

Learning Bayesian Game Families, with Application to Mechanism Design

Madelyn Gatchel
University of Michigan
Ann Arbor, MI, USA
gatchel@umich.edu

Michael P. Wellman
University of Michigan
Ann Arbor, MI, USA
wellman@umich.edu

ABSTRACT

Learning or estimating game models from data typically entails inducing separate models for each setting, even if the games are parametrically related. In empirical mechanism design, for example, this approach requires learning a new game model for each candidate setting of the mechanism parameter. Recent work has shown the data efficiency benefits of learning a single parameterized model for families of related games. In *Bayesian games*—a typical model for mechanism design—payoffs depend on both the actions and types of the players. We show how to exploit this structure by learning an *interim* game-family model that conditions on a single player’s type. We compare this to the baseline approach of directly learning the *ex ante* payoff function, which gives payoffs in expectation of all player types. By marginalizing over player type, the interim model can also provide *ex ante* payoff predictions, as necessary for Bayes-Nash equilibrium approximation. We also leverage the interim model to compute new beneficial piecewise best-response strategies, without any additional sample data. We validate our method through a case study of a dynamic sponsored search auction. For both payoff accuracy and Nash-approximation error, the interim model matches the *ex ante* model on the trained range, and outperforms *ex ante* in extrapolation. Our case study demonstrates that Bayesian game-family models can support comprehensive mechanism design, and that through interim-stage modeling we can enhance expressivity and reliability.

KEYWORDS

Game-Model Learning; Empirical Mechanism Design

ACM Reference Format:

Madelyn Gatchel and Michael P. Wellman. 2026. Learning Bayesian Game Families, with Application to Mechanism Design. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026)*, Paphos, Cyprus, May 25 – 29, 2026, IFAAMAS, 9 pages. <https://doi.org/10.65109/BAHE1423>

1 INTRODUCTION

Many real-world strategic interactions can be modeled as *Bayesian games*, where payoffs depend on players’ actions and their private information, or *types*. Outcomes may also depend on environment parameters, such that each parameter setting induces a different Bayesian game. Given limited modeling resources, analysts must

decide in advance the parameter range and its granularity. While domain knowledge may guide this selection, it cannot guarantee coverage of all salient settings. Ideally, analysts would have a model of the *Bayesian game family* from which they could reason about any relevant parameter setting.

A motivating application for Bayesian game families is *mechanism design*, where a designer sets or influences an environment parameter that affects strategic incentives. Each parameter value results in a different *game instance*. The designer’s goal is to find the parameter setting that optimizes a relevant objective function, such as social welfare or revenue. In *empirical mechanism design* (EMD), game model instances are induced from simulation data. Past EMD studies [4, 12, 23] have generally focused on a limited set of mechanism settings, separately modeling and analyzing each game instance.

Gatchel and Wiedenbeck [10] demonstrated that learning a single parameterized payoff model for families of related normal-form games is more data-efficient than training separate models for each game instance. We extend this approach to Bayesian game families, exploiting the type-conditional form of strategies for these games. Specifically, we investigate the learning of *interim payoff functions*, which explicitly condition on a single player’s type. By marginalizing out this type, we obtain the *ex ante payoff functions* which are essentially the payoffs learned in a normal-form model. We also explore learning *ex ante* payoff functions directly, and compare *ex ante* and interim learning approaches.

We validate our method through an EMD case study in the domain of sponsored search [14], where the publisher sets an auction reserve requirement to maximize revenue in equilibrium. Our search auction model is designed to capture the dynamic nature of bidding, where advertisers can revise their bids based on provisional results of earlier bidding rounds. We do so in a two-stage scenario, in which the bidders can *attempt* to modify their bids given the state of bidding after the first round. These attempts succeed probabilistically, thus providing an incentive for the players to submit meaningful first-round bids. The scenario is simple to describe and design heuristic strategies, yet too complex for straightforward analytic solution. We implement an agent-based simulation model of the scenario, and from the simulation-generated data learn Bayesian game-family models to support empirical game-theoretic analysis and mechanism design.

In our experiments, both *ex ante* and interim models achieve low payoff error on the trained parameter range, yet only the interim model maintains low error in extrapolation. Equilibria approximated using the interim model have regret error comparable to *ex ante* within the trained range and lower error beyond it. We also demonstrate that the learned game-family models support effective



This work is licensed under a Creative Commons Attribution International 4.0 License.

EMD procedures. Analysis of the expected revenue curve using a fine-grained parameter grid reveals the benefit of learning multiple game-family models. This analysis also provides concrete evidence that the interim model’s extrapolation capability is advantageous for mechanism design. A final advantage of the interim model is its ability to generate new strategies that outperform those in the original set. We introduce *piecewise-conditional* strategies that select from the original strategies based on the bidder’s own type, and we show how to construct such strategies that beneficially respond to equilibria over the original strategy set. Our investigation produces new insights about alternative model forms for Bayesian game families, and provides compelling experimental evidence in favor of learning interim payoff functions.

2 RELATED WORK

2.1 Learning Game Models from Data

In *empirical game-theoretic analysis* (EGTA) [27], complex multi-agent interactions are represented by an agent-based simulation model. Data from agent-based simulation is used to induce a formal game model, called an *empirical game*, which can be analyzed using standard methods, for example to derive approximate equilibria.¹ Given limited sampling resources, it is infeasible to estimate payoffs for every strategy profile from simulation data.

An alternative is to learn game models that generalize from the available data, such as using regression to learn pure-strategy payoff functions [25, 28] or inducing compact game structures [5, 8, 15, 17]. From a theoretical perspective, Fu and Lin [9] demonstrate that in non-truthful auctions, the interim utility function admits polynomial sample complexity, but the ex ante utility function does not. Sokota et al. [19] train a neural network to learn the *deviation payoff function*—the expected utility for a unilateral deviator—for symmetric normal-form empirical game instances. This technique leverages the compactness of payoff functions in role-symmetric games. The learned deviation payoff function supports equilibrium computation without the combinatorial mixture summations required for direct payoff functions. Li and Wellman [16] learn deviation payoffs as part of a method to approximate mixed-strategy Bayes-Nash equilibria through iterative evolutionary search. Gatchel and Wiedenbeck [10] learn the deviation payoff function for families of related symmetric normal-form game instances, showing that this approach achieves higher payoff accuracy with less data than learning separate models for each game instance.

2.2 Empirical Mechanism Design

Vorobeychik et al. [23] analyze five game instances with different storage costs in a supply-chain management scenario and find that none deter initial procurement without sacrificing profitability. Jordan et al. [12] examine how reserve scores impact revenue in online advertising auctions, analyzing 14 game instances and identifying the revenue-maximizing score. Brinkman and Wellman [4] investigate the optimal clearing interval in a frequent call market to maximize allocative efficiency as the number of agents and trade opportunities are varied. Most recently, Wang et al. [26] evaluate the effectiveness of different security levels at thwarting cyber attacks

¹Games defined in terms of simulation models have also been called *simulation-based* [1, 18, 19] or *black-box games* [30].

in mixnets, analyzing four game instances as part of an empirical mechanism design study.

Vorobeychik et al. [24] present an automated mechanism design procedure, framing EMD as a black-box optimization problem compatible with any evaluation method. Evaluating a candidate mechanism setting is a subproblem that involves approximating an equilibrium in the induced game and computing the objective in equilibrium. Areyan Viqueira et al. [2] introduce a PAC-learning EMD framework that includes approximate equilibrium estimation and parameter search with Bayesian optimization. Zhang et al. [29] reformulate multi-player extensive-form mechanism design problems as 2-player zero-sum games and solve for optimal equilibria via learning.

Other approaches have used neural networks to learn mechanism solutions without explicit game modeling. Dütting et al. [6] train allocation and payment neural networks to learn auction rules that are incentive compatible. Bichler et al. [3] employ self-play policy iteration with neural networks to learn approximate Bayes-Nash equilibrium bidding strategies in symmetric auction games. Gemp et al. [11] learn the auctioneer’s utility as a function of contest design in empirical all-pay auctions or crowdsourcing contests.

3 BACKGROUND

A *symmetric Bayesian game* Γ has p players who share the same action set A , type space T , and utility function U . In this work, we assume that A is finite, and that T is infinite and ordered. Each player has a private type $t \in T$, which we assume is drawn independently from a common probability distribution $\mu \in \Delta(T)$.² The utility function is given by $U : A^p \times T^p \rightarrow \mathbb{R}$, where the payoff for a player playing action a with type t while opponents with types \vec{t} play actions \vec{a} , is denoted as $U((a, \vec{a}), (t, \vec{t}))$. Vectors \vec{t} and \vec{a} have length $p - 1$, have a consistent order, and specify the corresponding types and actions of the $p - 1$ opponents. To improve readability, we omit the inner parentheses and write $U(a, \vec{a}, t, \vec{t})$.

Let S be the set of pure strategies, each strategy $s \in S$ a function $s : T \rightarrow A$. We assume S is finite, thus a strict (typically highly restricted) subset of all mappings from types to actions. An *opponent profile* is a length- $(p - 1)$ vector $\vec{s} \in \vec{S}$ that specifies each opponent’s strategy, where the order of entries matches the order of opponents in the type vector \vec{t} . We assume access to an oracle (e.g., a simulator) that gives a noisy payoff estimate for a player with type t who plays strategy s_j , given $(\vec{s}, \vec{t}) : \tilde{U}(s_j(t), \vec{s}(\vec{t}), t, \vec{t})$. Given player symmetry, we can define a utility function u_j for each strategy s_j that maps opponent profiles to payoffs. We first define the *ex ante expected payoff*:

$$u_j(\vec{s}) = \mathbb{E}_{(t, \vec{t}) \sim \mu^p} \left[U(s_j(t), \vec{s}(\vec{t}), t, \vec{t}) \right]. \quad (1)$$

In the *interim* stage, each player knows its own type but has only probabilistic beliefs about the others. The *interim expected payoff* is given by

$$u_j(\vec{s} | t) = \mathbb{E}_{\vec{t} \sim \mu^{p-1}} \left[U(s_j(t), \vec{s}(\vec{t}), t, \vec{t}) \right]. \quad (2)$$

We can express ex ante as a marginalized version of interim: $u_j(\vec{s}) = \mathbb{E}_t u_j(\vec{s} | t)$.

²Note that the player symmetry is *ex ante* (prior to drawing types); once types are drawn the players are in distinct situations.

We learn the deviation payoff function, a mapping from mixed-strategy profiles to vectors of expected utilities for unilateral strategy deviations. Let $\sigma \in \Delta(S)$ denote a mixed strategy, where $\Delta(S)$ is the probability simplex over the strategy set S . Further, let $\vec{\sigma}$ refer to a symmetric mixed-strategy profile where all p (or $p - 1$) players are playing according to σ . We define the **ex ante** and **interim deviation payoffs** for deviating to strategy s_j :

$$u_j(\vec{\sigma}) = \sum_{\vec{s} \in \vec{S}} \Pr(\vec{s} | \vec{\sigma}) u_j(\vec{s}), \quad (3)$$

$$u_j(\vec{\sigma} | t) = \sum_{\vec{s} \in \vec{S}} \Pr(\vec{s} | \vec{\sigma}) u_j(\vec{s} | t). \quad (4)$$

We can write the ex ante deviation payoff for strategy s_j using the interim deviation payoff for strategy s_j :

$$u_j(\vec{\sigma}) = \mathbb{E}_{t \sim \mu} u_j(\vec{\sigma} | t) = \int_t u_j(\vec{\sigma} | t) \mu(t) dt.$$

The **ex ante deviation payoff function** is given by the $|S|$ -dimensional vector $u(\vec{\sigma}) = [u_j(\vec{\sigma})]$, over $s_j \in S$. Similarly, the **interim deviation payoff function** is $u(\vec{\sigma} | t)$. We use this unsubscripted lowercase u throughout to denote a vector of deviation payoffs. We define **regret** using deviation payoffs:

$$\epsilon(\vec{\sigma}) = \max_j u_j(\vec{\sigma}) - \sigma \cdot u(\vec{\sigma}). \quad (5)$$

A **symmetric Bayes-Nash Equilibrium** (BNE) is a symmetric profile $\vec{\sigma}$ such that no player has an incentive to deviate; equivalently, $\epsilon(\vec{\sigma}) = 0$. As empirical games are themselves approximations, we focus on finding ϵ -BNE, which are symmetric profiles $\vec{\sigma}$ such that $\epsilon(\vec{\sigma}) \leq \epsilon$, where ϵ is small.

Parameterized Game Families. Let $\Gamma(v)$ denote the symmetric Bayesian game instance where the environment parameter V takes value v . A **parameterized game family**, $\mathcal{G}(V)$, is the set of game instances $\{\Gamma(v) | v \in V\}$. In the game family, all payoff functions and functions that depend on payoffs (e.g., Eqs. 1–5) are parameterized by V . Because regret depends on V , a profile $\vec{\sigma}$ that is an ϵ -BNE in one game instance may have regret larger than ϵ in another game instance. Simulator payoff samples take the form $\tilde{U}(s_j(t), \vec{s}(\vec{t}), t, \vec{t}, v)$.

4 LEARNING BAYESIAN GAME FAMILIES

Simulator queries are costly, and the results are noisy due to randomness in strategies or in the game environment. We assume a fixed budget of simulator queries for learning and validation, so we must allocate queries across the parameter space, strategy space, and type space. This simulation data is used to train a model representing the deviation payoff function for a symmetric Bayesian game family. We experimentally compare ex ante and interim game-family learning methods. The ex ante method becomes equivalent to the normal-form approach developed by Gatchel and Wiedenbeck [10] once types are abstracted away, and thus serves as our baseline in the Bayesian setting.

4.1 Ex Ante Deviation Payoffs

We first train a neural network representing the ex ante deviation payoff function $\hat{u} : \Delta(S) \times V \rightarrow \mathbb{R}^{|S|}$, where $\hat{u}_j(\vec{\sigma}, v)$ is the predicted payoff a symmetric player would receive by deviating to

strategy s_j when all other players play according to $\vec{\sigma}$ in game instance $\Gamma(v)$. Let m denote the number of $(\vec{\sigma}, v)$ pairs in our training set, let o denote the number of observations per pair, and let $Q = m \cdot o$. An **observation** for a given $(\vec{\sigma}, v)$ and deviation strategy s_j involves sampling $\vec{s} \sim \vec{\sigma}$ and $(t, \vec{t}) \sim \mu^p$, and querying the simulator for $\tilde{U}(s_j(t), \vec{s}(\vec{t}), t, \vec{t}, v)$. A training example has the form $(\vec{\sigma}, v) \mapsto [\tilde{u}_j(\vec{\sigma}, v)]$, where $\tilde{u}_j(\vec{\sigma}, v)$ is the sample average of deviation payoffs across all observations for a given strategy s_j . The **ground truth deviation payoff** is given by $\hat{u}_j(\vec{\sigma}, v) = \frac{1}{o} \sum_{i=1}^o \tilde{U}(s_j(t^{(i)}), \vec{s}^{(i)}(\vec{t}^{(i)}), t^{(i)}, \vec{t}^{(i)}, v)$, where $(t^{(i)}, \vec{t}^{(i)}) \sim \mu^p$ and $\vec{s}^{(i)} \sim \vec{\sigma}$. We seek to minimize the training loss:

$$\frac{1}{m \cdot |S|} \sum_{k=1}^m \sum_{s_j \in S} \left(\tilde{u}_j(\vec{\sigma}^{(k)}, v^{(k)}) - \hat{u}_j(\vec{\sigma}^{(k)}, v^{(k)}) \right)^2. \quad (6)$$

4.2 Interim Deviation Payoffs

In Bayesian games, sampling over types represents a distinct source of noise in payoff estimates. By conditioning estimates on the deviator’s type, we can leverage type-specific information in each sample. We therefore propose learning the interim deviation payoff function, which is explicitly conditional on the symmetric deviator’s type: $\hat{u} : \Delta(S) \times V \times T \rightarrow \mathbb{R}^{|S|}$, where $\hat{u}(\vec{\sigma}, v | t)$ gives the predicted vector of conditional deviation payoffs, $[\hat{u}_j(\vec{\sigma}, v | t)]$ for each $s_j \in S$ in game instance $\Gamma(v)$. An interim training example is a mapping $(\vec{\sigma}, v, t^{(i)}) \mapsto [\tilde{u}_j(\vec{\sigma}, v | t^{(i)})]$, where $\tilde{u}_j(\vec{\sigma}, v | t^{(i)})$ corresponds to the deviation payoff from a single observation. We aim to minimize the training loss:

$$\frac{1}{Q|S|} \sum_{k=1}^m \sum_{i=1}^o \sum_{s_j \in S} \left(\tilde{u}_j(\vec{\sigma}^{(k)}, v^{(k)} | t^{(i)}) - \hat{u}_j(\vec{\sigma}^{(k)}, v^{(k)} | t^{(i)}) \right)^2.$$

We can compute ex ante deviation payoffs from the interim by marginalization: $\int_t \hat{u}_j(\vec{\sigma}, v | t) \mu(t) dt$. In practice, this can be done via Monte Carlo integration by averaging predicted interim deviation payoffs over many sampled deviator types. The marginalized interim deviation payoff vector for $(\vec{\sigma}, v)$ is approximated using n Monte Carlo (MC) samples $t^{(i)} \sim \mu$ and the learned interim model:

$$\hat{u}(\vec{\sigma}, v) = \frac{1}{n} \sum_{i=1}^n \hat{u}(\vec{\sigma}, v | t^{(i)}).$$

5 USING LEARNED BAYESIAN GAME FAMILIES

5.1 Deriving Equilibria

Adapting existing techniques [10, 19] to the Bayesian setting, we run an approximate Nash-finding algorithm on each game instance $\Gamma(v)$ using deviation payoffs predicted by the model: $[\hat{u}_j(\vec{\sigma}, v)]$ (ex ante) or $[\int_t \hat{u}_j(\vec{\sigma}, v | t) \mu(t) dt]$ (marginalized interim). The resulting mixed-strategy profile is a **candidate ϵ -equilibrium**, $\vec{\sigma}_*$, if the predicted regret is at most ϵ ; otherwise, the algorithm did not converge. We validate the candidate with a modest number of additional simulator queries to compute a high-fidelity regret estimate of $\vec{\sigma}_*$ in $\Gamma(v)$; if this too is at most ϵ , the approximate equilibrium is **confirmed**. Running the algorithm from multiple starting points yields a set of equilibria suitable for any selection method.

5.2 Empirical Mechanism Design (EMD)

EMD is a motivating application for learned Bayesian game families. Once trained, the game-family model can evaluate any game instance within—and often beyond—the trained range, supporting finer parameter searches than previous EMD approaches. When used in an optimization algorithm, it eliminates the need to train separate models at each iteration, reducing the algorithm’s dependence on the sampling budget. See App. ?? for pseudocode.³

5.3 Piecewise-Conditional Strategies

In a BNE, no player can gain by deviating to another strategy in S . If S includes all mappings from type to action, then an ex ante equilibrium is an interim equilibrium, as a player would also not wish to deviate conditional on its own type. Given a restricted set S —the norm for empirical games—a player *can* often benefit by deviating to a different strategy in S once its type is revealed. We consider a higher-order strategy that exploits such opportunities by selecting a base-level **atomic strategy** $s \in S$ conditional on revealed type t . We can approximate the optimal deviation by deriving a piecewise best response to $(\vec{\sigma}, v)$. Let C be a set of contiguous type intervals partitioning T . A **piecewise strategy** is a mapping $\psi : C \rightarrow S$ that selects an atomic strategy for the interval containing t .

The learned interim model can approximate the ex ante expected payoff for playing a piecewise strategy in a given strategic context (e.g., where opponents play the BNE). First, we collect a large set of samples \mathcal{N} from the type distribution $\mu(T)$. Next, we compute the deviation payoff vector conditioned on the deviator having a type in interval $C \in \mathcal{C}$:

$$\hat{u}(\vec{\sigma}, v | C) = \frac{1}{|\mathcal{N}_C|} \sum_{t^{(i)} \in \mathcal{N}_C} \hat{u}(\vec{\sigma}, v | t^{(i)}), \quad (7)$$

where $\mathcal{N}_C = \mathcal{N} \cap C$. The **piecewise best-response strategy**, ψ , assigns each interval C the best-response strategy conditioned on having type $t \in C$:

$$\psi(C) = \arg \max_{s_j \in S} \hat{u}_j(\vec{\sigma}, v | C) \quad (8)$$

We can equivalently express ψ as a mapping from types to atomic strategies: for $t \in C$, $\psi(t) \equiv \psi(C)$. Given $\hat{u}(\vec{\sigma}, v | C)$ and \mathcal{N}_C for each $C \in \mathcal{C}$, the **predicted deviation payoff for playing the piecewise strategy** against $\vec{\sigma}$ is:

$$\hat{u}_\psi(\vec{\sigma}, v) = \frac{1}{|\mathcal{N}|} \sum_{C \in \mathcal{C}} |\mathcal{N}_C| \cdot \hat{u}_{\psi(C)}(\vec{\sigma}, v | C). \quad (9)$$

With N simulator queries per \vec{s} , a high-fidelity estimate of the **true deviation payoff for playing the piecewise strategy** against $\vec{\sigma}$ is:

$$\tilde{u}_\psi(\vec{\sigma}, v) = \frac{1}{N} \sum_{\vec{s} \in \vec{S}} \Pr(\vec{s} | \vec{\sigma}) \sum_{i=1}^N \tilde{U}(\psi(t^{(i)}), \vec{s}(\vec{t}^{(i)}), t^{(i)}, \vec{t}^{(i)}, v). \quad (10)$$

6 DYNAMIC SPONSORED SEARCH AUCTIONS

Our case study evaluates game-family learning approaches for EMD in a dynamic sponsored search auction. A **sponsored search auction** allocates ad slots on a search page to advertisers based on

their bids. The publisher ranks bids based on the offered price—typically adjusted by factors such as click-through rates and advertiser quality—and allocates slots in descending order of effective bids. Many studies model sponsored search as a one-shot simultaneous-move game [7, 13, 21, 22], typically assuming perfect information on the basis that advertisers learn about competitors through repeated interactions. When a bidder submits a bid, they may glean information based on the resulting state which may be used to adjust their bid. This dynamic adaptation, however, is abstracted away by one-shot models, which thereby do not capture imperfect adaptation or transient effects.

Our formulation aims to capture a simple form of the missing dynamics, by modeling a two-step bidding process. In the first stage, players simultaneously submit their initial bid based only on their valuation (type). Each player is then informed about their tentative slot and its price, as well as the prices for other slots assuming competitor bids remained fixed. In the second stage, they may update their bid based on this observation. Updates are submitted simultaneously, and each is received “on time” by the auction with a specified probability. The final bids determine the slot allocations and prices. For example, a strategy may set the initial bid to valuation minus two and the final bid as its optimal response to the provisional state. As all players have the option to change their bids, a player may sometimes be worse off by changing rather than keeping their initial bid. Because the updates are only probabilistically successful, players have an incentive to submit meaningful bids in the first stage. This setup captures agents’ imperfect information about opponents and their ability to gain information and adjust heuristically through interaction in a single-shot game.

We study a symmetric dynamic auction with 5 players, 4 ad slots, and 10 strategies. The game family is parameterized by a **reserve requirement** r —the minimum effective bid to win a slot—typically set by the publisher. The mechanism, applied to final bids, is a weighted generalized second-price auction with quality-weighted reserves and Varian preference model [22]. Each advertiser has a quality score, q , and valuation, θ . To participate, a bidder’s **effective bid**—its quality-weighted bid—must meet the reserve. See App. ?? for more details, including strategy specification.

7 EX ANTE VS INTERIM LEARNING

We evaluate deviation-payoff performance for two learned models of the same Bayesian game family. The **ex ante** model inputs a symmetric mixed strategy $\vec{\sigma}$ and reserve r . The **interim** model also conditions on deviator quality score q and valuation θ , and its predictions are marginalized over (q, θ) for evaluation. With a training budget of Q simulator queries, we build six datasets satisfying $Q = m \cdot o$, each containing m $(\vec{\sigma}, r)$ pairs and o observations per pair; reserves range from 0.01 to 8. For each dataset, we train one ex ante neural network (NN) on m examples (payoffs averaged over the o observations) and one interim NN on all $m \cdot o$ observations. See App. ?? for training details.

We use two test sets: a noisy, realistically sized dataset for a large empirical game family, and a larger, less noisy dataset for method validation. Our results show that with sufficient marginalization samples, the interim model matches ex ante error, and exhibits

³All appendices may be found in the arXiv version: <https://arxiv.org/pdf/2502.14078>

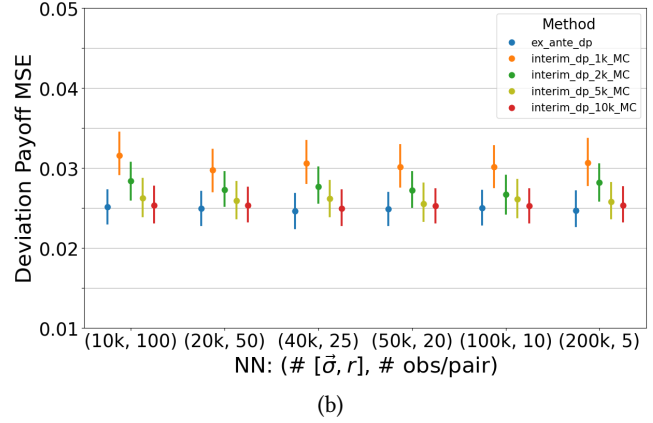
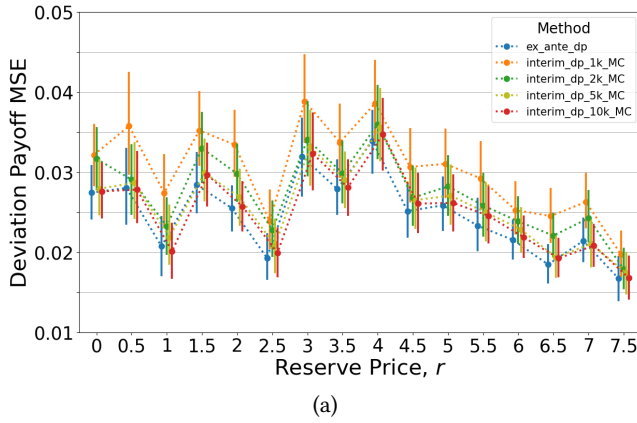


Figure 1: With sufficient marginalization samples, interim accuracy matches ex ante. Deviation payoff trends are consistent (a) across the reserve range and (b) across datasets.

better extrapolation. Performance trends on both test sets are consistent, suggesting the noisy test set provides reliable insights.

7.1 Model Learning (ML) Test Results

Fig. 1 compares deviation payoff errors among learned ex ante and interim game-family models. We compute the mean squared error (MSE) between ex ante or marginalized interim predictions and the ground truth using Eq. 6; error bars show 95% confidence intervals (CIs). We compute marginalized interim deviation payoffs by averaging predicted interim payoffs over $\{1k, 2k, 5k, 10k\}$ Monte Carlo samples of deviator quality and valuation. Plot 1(a) shows errors across the reserve price range, with test-set reserve prices binned in intervals of width 0.5; performance is aggregated across the six trained models for each method. The interim models with 5k and 10k Monte Carlo marginalization samples perform similarly to ex ante, with only a slight performance decrease for interim models with 2k and 1k marginalization samples. Plot 1(b) shows model performance across training datasets, aggregated across the reserve-price range. Relative performance trends across the five methods are consistent across the six training datasets, suggesting the trends are robust to different m and o combinations. For further discussion, see App. ??.

7.2 Fine-Grained Grid Mixture Results

For further evaluation, we construct a lower-noise dataset with 300 reserve prices ($r \in [0.05, 15]$ in 0.05 steps), more mixed strategies per reserve, and more observations per mixture (see App. ??). Figs. 2 and 3 show deviation-payoff MSEs (95% CIs shaded) for ex ante and interim models trained on $r \leq 8$, with errors computed as before. Performance is aggregated across the six NN models for each method. Fig. 2 shows that errors are now smaller and more uniform across the reserve range, indicating that the learned models are more accurate than the ML test set captured. Relative performance is unchanged: interim models with 5k-10k Monte Carlo samples match ex ante accuracy; all errors are small relative to payoff scale (App. ?? Table ??).

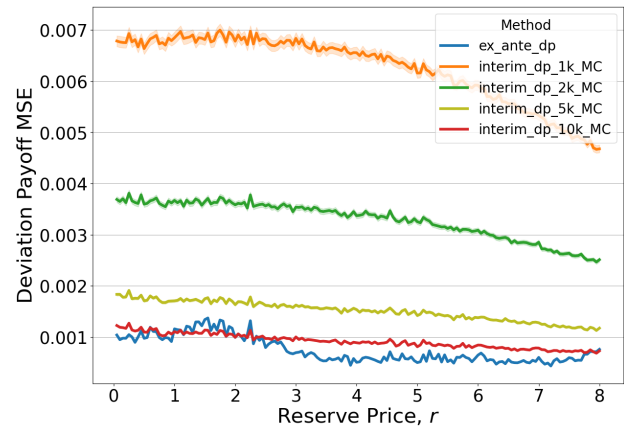


Figure 2: All models perform better on the fine-grained dataset than shown by the noisy ML test set (Fig. 1), but with consistent trends.

As the parameter range covered by the training data must be set prior to any game-theoretic analysis, it may be too narrow, so extrapolation is beneficial for applications like EMD. Fig. 3 shows error across the reserve range, with $r > 8$ representing extrapolation results. All marginalized interim models extrapolate well, while ex ante error rises sharply in extrapolation. These results suggest that interim models learn the relationship between q , θ , r , and deviation payoff: at high reserves, only bidders with $q \cdot \theta \geq r$ can substantively participate, and those few can secure large deviation payoffs by shading their bids. This extrapolation capability is a key advantage of the interim learning approach.

To examine extrapolation performance over different ranges, we train additional neural network models on subsets of the original training data. We train both ex ante and interim models using only samples with $r \leq \bar{r}$, for $\bar{r} \in \{1, \dots, 7\}$ and each of the six initial training datasets (2 functions \times 6 datasets \times 7 cutoffs = 84 total NNs). All models use the same architecture and hyperparameter

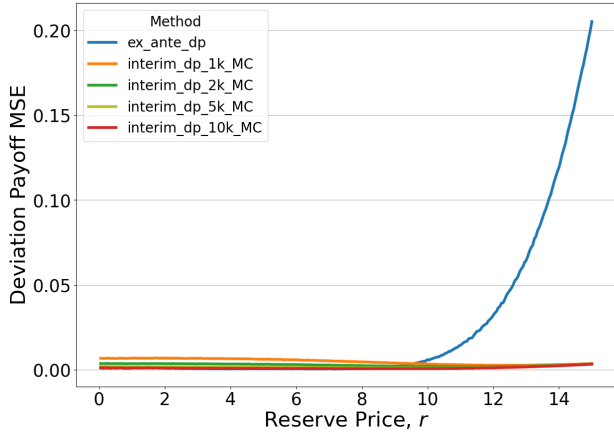


Figure 3: Interim but not ex ante models extrapolate well beyond the trained range, $r \in (0, 8]$.

settings that were identified as optimal for their respective payoff representation under the full training dataset. We evaluate models on the fine-grained test set from Figs. 2 and 3.

Fig. 4 shows performance for models trained on datasets with $\bar{r} \in \{4, \dots, 7\}$. Each curve aggregates results across six neural networks, with confidence intervals shaded. Under training conditions most similar to the original experiment ($\bar{r} \in \{6, 7\}$), interim models continue to extrapolate effectively, whereas ex ante models do not. For $\bar{r} \in \{4, 5\}$, extrapolation performance is comparable between the two methods. This difference is likely affected by the smaller filtered datasets and fixed hyperparameters—each unit decrease in \bar{r} reduces the training dataset size by roughly 12.5% relative to the original training set for which the architecture and hyperparameters were tuned. It is also plausible that the interim method benefits from learning conditional payoffs over a broader range, where the relationship between environment parameter, type, and payoff is more apparent. At lower maximum-reserve cutoffs, both methods exhibit expected degradation in extrapolation performance due to the limited training data. Together with Fig. 3, these results suggest that when trained on ample data across a sufficiently wide parameter range, interim—but not ex ante—models demonstrate robust and consistent extrapolation. App. ?? provides statistics for each filtered dataset, and shows ex ante and interim results for all \bar{r} values.

8 USING THE LEARNED BAYESIAN GAME-FAMILY MODEL

8.1 Deriving Equilibria

To derive equilibria for each of the 300 game instances used in §7.2, we separately run replicator dynamics (RD) [20] with each learned model \hat{u} from fixed initial points. Fig. 5 shows the mean absolute error between predicted and true⁴ regret across all candidate 0.01-BNE returned by RD using each NN (circles) and overall (diamonds) for the trained range (top) and in extrapolation (bottom). Both methods achieve low regret error for candidate equilibria on the

⁴Throughout, “true” payoff or regret refers to a high-fidelity simulated estimate.

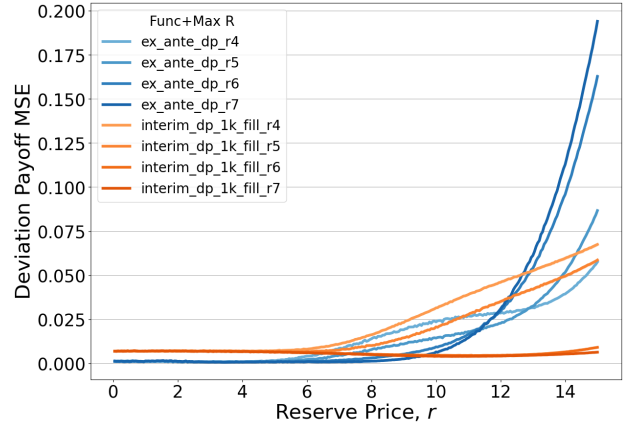


Figure 4: Extrapolation from training over various ranges. For training conditions most similar to those the models were tuned on, only interim models maintain strong extrapolation performance. With smaller datasets and more restricted ranges, ex ante and interim models perform comparably.

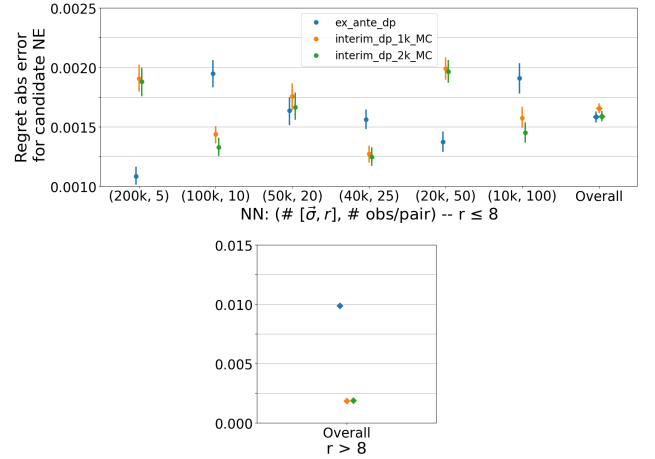


Figure 5: On the trained range ($r \leq 8$), the interim method has regret error comparable to ex ante. In extrapolation, the interim method has lower error than ex ante.

trained range, but the ex ante method is unreliable in extrapolation, with regret error around $\epsilon = 0.01$.

Recall that $\hat{\epsilon}(\vec{\sigma})$ refers to predicted regret and $\epsilon(\vec{\sigma})$ denotes (a high-fidelity estimate of) the true regret. We classify mixed-strategy profiles returned by RD into one of four categories:

Category	Description
Did Not Converge	$\hat{\epsilon}(\vec{\sigma}) > \epsilon$, mixture discarded
Dead Mixture	\hat{u} is $\vec{0}$, killing Nash-finding
Rejected Candidate Mixture	$\hat{\epsilon}(\vec{\sigma}) \leq \epsilon$; $\epsilon(\vec{\sigma}) > \epsilon$
Confirmed Candidate BNE	$\hat{\epsilon}(\vec{\sigma}) \leq \epsilon$; $\epsilon(\vec{\sigma}) \leq \epsilon$

Table 1 reports statistics on the percentage of rejected candidate mixtures, percentage of dead mixtures, and number of holes, separated by game instances within trained range ($r \leq 8$) and in

extrapolation ($r > 8$). A hole occurs when there are no confirmed BNE approximated by the model for a given game instance. Consistent with Fig. 5, both methods perform well on the trained range, with a candidate mixture rejection rate below 3% across all neural network models. In extrapolation, however, between 40% and 50% of mixtures derived by an ex ante model are rejected. Furthermore, for approximately 9% of RD runs with ex ante models, the predicted payoff vector consists entirely of zeros, effectively “killing” the RD update process. Collectively, this results in substantial holes in the expected revenue curve: ex ante revenue curves have *at least* 51 holes, and for 51 game instances (17%) *none* of the six ex ante models identify an equilibrium that is confirmed. Additional details and full RD mixture classification results are provided in App. ??.

Table 1: Summary of RD mixture classification results.

Range	Metric	Ex Ante	Interim (1k MC)
$r \leq 8$	Rejected NE		
	Min	0.51%	0.51%
	Avg	1.58%	1.52%
	Max	2.84%	2.50%
$r > 8$	Rejected NE		
	Min	39.87%	0.00%
	Avg	43.71%	1.85%
	Max	50.00%	4.29%
	Dead Mixture		
	Min	0.71%	0.00%
	Avg	8.69%	0.00%
	Max	15.71%	0.00%
	Num Holes		
	Min	51	0
Avg	69.17	2.17	
Max	77	6	

8.2 Application to Empirical Mechanism Design

Our objective is to identify the reserve price(s) that maximize expected revenue in equilibrium. Game-family learning facilitates the approximation of multiple, distinct BNE, improving the robustness of statistics computed based on the ϵ -BNE set (e.g., worst case, average, max entropy). While the appropriate equilibrium selection method is domain-dependent, we found that relying on the first-returned equilibrium was unreliable, as neighboring game instances sometimes had different equilibria, affecting revenue smoothness.

8.2.1 Grid Optimization. We first conduct a grid search on the average expected revenue curve using the ϵ -BNE sets from §8.1. Fig. 6 shows expected revenue in equilibrium across all confirmed equilibria found by each (a) ex ante and (b) interim model. The overall shape of revenue curves is consistent across models, demonstrating the effectiveness of conducting EMD with a learned game family. For some game instances, the curves diverge: each learned model may derive distinct—yet still valid—equilibria, and small shifts in the set of equilibria derived can translate into noticeable differences in the objective. Training multiple models (on same or

different data) and evaluating the objective on the full set of equilibria derived may address this challenge. Moreover, the presence of multiple equilibria and the possible need for extrapolation suggest that learning the objective function directly would be insufficient.

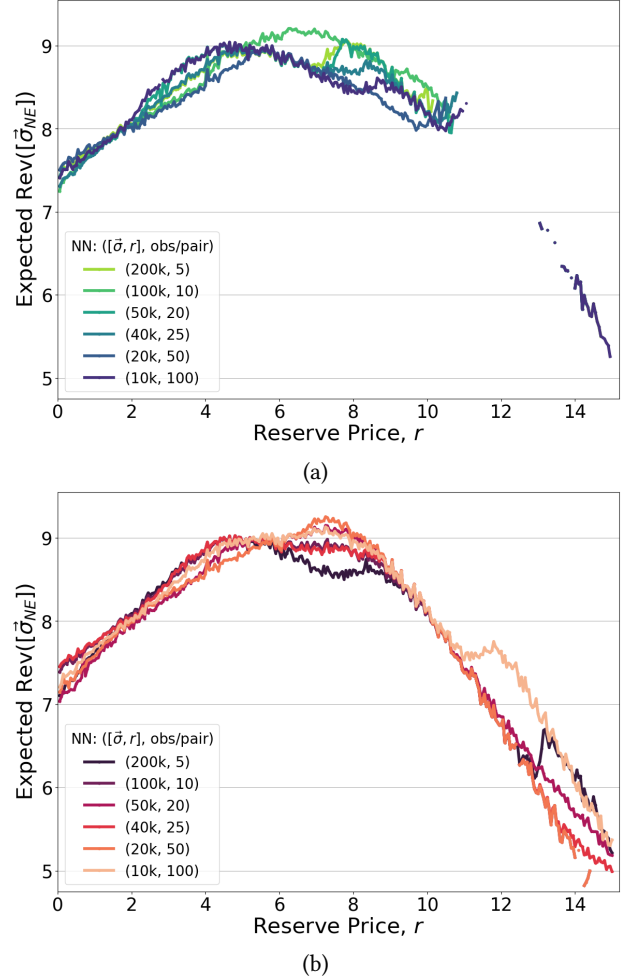


Figure 6: Expected revenue for equilibria derived by (a) ex ante and (b) interim models.

In extrapolation, the ex ante curves exhibit significant gaps where every candidate equilibrium is rejected. Relying solely on an ex ante revenue curve leaves the possibility of a second revenue peak around $r = 11$. In contrast, with any interim model we have confidence that the optimal reserve is below 9, and interim extrapolation is crucial for this confirmation. Although no single optimal reserve price emerges, identifying the optimal revenue plateau (e.g., $6 \leq r \leq 8$) remains valuable as optimal plateaus have been present in past EMD studies [4]. Worst-case expected revenue results are provided in App. ??.

8.2.2 Local Search Optimization. We compare using ex ante and interim models in stochastic hill climbing and simulated annealing local search algorithms (described in App. ??). Stochastic hill

climbing terminates when all uphill neighbors have been explored ($r \pm 0.05$, $r \pm 0.1$, $r \pm 0.25$) or after 50 iterations. Simulated annealing terminates after 50 iterations, but typically evaluates a comparable number of game instances due to initial high temperature causing repeated selections. To evaluate a given game instance with learned model \hat{u} , we run RD with \hat{u} (using 1000 Monte Carlo type samples for interim) starting from the same initial mixtures as in the previous section. We use model \hat{u} to compute predicted regret, and identify candidate equilibria. Expected revenue for a candidate equilibrium $\bar{\sigma}$ is approximated by sampling 100 pure profiles from $\bar{\sigma}$, and computing average revenue across 10,000 auction settings for each pure profile. The expected revenue in equilibrium is the average revenue across all *candidate* equilibria. After search termination, we identify the game instance with the highest revenue in equilibrium, and use true deviation payoff samples to determine the set of confirmed equilibria. The final reported expected revenue in equilibrium is the average revenue over all *confirmed* equilibria.

Fig. 7 shows 95% bootstrap confidence intervals for optimal expected revenue in equilibrium from hill climbing (HC) and simulated annealing (SA) using ex ante (EA) or interim (I) models. For each combination, we perform 50 random r restarts for robust evaluation. As 5 restarts is more practically realistic, we sample 5 (of 50) restart experiments with replacement, and determine the game instance with the max revenue for confirmed equilibria. We repeat this process 100,000 times, and plot 95% confidence intervals around the mean optimal expected revenue for confirmed equilibria. For each model, we also plot the maximum expected revenue based on grid optimization. In nearly all cases, the upper confidence bound aligns with the optimal value from grid optimization. Note that Nash approximation with ex ante models is deterministic whereas Nash approximation with interim is not, which is why the interim confidence interval may extend beyond the max revenue from grid optimization. The decrease in max revenue from grid optimization to average max revenue in local search is less than 2.25%. This is notable considering the granularity ($\delta = 0.05$, $0.05 \leq r \leq 15$) and that local search evaluates only 30-50% of the game instances on average compared to grid optimization (App. ?? Fig. ??). This experiment highlights the effectiveness of local search with learned game families and a modest number of restarts.

8.3 Piecewise-Conditional Strategies

Without additional simulation or training, we use the learned interim model to (1) compute a piecewise best-response strategy to an ε -BNE, and (2) quantify the payoff gain by playing this strategy while opponents play the ε -BNE. We reduce the type space to a single dimension by using $q \cdot \theta$, which represents a deviator’s maximum effective bid. We partition this type space into $|C| = 5$ equiprobable intervals. For a given r and confirmed $\bar{\sigma}_{BNE}$, we use the learned interim model to compute ψ according to Eq. 8 with $|\mathcal{N}| = 100,000$ sampled values of q and θ . The predicted deviation gain from $\bar{\sigma}_{BNE}$ to ψ is $\hat{u}_\psi(\bar{\sigma}_{BNE}, r) - \bar{\sigma}_{BNE} \cdot \hat{u}(\bar{\sigma}, r)$, where \hat{u}_ψ is given by Eq. 9 and \hat{u} is the marginalized interim deviation payoff vector. The corresponding high-fidelity gain, $\tilde{u}_\psi(\bar{\sigma}_{BNE}, r) - \bar{\sigma}_{BNE} \cdot \tilde{u}(\bar{\sigma}_{BNE}, r)$, uses Eq. 10 for \tilde{u}_ψ .

Fig. 8 compares predicted and true gains for playing ψ in response to equilibria in optimal game instances identified via grid

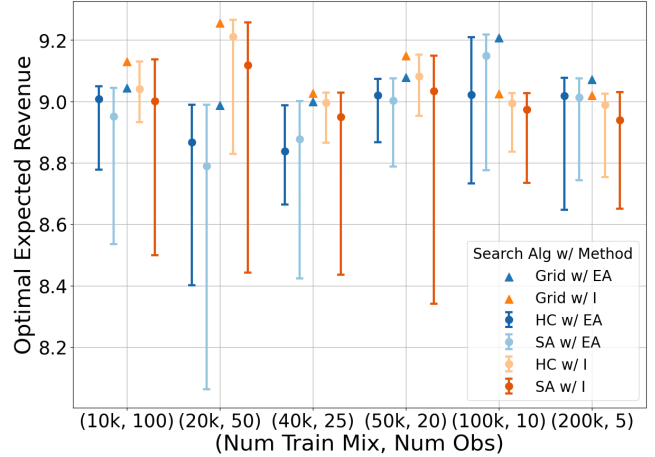


Figure 7: Local search with learned game families and limited restarts identifies high-revenue game instances comparable on average to grid optimization.

optimization. For each r , interim NN, and confirmed ε -BNE, we compute ψ and its corresponding predicted and true payoff gains; each point shows the mean gain across all 6 NNs and the equilibria they derived. The gains from deviation entail a corresponding regret for the profile they are deviating from in a game with the ψ added to S . In other words, the derived BNE candidates have been refuted as 0.01-BNE. Extending the learned Bayesian game family model to include these composite strategies would be needed to derive new equilibria and optimal mechanism parameters in the augmented game family.

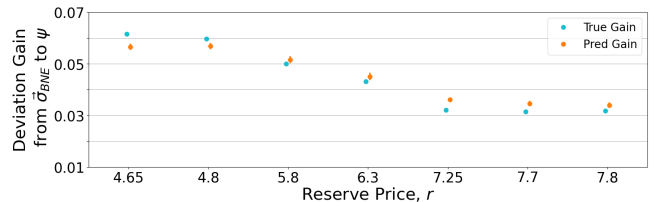


Figure 8: A player can gain payoff larger than $\varepsilon = 0.01$ by playing a piecewise best response (ψ) to an ε -BNE compared to playing the ε -BNE.

9 CONCLUSION

Through an in-depth EMD study of a dynamic sponsored search auction, we demonstrate the advantages of learning an interim model for Bayesian game families. It achieves low payoff error across the trained parameter range and in extrapolation, and is overall more reliable at approximating Bayes-Nash equilibria than the ex ante model. Its extrapolation capability enables confident identification of the optimal reserve range via grid optimization. Finally, the model supports the computation of piecewise best-response strategies without additional sampling, which may effectively guide expansion of the strategy set.

REFERENCES

- [1] Enrique Areyan Viqueira, Cyrus Cousins, and Amy Greenwald. 2020. Improved Algorithms for Learning Equilibria in Simulation-Based Games. In *19th International Conference on Autonomous Agents and Multiagent Systems*. 79–87.
- [2] Enrique Areyan Viqueira, Cyrus Cousins, Yasser Mohammad, and Amy Greenwald. 2020. Empirical Mechanism Design: Designing Mechanisms from Data. In *35th Conference on Uncertainty in Artificial Intelligence*. 1094–1104.
- [3] Martin Bichler, Maximilian Fichtl, Stefan Heidekrüger, Nils Kohring, and Paul Sutterer. 2021. Learning Equilibria in Symmetric Auction Games Using Artificial Neural Networks. *Nature Machine Intelligence* 3, 8 (2021), 687–695.
- [4] Erik Brinkman and Michael P. Wellman. 2017. Empirical Mechanism Design for Optimizing Clearing Interval in Frequent Call Markets. In *18th ACM Conference on Economics and Computation*. 205–221.
- [5] Quang Duong, Yevgeniy Vorobeychik, Satinder Singh, and Michael P. Wellman. 2009. Learning Graphical Game Models. In *21st International Joint Conference on Artificial Intelligence*. 116–121.
- [6] Paul Dütting, Zhe Feng, Harikrishna Narasimhan, David Parkes, and Sai Srivatsa Ravindranath. 2019. Optimal Auctions through Deep Learning. In *36th International Conference on Machine Learning*. PMLR, 1706–1715.
- [7] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords. *American Economic Review* 97, 1 (2007), 242–259.
- [8] Sevan G. Ficici, David C. Parkes, and Avi Pfeffer. 2008. Learning and Solving Many-Player Games through a Cluster-Based Representation. In *24th Conference on Uncertainty in Artificial Intelligence*. 188–195.
- [9] Hu Fu and Tao Lin. 2020. Learning Utilities and Equilibria in Non-Truthful Auctions. In *34th Annual Conference on Neural Information Processing Systems*. 14231–14242.
- [10] Madelyn Gatchel and Bryce Wiedenbeck. 2023. Learning Parameterized Families of Games. In *22nd International Conference on Autonomous Agents and Multiagent Systems*. 1044–1052.
- [11] Ian Gemp, Thomas Anthony, Janos Kramar, Tom Eccles, Andrea Tacchetti, and Yoram Bachrach. 2022. Designing All-Pay Auctions Using Deep Learning and Multi-Agent Simulation. *Scientific Reports* 12, 1 (2022), 16937.
- [12] Patrick R. Jordan, L. Julian Schwartzman, and Michael P. Wellman. 2010. Strategy Exploration in Empirical Games. In *9th International Conference on Autonomous Agents and Multiagent Systems*. 1131–1138.
- [13] Sébastien Lahaie. 2006. An Analysis of Alternative Slot Auction Designs for Sponsored Search. In *7th ACM Conference on Electronic Commerce*. 218–227.
- [14] Sébastien Lahaie, David M. Pennock, Amin Saberi, and Rakesh V. Vohra. 2007. Sponsored Search Auctions. In *Algorithmic Game Theory*, Eva Tardos, Noam Nisan, Tim Roughgarden, and Vijay V. Vazirani (Eds.). Cambridge University Press, 699–716.
- [15] Zun Li and Michael P. Wellman. 2020. Structure Learning for Approximate Solution of Many-Player Games. In *34th AAAI Conference on Artificial Intelligence*. 2119–2127.
- [16] Zun Li and Michael P. Wellman. 2021. Evolution Strategies for Approximate Solution of Bayesian Games. In *35th AAAI Conference on Artificial Intelligence*. 5531–5540.
- [17] Siqi Liu, Luke Marris, Georgios Piliouras, Ian Gemp, and Nicolas Hees. 2023. NfgTransformer: Equivariant Representation Learning for Normal-form Games. In *12th International Conference on Learning Representations*.
- [18] Alberto Marchesi, Francesco Trovò, and Nicola Gatti. 2020. Learning Probably Approximately Correct Maximin Strategies in Simulation-Based Games with Infinite Strategy Spaces. In *19th International Conference on Autonomous Agents and Multiagent Systems*. 834–842.
- [19] Samuel Sokota, Caleb Ho, and Bryce Wiedenbeck. 2019. Learning Deviation Payoffs in Simulation-Based Games. In *33rd AAAI Conference on Artificial Intelligence*. 2173–2180.
- [20] Peter D. Taylor and Leo B. Jonker. 1978. Evolutionary Stable Strategies and Game Dynamics. *Mathematical Biosciences* 40, 1 (1978), 145–156.
- [21] David R. M. Thompson and Kevin Leyton-Brown. 2009. Computational Analysis of Perfect-Information Position Auctions. In *10th ACM Conference on Electronic Commerce*. 51–60.
- [22] Hal R. Varian. 2007. Position Auctions. *International Journal of Industrial Organization* 25, 6 (2007), 1163–1178.
- [23] Yevgeniy Vorobeychik, Christopher Kiekintveld, and Michael P. Wellman. 2006. Empirical Mechanism Design: Methods, with Application to a Supply-Chain Scenario. In *7th ACM Conference on Electronic Commerce*. 306–315.
- [24] Yevgeniy Vorobeychik, Daniel M. Reeves, and Michael P. Wellman. 2012. Constrained Automated Mechanism Design for Infinite Games of Incomplete Information. *Autonomous Agents and Multi-Agent Systems* 25, 2 (2012), 313–351.
- [25] Yevgeniy Vorobeychik, Michael P. Wellman, and Satinder Singh. 2007. Learning Payoff Functions in Infinite Games. *Machine Learning* 67, 1 (2007), 145–168.
- [26] Yongzhao Wang, Tariq Elahi, Vasilios Mavroudis, Edward Plumb, Rahul Savani, and Theodore Turocy. 2025. Empirical Mixnet Design. In *16th Conference on Game Theory and AI for Security*. Springer-Verlag.
- [27] Michael P. Wellman, Karl Tuyls, and Amy Greenwald. 2025. Empirical Game-Theoretic Analysis: A Survey. *Journal of Artificial Intelligence Research* 82 (2025), 1017–1076.
- [28] Bryce Wiedenbeck, Fengjun Yang, and Michael P. Wellman. 2018. A Regression Approach for Modeling Games with Many Symmetric Players. In *32nd AAAI Conference on Artificial Intelligence*. 1266–1273.
- [29] Brian Hu Zhang, Gabriele Farina, Ioannis Anagnostides, Federico Cacciamani, Stephen Marcus McAleer, Andreas Alexander Haupt, Andrea Celli, Nicola Gatti, Vincent Conitzer, and Tuomas Sandholm. 2023. Computing Optimal Equilibria and Mechanisms via Learning in Zero-Sum Extensive-Form Games. In *37th Annual Conference on Neural Information Processing Systems*.
- [30] Brian Hu Zhang and Tuomas Sandholm. 2021. Finding and Certifying (Near-)Optimal Strategies in Black-Box Extensive-Form Games. In *35th AAAI Conference on Artificial Intelligence*. 5779–5788.