(\boldsymbol{t}) \tag{28}$$

Then,

$$\begin{aligned}\mathrm{ASW}(k^*(\boldsymbol{t});\boldsymbol{t}) =& \lambda^{k^*(\boldsymbol{t})} + \sum_{i\in[n]} \langle t_i, x_i^{k^*(\boldsymbol{t})}\rangle \\ =& \lambda^{k^*(\boldsymbol{t})} + \sum_{i\in[n]} \langle t_i, x_i^{d}(\boldsymbol{t})\rangle \cdots \text{by definition of } \boldsymbol{x}^d(\boldsymbol{t}) \\ <& \lambda^{\mathrm{NAM}}(\boldsymbol{x}^d(\boldsymbol{t})) + \sum_{i\in[n]} \langle t_i, x_i^{d}(\boldsymbol{t})\rangle \cdots \text{by assumption} \\ =& \sum_k \alpha^k \left(\lambda^k + \sum_{i\in[n]} \langle t_i, x_i^k\rangle\right) \cdots \text{by equation in Equation (28)} \\ \leq& \max_k \lambda^k + \sum_{i\in[n]} \langle t_i, x_i^k\rangle \\ =& \mathrm{ASW}(k^*(\boldsymbol{t});\boldsymbol{t})\end{aligned} \tag{29}$$

which leads a contradiction.

Similarly, we also have that $\lambda^{k^{-i,*}(\boldsymbol{t})} = \lambda^{\mathrm{NAM}}(\boldsymbol{x}^{-i}(\boldsymbol{t}))$.

Based on above, we have

$$\begin{aligned}\mathrm{ASW}_{-i}(k^{-i,*}(\boldsymbol{t});\boldsymbol{t}) =& \sum_{j\neq i} \langle t_j, x_j^{k^{-i,*}(\boldsymbol{t})}\rangle + \lambda^{k^{-i,*}(\boldsymbol{t})} \\ =& \sum_{j\neq i} \langle t_j, x_j^{-i}(\boldsymbol{t})\rangle + \lambda^{\mathrm{NAM}}(\boldsymbol{x}^{-i}(\boldsymbol{t})) = \mathrm{ASW}(\boldsymbol{x}^{-i}(\boldsymbol{t});\boldsymbol{t}[-i])\end{aligned} \tag{30}$$

and

$$\begin{aligned}\mathrm{ASW}_{-i}(k^{*}(\boldsymbol{t});\boldsymbol{t}) =& \sum_{j\neq i} \langle t_j, x_j^{k^{*}(\boldsymbol{t})}\rangle + \lambda^{k^{*}(\boldsymbol{t})} \\ =& \sum_{j\neq i} \langle t_j, x_j^{d}(\boldsymbol{t})\rangle + \lambda^{\mathrm{NAM}}(\boldsymbol{x}^{d}(\boldsymbol{t})) = \mathrm{ASW}(\boldsymbol{x}^{d}(\boldsymbol{t});\boldsymbol{t}[-i])\end{aligned} \tag{31}$$

which together completes the proof. □

B.4 Proof of Theorem 4.2

Theorem 4.2. *Fix $\mathcal{X} \subseteq \mathbb{R}^{d_x}$ be a convex, compact subset of d_x-dimensional Euclidean space. Then, ICNN is a universal approximator of all functions $f(x) : \mathcal{X} \to \mathbb{R}$ that are convex and continuous on x.*

We begin with a rigorous definition about universal approximation.

Denote $\mathcal{H} = \mathbb{R}^{\mathcal{X}}$ to be the set of all functions with domain $\mathcal{X}$ and range $\mathbb{R}$. Let $\mathcal{F}, \mathcal{G} \subseteq \mathcal{H}$ be two subsets of such functions. We say that $\mathcal{F}$ is a universal approximator of $\mathcal{G}$ if the following two conditions hold:

(1) $\mathcal{F} \subseteq \mathcal{G}$

(2) For all $g \in \mathcal{G}, \varepsilon > 0$, there is $f \in \mathcal{F}$ such that $l_\infty(f, g) \leq \varepsilon$.

The following theorem is identical with the original statement.

Theorem B.1. *Let $\mathcal{G} = \{g \in \mathcal{H} : g \text{ is continuous and convex on } \mathcal{X}\}$ and $\mathcal{F} = \{f : f \text{ can be represented by an ICNN}\}$. Then, $\mathcal{F}$ is a universal approximator of $\mathcal{G}$.*

Proof of Theorem B.1. **Prove (1).**

Let $f : \mathcal{X} \to \mathbb{R}$ be a function represented by an ICNN. We will actually prove that f is continuous and convex.

First, notice that the composition of continuous functions is always continuous. Also note that in ICNN, all computation flows, including linear transformations and activation functions, are continuous transformations. Therefore, f is also continuous.

Second, recall that a non-negatively weighted sum of convex functions is convex, and that $g \circ h$ is convex if g is convex and non-decreasing and h is convex. In Figure 1, every element in h_1 is a linear function of x. Then, every element of x_1, which is a convex, non-decreasing function of some element in h_1, is a convex function of x. Now assume that every element of x_k is a convex function of x. Then, every element of h_{k+1}, which is a non-negatively weighted sum of x_k plus a linear function of x, is a convex function of x; every element of x_{k+1}, which is a convex, non-decreasing function of some element in h_{k+1}, is also convex in x. In all, we conclude that x_K is convex in x. Therefore, y, the output of ICNN, which is a non-negative-weighted sum of x_K, is convex on x. This completes the proof of (1).

Prove (2).

It is not hard to verify that if $\mathcal{F}_1$ is a universal approximator of $\mathcal{F}_2$ and $\mathcal{F}_2$ is a universal approximator of $\mathcal{F}_3$, then $\mathcal{F}_1$ is a universal approximator of $\mathcal{F}_3$.

A well-known result is that the set of max-of-affine functions is a universal approximator of $\mathcal{G}$ (Calafiore et al., 2019). Therefore, we only need to prove that every max-of-affine function can be exactly represented by some ICNN.

A max-of-affine function is represented by

$$f(x) = \max_{i \in [n]} \quad \langle w_i, x \rangle + b_i \tag{32}$$

with $b_i \in \mathbb{R}, w_i \in \mathbb{R}^{d_x}$.

We will prove this by mathematical induction on n, and prove that every max-of-n-affine function can be exactly represented by ICNN with $d_h = 1$.

When $n = 1$, $f(x) = \langle w_1, x \rangle + b_1$. Letting $K = 0$, $w^r = w_1$, and $b = b_1$ directly finishes the proof.

Now let $n \geq 2$. By the inductive hypothesis, every max-of-$(n-1)$-affine function can be exactly represented by ICNN. It is clear that Equation (32) can be rewritten as:

$$f(x) = \langle w_n, x \rangle + b_n + \max(0, g(x)) \tag{33}$$

where $g(x) = \max_{i \in [n-1]} \langle w_i - w_n, x \rangle + (b_i - b_n)$. We know that $g(x)$ can be represented by a K-layer ICNN.

Let θ, including $\{W_k^c\}_{1 \leq k \leq K-1}$, $\{W_k^r\}_{0 \leq k \leq K-1}$, $\{b_k\}_{0 \leq k \leq K-1}$, $w^c \in \mathbb{R}_+$, $w^r \in \mathbb{R}^{d_x}$, $b \in \mathbb{R}$, $\{\sigma_k\}_{k \in [K]}$, where $W_k^c \in \mathbb{R}_+$, $W_k^r \in \mathbb{R}^{d_x}$, $b_k \in \mathbb{R}$, be the K-layer ICNN that represents $g(x)$.

Next, we construct a $K+1$-layer ICNN with parameters θ' including $\{W_k'^c\}_{1 \leq k \leq K}$, $\{W_k'^r\}_{0 \leq k \leq K}$, $\{b_k'\}_{0 \leq k \leq K}$, $w'^c \in \mathbb{R}_+$, $w'^r \in \mathbb{R}^{d_x}$, $b' \in \mathbb{R}$, $\{\sigma_k'\}_{k \in [K+1]}$, where $W_k'^c \in \mathbb{R}_+$, $W_k'^r \in \mathbb{R}^{d_x}$, $b_k \in \mathbb{R}$.

We set $W_k'^c = W_k^c, W_k'^r = W_k^r, b_k' = b_k$ for $k \leq K-1$, $\sigma_k' = \sigma_k$ for $k \in [K]$ and $W_K'^c = w^c, W_K'^r = w^r, b_K' = b$. It's clear to see that $h_{K+1}(x; \theta') = y(x; \theta) = g(x)$, where $h_{K+1}(x; \theta')$ represents the hidden element h_{K+1} in the ICNN parameterized by θ'.

Let $\sigma_{K+1}(\cdot) = \mathrm{RELU}(\cdot)$, then $x_{K+1}(x; \theta') = \max(0, h_{K+1}(x; \theta')) = \max(0, g(x))$. Let $w_c' = 1, w_r' = w_n, b' = b_n$, we know that $y(x; \theta') = w_c' \cdot x_{K+1}(x; \theta') + \langle w_r', x \rangle + b' = \langle w_n, x \rangle + b_n + \max(0, g(x)) = f(x)$. Therefore, we complete the proof.

□

B.5 Proof of Proposition 4.3

Proposition 4.3. *Following identity holds universally,*

$$\nabla_\theta \mathbb{E}_{\tilde{\boldsymbol{x}}_i}[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})] = \mathbb{E}_{\tilde{\boldsymbol{x}}_i}\left[\nabla u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\right] + \beta \cdot \mathrm{Cov}_{\tilde{\boldsymbol{x}}_i}\left[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t}), \nabla_\theta \sum_{i \in [n+1]} \mathrm{ASW}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{t}}_i, \theta)\right] \tag{10}$$

where $\tilde{\boldsymbol{x}}_i \overset{\text{i.d.}}{\sim} q^\beta(\cdot|\tilde{\boldsymbol{t}}_i, \theta)$.

Proof. Fix $\boldsymbol{t}$ and $\beta > 0$. For each $i \in [n+1]$, write

$$q_i(\tilde{\boldsymbol{x}}_i;\theta) \;:=\; q_\beta(\tilde{\boldsymbol{x}}_i \mid \tilde{\boldsymbol{t}}_i, \theta) \;=\; \frac{\exp\big(\beta \cdot \mathrm{ASW}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{t}}_i, \theta)\big)}{Z_\beta(\tilde{\boldsymbol{t}}_i, \theta)}, \qquad Z_\beta(\tilde{\boldsymbol{t}}_i, \theta) \;:=\; \int_{a \in \mathcal{X}} \exp\big(\beta \cdot \mathrm{ASW}(a; \tilde{\boldsymbol{t}}_i, \theta)\big)\, \mathrm{d}a.$$

Let $\tilde{\boldsymbol{x}} = (\tilde{\boldsymbol{x}}_i)_{i\in[n+1]} \in \mathcal{X}^{n+1}$ and define the product density $q(\tilde{\boldsymbol{x}};\theta) := \prod_{i\in[n+1]} q_i(\tilde{\boldsymbol{x}}_i;\theta)$. Then

$$\mathbb{E}_{\tilde{\boldsymbol{x}}_i \sim q_\beta(\cdot|\tilde{\boldsymbol{t}}_i,\theta)}\big[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\big] = \int_{\tilde{\boldsymbol{x}} \in \mathcal{X}^{n+1}} u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\, q(\tilde{\boldsymbol{x}};\theta) \prod_{i\in[n+1]} \mathrm{d}x_i.$$

By the chain rule of differentiation,

$$\begin{aligned}\nabla_\theta \mathbb{E}_{\tilde{\boldsymbol{x}}_i}\big[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\big] &= \int \nabla_\theta u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\, q(\tilde{\boldsymbol{x}};\theta) \prod_i \mathrm{d}x_i \;+\; \int u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\, \nabla_\theta q(\tilde{\boldsymbol{x}};\theta) \prod_i \mathrm{d}x_i \\ &= \mathbb{E}_{\tilde{\boldsymbol{x}}_i}\big[\nabla_\theta u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\big] \;+\; \mathbb{E}_{\tilde{\boldsymbol{x}}_i}\Big[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\, \nabla_\theta \log q(\tilde{\boldsymbol{x}};\theta)\Big],\end{aligned} \tag{34}$$

where we used the log-gradient identity $\nabla_\theta q = q\, \nabla_\theta \log q$.

Since $q(\tilde{\boldsymbol{x}};\theta) = \prod_i q_i(\tilde{\boldsymbol{x}}_i;\theta)$, we have $\nabla_\theta \log q(\tilde{\boldsymbol{x}};\theta) = \sum_{i\in[n+1]} \nabla_\theta \log q_i(\tilde{\boldsymbol{x}}_i;\theta)$. Moreover,

$$\log q_i(a;\theta) = \beta\, \mathrm{ASW}(a; \tilde{\boldsymbol{t}}_i, \theta) - \log Z_\beta(\tilde{\boldsymbol{t}}_i, \theta),$$

so

$$\nabla_\theta \log q_i(a;\theta) = \beta\, \nabla_\theta \mathrm{ASW}(a; \tilde{\boldsymbol{t}}_i, \theta) - \nabla_\theta \log Z_\beta(\tilde{\boldsymbol{t}}_i, \theta).$$

A direct calculation gives

$$\begin{aligned}\nabla_\theta Z_\beta(\tilde{\boldsymbol{t}}_i, \theta) &= \int_{a\in\mathcal{X}} \exp\big(\beta \cdot \mathrm{ASW}(a; \tilde{\boldsymbol{t}}_i, \theta)\big) \cdot \beta\, \nabla_\theta \mathrm{ASW}(a; \tilde{\boldsymbol{t}}_i, \theta)\, \mathrm{d}a, \\ \nabla_\theta \log Z_\beta(\tilde{\boldsymbol{t}}_i, \theta) &= \frac{\nabla_\theta Z_\beta(\tilde{\boldsymbol{t}}_i, \theta)}{Z_\beta(\tilde{\boldsymbol{t}}_i, \theta)} = \beta \int_{a\in\mathcal{X}} q_i(a;\theta)\, \nabla_\theta \mathrm{ASW}(a; \tilde{\boldsymbol{t}}_i, \theta)\, \mathrm{d}a \\ &= \beta\, \mathbb{E}_{\tilde{\boldsymbol{x}}_i \sim q_i(\cdot;\theta)}\big[\nabla_\theta \mathrm{ASW}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{t}}_i, \theta)\big].\end{aligned}$$

Therefore, for each $i \in [n+1]$,

$$\nabla_\theta \log q_i(\tilde{\boldsymbol{x}}_i;\theta) = \beta\Big(\nabla_\theta \mathrm{ASW}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{t}}_i, \theta) - \mathbb{E}_{\tilde{\boldsymbol{x}}_i}\big[\nabla_\theta \mathrm{ASW}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{t}}_i, \theta)\big]\Big).$$

Plugging this back yields

$$\begin{aligned}\mathbb{E}_{\tilde{\boldsymbol{x}}_i}\Big[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\, \nabla_\theta \log q(\tilde{\boldsymbol{x}};\theta)\Big] &= \sum_{i\in[n+1]} \mathbb{E}_{\tilde{\boldsymbol{x}}_i}\Big[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\, \nabla_\theta \log q_i(\tilde{\boldsymbol{x}}_i;\theta)\Big] \\ &= \beta \sum_{i\in[n+1]} \Big(\mathbb{E}\big[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})\, \nabla_\theta \mathrm{ASW}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{t}}_i, \theta)\big] - \mathbb{E}[u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t})]\, \mathbb{E}[\nabla_\theta \mathrm{ASW}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{t}}_i, \theta)]\Big) \\ &= \beta\, \mathrm{Cov}_{\tilde{\boldsymbol{x}}_i}\Big(u_0(\tilde{\boldsymbol{x}}, \theta; \boldsymbol{t}),\, \nabla_\theta \sum_{i\in[n+1]} \mathrm{ASW}(\tilde{\boldsymbol{x}}_i; \tilde{\boldsymbol{t}}_i, \theta)\Big),\end{aligned} \tag{35}$$

Combining Equation (35) with Equation (34) proves Proposition 4.3.

□

C A VISUALIZATION OF $U_{2,2}$

We visualize the mechanism learned by NAM in $U_{2,2}$ in Figures 4 to 6. Specifically, we fix buyer 2's type to be $t_2 = (x, y)$, where $x, y \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. We let t_1 vary over $[0, 1]^2$ and visualize the allocation and monetary transfer to buyer 1, namely $x_{1,1}, x_{1,2}, p_1$, in each figure.

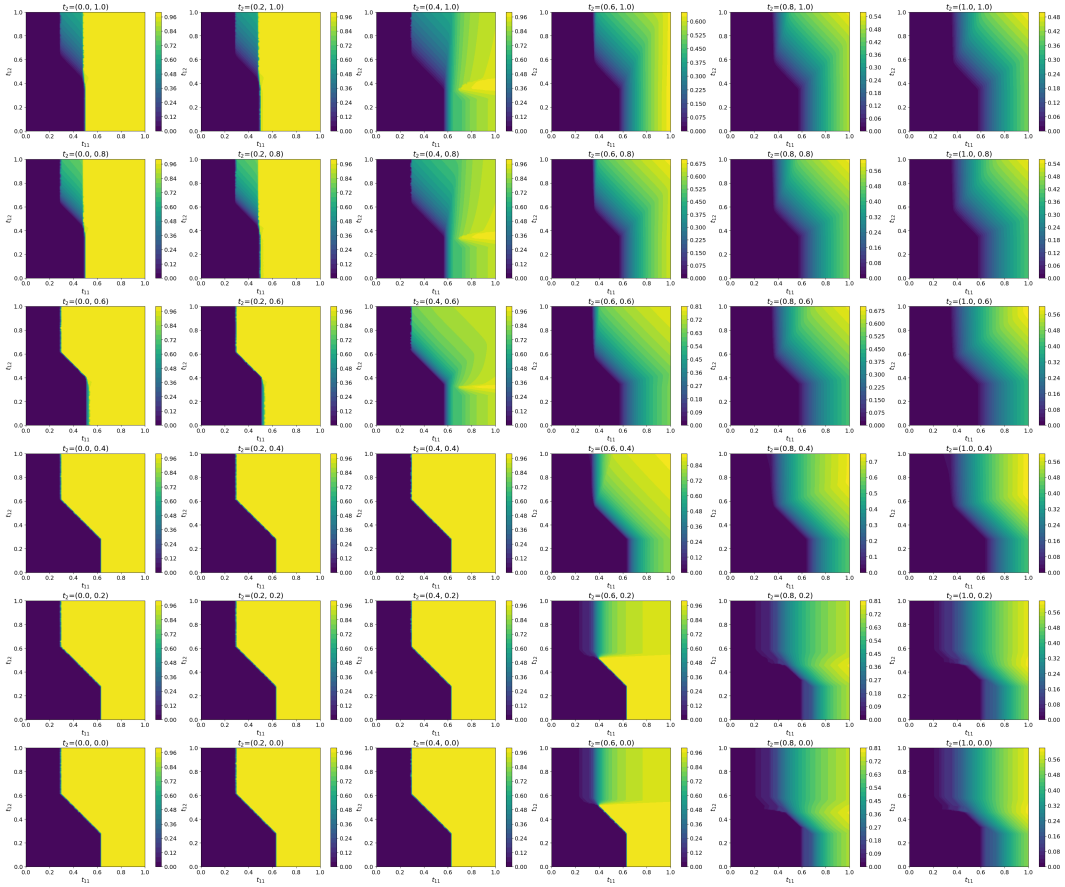


Figure 4: The visualization of $x_{1,1}$.

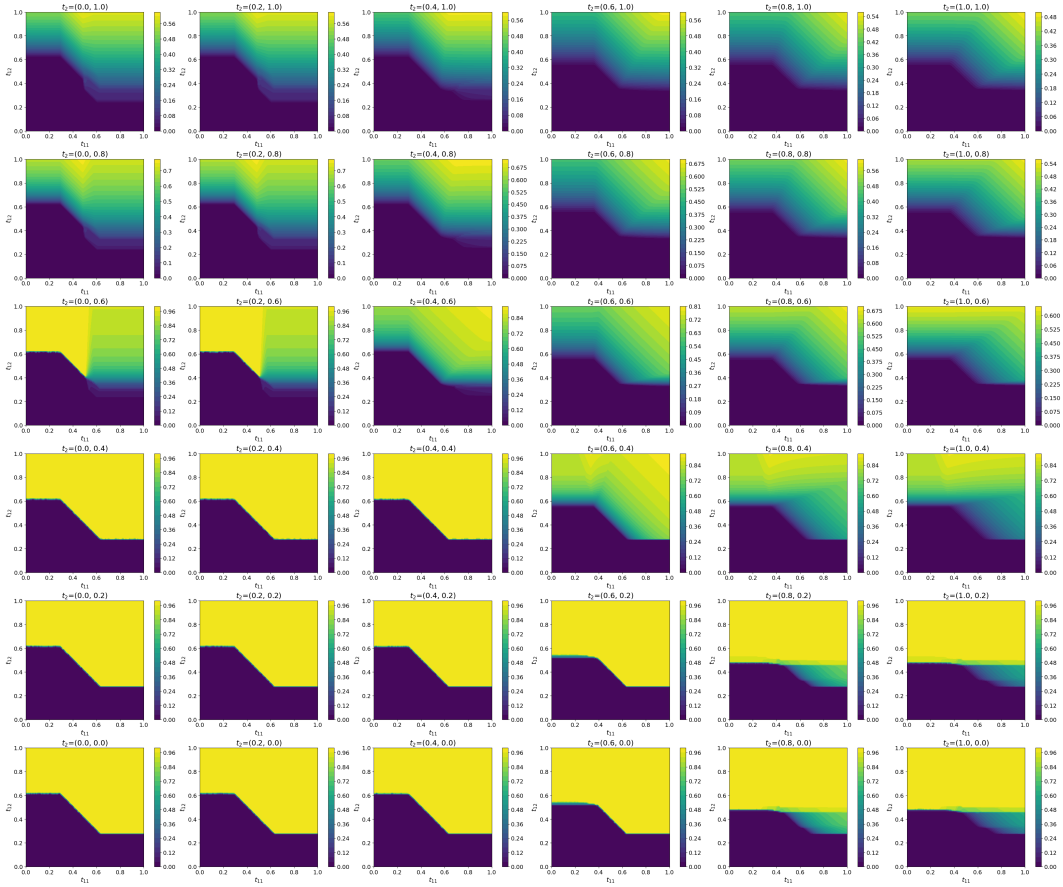


Figure 5: The visualization of $x_{1,2}$.

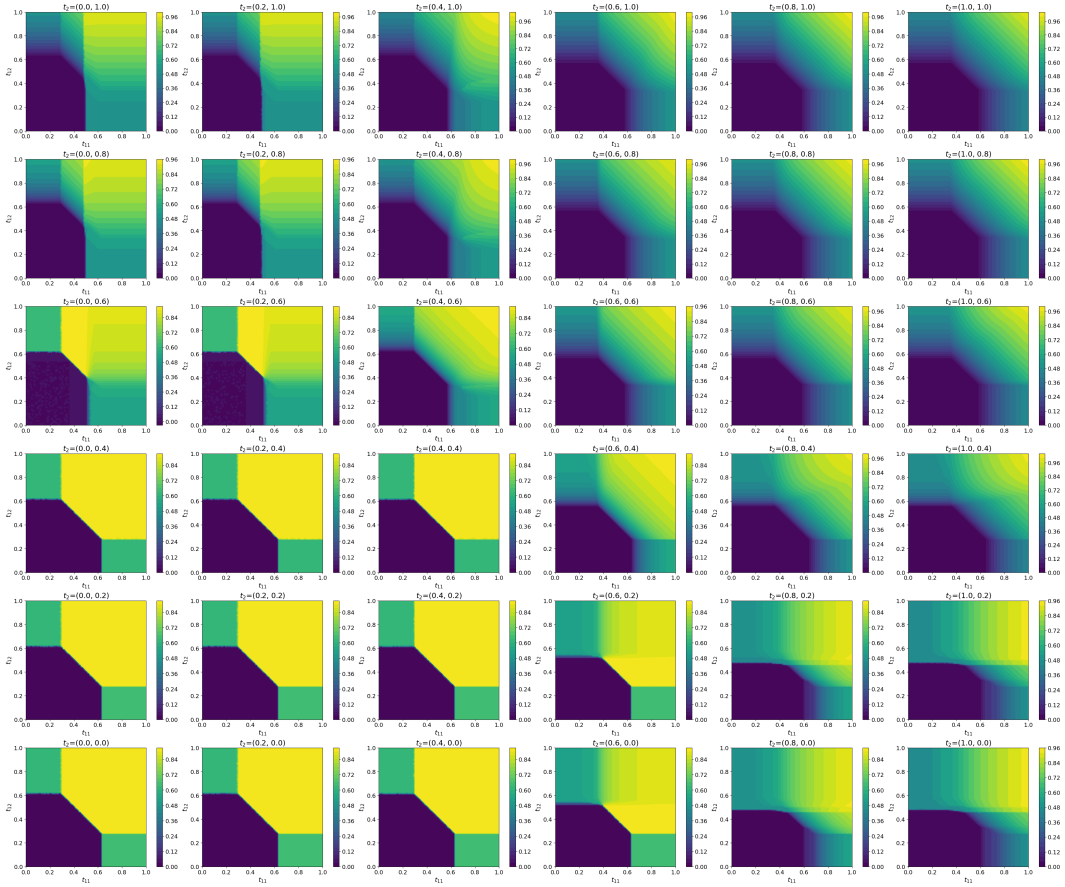


Figure 6: The visualization of p_1.

D More Experimental Details

We list the hyperparameters used for NAM in Table 3. Recall that n is the number of buyers and m is the number of goods.

Table 3: Hyperparameters.

Hyperparameter	Value
samples for training: M	2^{14}
samples for evaluation:	2^{14}
ICNN network layer: K	3
ICNN network hidden dimension: d_h	$10nm$
ICNN initial learning rate	$3e-4$
ICNN final learning rate	$1e-4$
ICNN optimizer	Adam($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1e-8$)
step size for continuous sampling	$4e-2$
continuous sampling iteration T	32
training iterations	$10,000$
temperature: β	512 ($<$ 5buyers) & $2,048$ ($\geq$ 5buyers)
menu size (for LotteryAMA and AMenuNet)	$8(n+m)^2$

For other hyperparameters (including training iterations, batch size and temperatures for baselines, as well as AMenuNet's network architecture), we use the same values reported in the original papers (Curry et al., 2023; Duan et al., 2023b).

D.1 Complexity Analysis

As ICNN has a fully connected architecture, computing each $\mathrm{ASW}(\boldsymbol{x})$ takes $\Theta(d_h^2) = \Theta(n^2m^2)$ time. Since one inference of NAM requires $n+1$ evaluations of $\mathrm{ASW}(\boldsymbol{x})$, the total training time for NAM is $\Theta(n^3m^2)$.

For both LotteryAMA and AMenuNet, one inference of $k^*(\boldsymbol{t})$ requires computing the $\arg\max$ of $\mathrm{ASW}(k;\boldsymbol{t})$ over all candidate outcomes $\boldsymbol{x}^k$. In addition, each candidate outcome is of size nm. As a consequence, the total training time for LotteryAMA and AMenuNet is $\Theta((n+1)\cdot 8(n+m)^2\cdot nm) = \Omega(n^3m^2)$.