Learning Theory and Algorithms for Revenue Optimization in Second-Price Auctions with Reserve

Mehryar Mohri MOHRI@CIMS.NYU.EDU
Courant Institute and Google Research, 251 Mercer Street, New York, NY 10012

Andres Muñoz Medina MUNOZ@CIMS.NYU.EDU
Courant Institute, 251 Mercer Street,New York, NY 10012

Abstract

Second-price auctions with reserve play a critical role in the revenue of modern search engine and popular online sites since the revenue of these companies often directly depends on the outcome of such auctions. The choice of the reserve price is the main mechanism through which the auction revenue can be influenced in these electronic markets. We cast the problem of selecting the reserve price to optimize revenue as a learning problem and present a full theoretical analysis dealing with the complex properties of the corresponding loss function. We further give novel algorithms for solving this problem and report the results of several experiments demonstrating their effectiveness.

1. Introduction

Over the past few years, advertisement has gradually moved away from the traditional printed promotion to the more tailored and directed online publicity. The advantages of online advertisement are clear: since most modern search engine and popular online site companies such as as Microsoft, Facebook, Google, eBay, or Amazon, may collect information about the users' behavior, advertisers can better target the population sector their brand is intended for.

More recently, a new method for selling advertisements has gained momentum. Unlike the standard contracts between publishers and advertisers where some amount of impressions is required to be fulfilled by the publisher, an Ad Exchange works in a way similar to a financial exchange where advertisers bid and compete between each other for an ad slot. The winner then pays the publisher and his ad is displayed.

The design of such auctions and their properties are crucial since they generate a large fraction of the revenue of popular online sites. These questions have motivated extensive research on the topic of auctioning in the last decade or so, particularly in the theoretical computer science and economic theory communities. Much of this work has focused on the analysis of mechanism design, either to prove some useful property of an existing auctioning mechanism, to analyze its computational efficiency, or to search for an optimal revenue maximization truthful mechanism (see (Muthukrishnan, 2009) for a good discussion of key research problems related to Ad Exchange and references to a fast growing literature therein).

One important problem is that of determining an auction mechanism that achieves optimal revenue (Muthukrishnan, 2009). In the ideal scenario where the valuation of the bidders is drawn i.i.d. from a given distribution, this is known to be achievable (see for example (Myerson, 1981)). But, even good approximations of such distributions are not known in practice. Game theoretical approaches to the design of auctions have given a series of interesting results including (Riley & Samuelson, 1981; Milgrom & Weber, 1982; Myerson, 1981; Nisan et al., 2007), all of them based on some assumptions about the distribution of the bidders, e.g., the monotone hazard rate assumption.

The results of the recent publications have nevertheless set the basis for most Ad Exchanges in practice: the mechanism widely adopted for selling ad slots is that of a *Vickrey auction* (Vickrey, 1961) or *second-price auction with reserve price* r (Easley & Kleinberg, 2010). In such auctions, the winning bidder (if any) pays the maximum of the second-place bid and the reserve price r. The reserve price can be set by the publisher or automatically by the exchange. The popularity of these auctions relies on the fact that they are incentive compatible, i.e., bidders bid exactly what they are willing to pay. It is clear that the revenue of the publisher depends greatly on how the reserve price is set: if set too low, the winner of the auction might end up paying only a small amount, even if his bid was really high; on the other hand, if it is set too high, then bidders may not bid higher than the reserve price and the ad slot will not be sold.

Proceedings of the 31^{st} *International Conference on Machine Learning*, Beijing, China, 2014. JMLR: W&CP volume 32. Copyright 2014 by the author(s).

We propose a machine learning approach to the problem of determining the reserve price to optimize revenue in such auctions. The general idea is to leverage the information gained from past auctions to predict a beneficial reserve price. Since every transaction on an Exchange is logged, it is natural to seek to exploit that data. This could be used to estimate the probability distribution of the bidders, which can then be used indirectly to come up with the optimal reserve price (Myerson, 1981; Ostrovsky & Schwarz, 2011). Instead, we will seek a discriminative method making use of the loss function related to the problem and taking advantage of existing user features.

Machine learning methods have already been used for the related problems of designing incentive compatible auction mechanisms (Balcan et al., 2008; Blum et al., 2004), for algorithmic bidding (Langford et al., 2010; Amin et al., 2012), and even for predicting bid landscapes (Cui et al., 2011). But, to our knowledge, no prior work has used historical data in combination with user features for the sole purpose of revenue optimization in this context. In fact, the only publications we are aware of that are directly related to our objective are (Ostrovsky & Schwarz, 2011) and the interesting work of Cesa-Bianchi et al. (2013) which considers a more general case than (Ostrovsky & Schwarz, 2011). The scenario studied by Cesa-Bianchi et al. is that of censored information, which motivates their use of a bandit model to optimize the revenue of the seller. Our analysis assumes instead access to full information. We argue that this is a more realistic scenario since most companies do in fact have access to the full historical data.

The learning scenario we consider is more general since it includes the use of features, as is standard in supervised learning. Since user information is sent to advertisers and bids are made based on this information, it is only natural to include user features in our learning solution. A special case of our analysis coincides with the no-feature scenario considered by Cesa-Bianchi et al. (2013), assuming full information. But, our results further extend those of this paper even in that scenario. In particular, we present an $O(m \log m)$ algorithm for solving a key optimization problem used as a subroutine by the authors, for which they do not seem to give an algorithm. We also do not require an i.i.d. assumption about the bidders, although this is needed in (Cesa-Bianchi et al., 2013) only for the bandit approach.

The theoretical and algorithmic analysis of this learning problem raises several non-trivial technical issues. This is because, unlike some common problems in machine learning, here, the use of a convex surrogate loss cannot be successful. Instead, we must derive an alternative non-convex surrogate requiring novel theoretical guarantees (Section 3) and a new algorithmic solution (Section 4). We present a detailed analysis of possible surrogate losses and select a continuous loss that we prove to be calibrated and for which we give generalization bounds. This leads to an optimization problem cast as a DC-programming problem whose solutions are examined in detail: we first present an efficient combinatorial algorithm for solving that optimization in the no-feature case, next we combine that solution with the DC algorithm (DCA) (Tao & An, 1998) to solve the general case. Section 5 reports the results of our experiments with synthetic data in both the no-feature case and the general case. We first introduce the problem of selecting the reserve price to optimize revenue and cast it as a learning problem (Section 2).

2. Reserve price selection problem

As already discussed, the choice of the reserve price r is the main mechanism through which a seller can influence the auction revenue. To specify the results of a second-price auction we need only the vector of first and second highest bids which we denote by $\mathbf{b} = (b^{(1)}, b^{(2)}) \in \mathcal{B} \subset \mathbb{R}_+^2$. For a given reserve price r and bid pair $\mathbf{b}$, the revenue of an auction is given by

$$\text{Revenue}(r, \mathbf{b}) = b^{(2)} \mathbb{1}_{r < b^{(2)}} + r \mathbb{1}_{b^{(2)} \leq r \leq b^{(1)}}. \quad (1)$$

The simplest setup is one where there are no features associated with the auction. In that case, the objective is to select r to optimize the expected revenue, which can be expressed as follows (see Appendix A.1):

$$\mathop{\mathbb{E}}_{\mathbf{b}}[\text{Revenue}(r, \mathbf{b})] = \int_r^{\infty} \mathbb{P}[b^{(2)} > t] dt + r\, \mathbb{P}[b^{(1)} \geq r]. \quad (2)$$

A similar derivation is given by Cesa-Bianchi et al. (2013). In fact, this expression is precisely the one optimized by these authors. If we now associate with each auction a feature vector $\mathbf{x} \in \mathcal{X}$, the so-called *public information*, and set the reserve price to $h(\mathbf{x})$, where $h \colon \mathcal{X} \to \mathbb{R}_+$ is our reserve price hypothesis function, the problem can be formulated as that of selecting out of some hypothesis set H a hypothesis h with large expected revenue:

$$\mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{b}) \sim D}[\text{Revenue}(h(\mathbf{x}), \mathbf{b})], \quad (3)$$

where D is the unknown distribution according to which the pairs $(\mathbf{x}, \mathbf{b})$ are drawn. Instead of the revenue, we will consider a loss function L defined for all $(r, \mathbf{b})$ by $L(r, \mathbf{b}) = -\text{Revenue}(r, \mathbf{b})$, and will seek a hypothesis h with small expected loss $\mathcal{L}(h) := \mathbb{E}_{(\mathbf{x},\mathbf{b}) \sim D}[L(h(\mathbf{x}), \mathbf{b})]$. As in standard supervised learning scenarios, we assume access to a training sample $S = ((\mathbf{x}_1, \mathbf{b}_1), \ldots, (\mathbf{x}_m, \mathbf{b}_m))$ of size $m \geq 1$ drawn i.i.d. according to D and denote by $\widehat{\mathcal{L}}_S(h)$ the empirical loss $\frac{1}{m} \sum_{i=1}^m L(h(\mathbf{x}_i, \mathbf{b}_i)$. In the next sections, we present a detailed study of this learning problem.

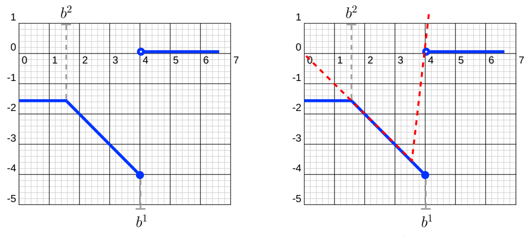


Figure 1. (a) Plot of the loss function $r \mapsto L(r, \mathbf{b})$ for fixed values of $b^{(1)}$ and $b^{(2)}$; (b) piecewise linear convex surrogate loss.

3. Learning guarantees

To derive generalization bounds for the learning problem formulated in the previous section, we need to analyze the complexity of the family of functions L_H mapping $\mathcal{X} \times \mathcal{B}$ to $\mathbb{R}$ defined by $L_H = \{(\mathbf{x}, \mathbf{b}) \mapsto L(h(\mathbf{x}), \mathbf{b}) \colon h \in H\}$. The loss function L is neither Lipschitz continuous nor convex (see Figure 1). To analyze its complexity, we decompose L as a sum of two loss functions l_1 and l_2 with more convenient properties. We have $L = l_1 + l_2$ with l_1 and l_2 defined for all $(\mathbf{x}, \mathbf{b}) \in \mathcal{X} \times \mathcal{B}$ by

$$l_1(r, \mathbf{b}) = -b^{(2)} \mathbb{1}_{r<b^{(2)}} - r \mathbb{1}_{b^{(2)} \leq r \leq b^{(1)}} - b^{(1)} \mathbb{1}_{r>b^{(1)}}$$
$$l_2(r, \mathbf{b}) = b^{(1)} \mathbb{1}_{r>b^{(1)}}.$$

Note that for a fixed $\mathbf{b}$, the function $r \mapsto l_1(r, \mathbf{b})$ is 1-Lipschitz since the slope of the lines defining the function is at most 1. We will consider the corresponding family of loss functions: $l_{1H} = \{(\mathbf{x}, \mathbf{b}) \mapsto l_1(h(\mathbf{x}), \mathbf{b}) \colon h \in H\}$ and $l_{2H} = \{(\mathbf{x}, \mathbf{b}) \mapsto l_2(h(\mathbf{x}), \mathbf{b}) \colon h \in H\}$ and use the notions of pseudo-dimension as well as empirical and average Rademacher complexity. The pseudo-dimension is a standard complexity measure (Pollard, 1984) extending the notion of VC-dimension to real-valued functions (see also (Mohri et al., 2012)). For a family of functions G and finite sample $S = (z_1, \ldots, z_m)$ of size m, the empirical Rademacher complexity is defined by $\widehat{\mathfrak{R}}_S(G) = \mathbb{E}_{\boldsymbol{\sigma}} \left[\sup_{g \in G} \frac{1}{m} \sum_{i=1}^m \sigma_i g(z_i)\right]$, where $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_m)^\top$, with σ_is independent uniform random variables taking values in $\{-1, +1\}$. The Rademacher complexity of G is defined as $\mathfrak{R}_m(G) = \mathbb{E}_{S \sim D^m}[\widehat{\mathfrak{R}}_S(G)]$.

Theorem 1. *Let $M = \sup_{\mathbf{b} \in \mathcal{B}} b^{(1)}$ and let H be a hypothesis set with pseudo-dimension $d = Pdim(H)$. Then, for any $\delta > 0$, with probability at least $1 - \delta$ over the choice of a sample S of size m, the following inequality holds for all $h \in H$:*

$$\mathcal{L}(h) \leq \widehat{\mathcal{L}}_S(h) + 2\mathfrak{R}_m(H) + 2M\sqrt{\frac{2d \log \frac{em}{d}}{m}} + M\sqrt{\frac{\log \frac{1}{\delta}}{2m}}.$$

Proof. By a standard property of the Rademacher complexity, since $L = l_1 + l_2$, the following inequality holds: $\mathfrak{R}_m(L_H) \leq \mathfrak{R}_m(l_{1H}) + \mathfrak{R}_m(l_{2H})$. Thus, in view of Propositions 9 and 10, the Rademacher complexity of L_H can be bounded via

$$\mathfrak{R}_m(L_H) \leq \mathfrak{R}_m(H) + M\sqrt{\frac{2d \log \frac{em}{d}}{m}}.$$

The result then follows by the application of a standard Rademacher complexity bound (Koltchinskii & Panchenko, 2002). □

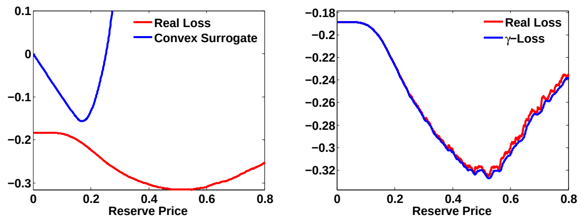


Figure 2. Comparison of the sum of real losses $\sum_{i=1}^m L(\cdot, \mathbf{b}_i)$ for $m = 500$ versus two different surrogates. (a) Sum of convex surrogate losses: the minimizer significantly differs from that of the sum of the original losses. (b) The surrogate loss sum $\sum_{i=1}^m L_\gamma(\cdot, \mathbf{b}_i)$ for $\gamma = .02$

This learning bound invites us to consider an algorithm seeking $h \in H$ to minimize the empirical loss $\widehat{\mathcal{L}}_S(h)$, while controlling the complexity (Rademacher complexity and pseudo-dimension) of the hypothesis set H. However, as in the familiar case of binary classification, in general, minimizing this empirical loss is a computationally hard problem. Thus, in the next section, we study the question of using a surrogate loss instead of the original loss L.

3.1. Surrogate loss

As pointed out earlier, the loss function L does not admit some common useful properties: for any fixed $\mathbf{b}$, $L(\cdot, \mathbf{b})$ is not differentiable at two points, is not convex, and is not Lipschitz, in fact it is discontinuous. For any fixed $\mathbf{b}$, $L(\cdot, \mathbf{b})$ is quasi-convex, a property that is often desirable since there exist several solutions for quasi-convex optimization problems. However, in general, a sum of quasi-convex functions, such as the sum $\sum_{i=1}^m L(\cdot, \mathbf{b}_i)$ appearing in the definition of the empirical loss, is not quasi-convex and a fortiori not convex.[1] In fact, in general, such a sum may admit exponentially many local minima. This leads us to seek a surrogate loss function with more favorable optimization properties.

A standard method in machine learning consists of replacing the loss function L with a convex upper bound (Bartlett et al., 2006). A natural candidate in our case is the piecewise linear convex function shown in Figure 1(b). However, while this convex loss function is convenient for optimization, it is not calibrated and does not provide a useful

[1]It is known that under some separability condition if a finite sum of quasi-convex functions on an open convex set is quasi-convex then all but perhaps one of them is convex (Debreu & Koopmans, 1982).

surrogate. The calibration problem is illustrated by Figure 2(a) in dimension one, where the true objective function to be minimized $\sum_{i=1}^{m} L(r, \mathbf{b}_i)$ is compared with the sum of the surrogate losses. The next theorem shows that this problem affects in fact any non-constant convex surrogate. It is expressed in terms of the loss $\widetilde{L}\colon \mathbb{R} \times \mathbb{R}_+ \to \mathbb{R}$ defined by $\widetilde{L}(r, b) = -r 1\!\!1_{r \le b}$, which coincides with L when the second bid is 0.

Theorem 2 (convex surrogates)**.** *There exists no non-constant function $L_c\colon \mathbb{R} \times \mathbb{R}_+ \to \mathbb{R}$ convex with respect to its first argument and satisfying the following conditions:*

- *for any $b_0 \in \mathbb{R}_+$, $\lim_{b \to b_0^-} L_c(b_0, b) = L_c(b_0, b_0)$.*
- *for any distribution D on $\mathbb{R}_+$, there exists a non-negative minimizer $r^* \in \operatorname{argmin}_r \mathbb{E}_{b \sim D}[\widetilde{L}(r, b)]$ such that $\min_r \mathbb{E}_{b \sim D} L_c(r, b) = \mathbb{E}_{b \sim D} L_c(r^*, b)$.*

This theorem, proven in Appendix A.4, leads us to consider alternative non-convex loss functions. Perhaps, the most natural surrogate loss is then L'_γ, an upper bound on L defined for all $\gamma > 0$ by:

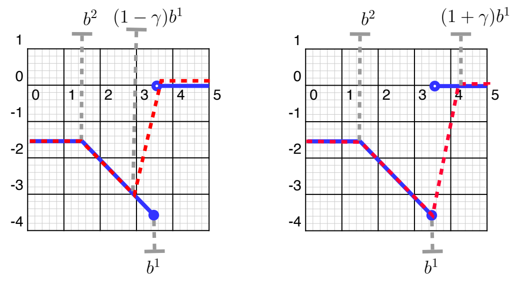
$$L'_\gamma(r, \mathbf{b}) = -b^{(2)} 1\!\!1_{r \le b^{(2)}} - r 1\!\!1_{b^{(2)} < r \le \left((1-\gamma) b^{(1)}\right) \vee b^{(2)}}$$
$$+ \left(\frac{1-\gamma}{\gamma} \vee \frac{b^{(2)}}{b^{(1)} - b^{(2)}}\right)(r - b^{(1)}) 1\!\!1_{\left((1-\gamma)b^{(1)}\right) \vee b^{(2)}) < r \le b^{(1)}},$$

where $c \vee d = \max(c, d)$. The plot of this function is shown in Figure 3(a). The max terms ensure that the function is well defined if $(1-\gamma) b^{(1)} < b^{(2)}$. However, this turns out to be also a poor choice because L'_γ is a loose upper bound of L in the most critical region, that is around the minimum of the loss L. Thus, instead, we will consider, for any $\gamma > 0$, the loss function L_γ defined as follows:

$$L_\gamma(r, \mathbf{b}) = -b^{(2)} 1\!\!1_{r \le b^{(2)}} - r 1\!\!1_{b^{(2)} < r \le b^{(1)}} + \frac{1}{\gamma}(r - (1+\gamma) b^{(1)}) 1\!\!1_{b^{(1)} < r \le (1+\gamma) b^{(1)}}, \tag{4}$$

and shown in Figure 3(b).[2] A comparison between the sum of L-losses and the sum of L_γ-losses is shown in Figure 2(b). Observe that the fit is considerably better than when using a piecewise linear convex surrogate loss. A possible concern associated with the loss function L_γ is that it is a lower bound for L. One might think then that minimizing it would not lead to an informative solution. However, we argue that this problem arises significantly with upper bounding losses such as the convex surrogate, which we showed not to lead to a useful minimizer, or L'_γ, which is a poor approximation of L near its minimum. By matching the original loss L in the region of interest,

[2]Technically, the theoretical and algorithmic results we present for L_γ could be developed in a somewhat similar way for L'_γ.



(a) (b)

Figure 3. Comparison of the true loss L with (a) the surrogate loss L'_γ; (b) the surrogate loss L_γ, for $\gamma = 0.1$.

around the minimal value, the loss function L_γ leads to more informative solutions in this problem. We further analyze the difference of the expectations of L and L_γ and show that L_γ is calibrated. We will use for any $h \in H$, the notation $\mathcal{L}_\gamma(h) := \mathbb{E}_{(\mathbf{x},\mathbf{b}) \sim D}[L_\gamma(h(\mathbf{x}), \mathbf{b})]$.

Theorem 3. *Let H be a closed, convex subset of a linear space of functions containing 0. Denote by h^*_γ the solution of $\min_{h \in H} \mathcal{L}_\gamma(h)$. If $\sup_{\mathbf{b} \in \mathcal{B}} b^{(1)} = M < \infty$, then*

$$\mathcal{L}(h^*_\gamma) - \mathcal{L}_\gamma(h^*_\gamma) \le \gamma M.$$

The following sets, which will be used in our proof, form a partition of $\mathcal{X} \times \mathcal{B}$

$$I_1 = \{(\mathbf{x}, \mathbf{b}) | h^*_\gamma(\mathbf{x}) \le b^{(2)}\}$$
$$I_2 = \{(\mathbf{x}, \mathbf{b}) | h^*_\gamma(\mathbf{x}) \in (b^{(2)}, b^{(1)}]\}$$
$$I_3 = \{(\mathbf{x}, \mathbf{b}) | h^*_\gamma(\mathbf{x}) \in (b^{(1)}, (1+\gamma) b^{(1)}]\}$$
$$I_4 = \{(\mathbf{x}, \mathbf{b}) | h^*_\gamma(\mathbf{x}) > (1+\gamma) b^{(1)}\}$$

This sets represent the different regions where L_γ is defined. In each region the function is affine. We will now state a technical lemma that will help us prove Theorem 3. The proof of this lemma is given in Appendix A.4.

Lemma 4. *Under the conditions of Theorem 3,*

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}} \left[h^*_\gamma(\mathbf{x}) 1\!\!1_{I_2}(\mathbf{x})\right] \ge \frac{1}{\gamma} \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}} \left[h^*_\gamma(\mathbf{x}) 1\!\!1_{I_3}(\mathbf{x})\right].$$

Proof. Of Theorem 3. We can express the difference as

$$\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}} \left[L(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})\right] \tag{5}$$
$$= \sum_{k=1}^{4} \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}} \left[(L(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})) 1\!\!1_{I_k}(\mathbf{x})\right]$$
$$= \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}} \left[(L(h^*_\gamma(\mathbf{x}), \mathbf{b}) - L_\gamma(h^*_\gamma(\mathbf{x}), \mathbf{b})) 1\!\!1_{I_3}(\mathbf{x})\right]$$
$$= \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}} \left[\frac{1}{\gamma}((1+\gamma) b^{(1)} - h^*_\gamma(\mathbf{x})) 1\!\!1_{I_3}(\mathbf{x}))\right]. \tag{6}$$

Furthermore, for $(\mathbf{x}, \mathbf{b}) \in I_3$, we know that $b^{(1)} < h^*_\gamma(\mathbf{x})$. Thus, we can bound (6) by $\mathbb{E}_{\mathbf{x},\mathbf{b}}[h^*_\gamma(\mathbf{x}) 1\!\!1_{I_3}(\mathbf{x})]$, which, by

Lemma 4, is less than $\gamma\, \mathbb{E}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_2}(\mathbf{x})\right]$. Thus, we can write

$$\begin{aligned}&\mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L(h^*_\gamma(\mathbf{x}),\mathbf{b})\right] - \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[L_\gamma(h^*_\gamma(\mathbf{x}),\mathbf{b})\right]\\ &\leq \gamma \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[h^*_\gamma(\mathbf{x})\mathbb{1}_{I_2}(\mathbf{x})\right] \leq \gamma \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{b}}\left[b^{(1)}\mathbb{1}_{I_2}(\mathbf{x})\right] \leq \gamma M,\end{aligned}$$

since $h^*_\gamma(\mathbf{x}) \leq b^{(1)}$ for $(\mathbf{x},\mathbf{b}) \in I_2$. □

Notice that, since $L \geq L_\gamma$ for all $\gamma \geq 0$, it follows easily from the proposition that $\mathcal{L}_\gamma(h^*_\gamma) \to \mathcal{L}(h^*)$. Indeed, if h^* is the best hypothesis in class for the real loss, then the following inequalities are straightforward:

$$\begin{aligned}0 \leq \mathcal{L}_\gamma(h^*) - \mathcal{L}_\gamma(h^*_\gamma) &\leq \mathcal{L}(h^*) - \mathcal{L}_\gamma(h^*_\gamma)\\ &\leq \mathcal{L}(h^*_\gamma) - \mathcal{L}_\gamma(h^*_\gamma) \leq \gamma M\end{aligned}$$

The $1/\gamma$-Lipschitzness of L_γ can be used to prove the following generalization bound (see Appendix A.5).

Theorem 5. *Fix $\gamma \in (0,1]$ and let S denotes a sample of size m. Then, for any $\delta > 0$, with probability at least $1-\delta$ over the choice of the sample S, for all $h \in H$, the following holds:*

$$\mathcal{L}_\gamma(h) \leq \widehat{\mathcal{L}}_\gamma(h) + \frac{2}{\gamma}\mathfrak{R}_m(H) + M\sqrt{\frac{\log\frac{1}{\delta}}{2m}}. \tag{7}$$

The theorem can be used to derive a learning bound that holds uniformly for all $\gamma \in (0,1]$, at the price of an additional term of the form $O(\sqrt{\log\log(1/\gamma)/m})$. These results are reminiscent of the standard margin bounds with γ playing the role of a margin. The situation here is however somewhat different. Our learning bounds suggest, for a fixed $\gamma \in (0,1]$, to seek a hypothesis h minimizing the empirical loss $\widehat{\mathcal{L}}_\gamma(h)$ while controlling a complexity term upper bounding $\mathfrak{R}_m(H)$, which in the case of a family of linear hypotheses could be $\|h\|^2_K$ for some PSD kernel K. Since the bound can hold uniformly for all γ, we can use it to select γ out of a finite set of possible grid search values. Alternatively, γ can be set via cross-validation.

4. Algorithms

In this section we present algorithms for solving the optimization problem for selecting the reserve price. We start with the no-feature case and then treat the general case.

4.1. No feature case

We present a general algorithm to optimize sums of functions similar to L_γ or L in the one-dimensional case.

Definition 6. *We will say that function $V\colon \mathbb{R}\times\mathcal{B} \to \mathbb{R}$ is a* v-function *if it admits the following form:*

$$\begin{aligned}V(r,\mathbf{b}) = -a^{(1)}\mathbb{1}_{r\leq b^{(2)}} - a^{(2)}r\mathbb{1}_{b^{(2)}<r\leq b^{(1)}} +&\\ (a^{(3)}r - a^{(4)})\mathbb{1}_{b^{(1)}<r<(1+\eta)b^{(1)}},&\end{aligned}$$

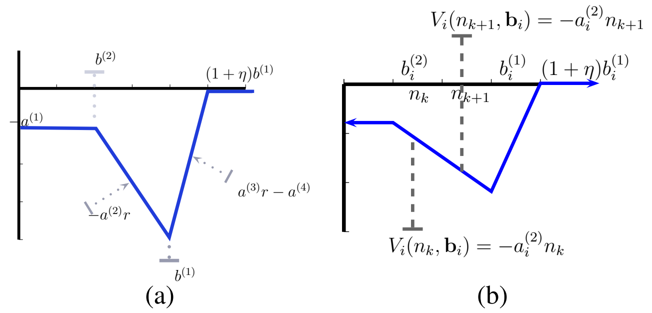


Figure 4. (a) Prototypical v-function. (b) Illustration of the fact that the definition of $V_i(r,\mathbf{b}_i)$ does not change on an interval $[n_k, n_{k+1}]$.

with $a^{(1)} > 0$ and $\eta > 0$ constants and $a^{(1)}, a^{(2)}, a^{(3)}, a^{(4)}$ defined by $a^{(1)} = \eta a^{(3)}b^{(2)}$, $a^{(2)} = \eta a^{(3)}$, and $a^{(4)} = a^{(3)}(1+\eta)b^{(1)}$.

Figure 4(a) illustrates this family of loss functions. A v-function is a generalization of L_γ and L. Indeed, any v-function V satisfies $V(r,\mathbf{b}) \leq 0$ and attains its minimum at $b^{(1)}$. Finally, as can be seen straightforwardly from Figure 3, L_γ is a v-function for any $\gamma \geq 0$. We consider the following general problem of minimizing a sum of v-functions:

$$\min_{r\geq 0} F(r) := \sum_{i=1}^{m} V_i(r,\mathbf{b}_i). \tag{8}$$

Observe that this is not a trivial problem since, for any fixed $\mathbf{b}_i$, $V_i(\cdot,\mathbf{b}_i)$ is non-convex and that, in general, a sum of m such functions may admit many local minima. The following proposition shows that the minimum is attained at one of the highest bids, which matches the intuition.

Proposition 7. *Problem (8) admits a solution r^* that satisfies $r^* = b_i^{(1)}$ for some $i \in [1,m]$.*

The problem can thus be reduced to examining the value of the function for the m arguments $b_i^{(1)}$, $i \in [1,m]$. This yields a straightforward method for solving the optimization which consists of computing $F(b_i^{(1)})$ for all i and taking the minimum. But, since the computation of each $F(b_i^{(1)})$ takes $O(m)$, the overall computational cost is in $O(m^2)$, which can be prohibitive for even moderately large values of m.

Instead, we have devised a more efficient combinatorial algorithm that can be used to solve the problem in $O(m\log m)$ time. The algorithm consists of first sorting all *boundary points*, that is the points in $\mathcal{N} = \bigcup_i\{b_i^{(1)}, b_i^{(2)}, (1+\eta)b_i^{(1)}\}$ associated with the functions $V_i(\cdot,\mathbf{b}_i)$, $i \in [1,m]$. We then show that for the ordered sequence $(n_1,\ldots,n_{3m})$, $F(n_{k+1})$ can be computed from $F(n_k)$ in constant time, using the fact that the definition of $V_i(\cdot, \mathbf{b}_i)$ can only change at boundary points (see Figure 4(b)). A more detailed description and the proof of the correctness of the algorithm are given in the Appendix B. Furthermore, the algorithm can be straightforwardly extended to solve the minimization of F over a set of r-values bounded by Λ, that is $\{r\colon 0 \leq r \leq \Lambda\}$. Indeed, we then only need to compute $F(b_i^{(1)})$ for $i \in [1, m]$ such that $b_i^{(1)} < \Lambda$ and of course also $F(\Lambda)$, thus the computational complexity in that regularized case remains $O(m \log m)$.

4.2. General case

We first consider the case of a hypothesis set H of linear functions $\mathbf{x} \mapsto \mathbf{w} \cdot \mathbf{x}$ with bounded norm, $\|\mathbf{w}\| \leq \Lambda$, for some $\Lambda \geq 0$. This can be immediately generalized to the case where a positive definite kernel is used.

The results of Theorem 5 suggest seeking, for a fixed $\gamma \geq 0$, the vector $\mathbf{w}$ solution of the following optimization problem: $\min_{\|\mathbf{w}\| \leq \Lambda} \sum_{i=1}^m L_\gamma(\mathbf{w} \cdot \mathbf{x}_i, \mathbf{b}_i)$. Replacing the original loss L with L_γ helped us remove the discontinuity of the loss. But, we still face an optimization problem based on a sum of non-convex functions. This problem can be formulated as a DC-programming (difference of convex functions programming) problem. Indeed, L_γ can be decomposed as follows for all $(r, \mathbf{b}) \in \mathcal{X} \times \mathcal{B}$: $L_\gamma(r, \mathbf{b}) = u(r, \mathbf{b}) - v(r, \mathbf{b})$, with the convex functions u and v defined by

$$u(r, \mathbf{b}) = -r \mathbb{1}_{r < b^{(1)}} + \tfrac{r - (1+\gamma) b^{(1)}}{\gamma} \mathbb{1}_{r \geq b^{(1)}}$$
$$v(r, \mathbf{b}) = (-r + b^{(2)}) \mathbb{1}_{r < b^{(2)}} + \tfrac{r - (1+\gamma) b^{(1)}}{\gamma} \mathbb{1}_{r > b^{(1)}}.$$

Using the decomposition $L_\gamma = u - v$, our optimization problem can be formulated as follows:

$$\min_{\mathbf{w} \in \mathbb{R}^N} U(\mathbf{w}) - V(\mathbf{w}) \qquad \text{subject to } \|\mathbf{w}\| \leq \Lambda, \tag{9}$$

where $U(\mathbf{w}) = \sum_{i=1}^m u(\mathbf{w} \cdot \mathbf{x}_i, \mathbf{b}_i)$ and $V(\mathbf{w}) = \sum_{i=1}^m v(\mathbf{w} \cdot \mathbf{x}_i, \mathbf{b}_i)$, which shows that it can be formulated as a DC-programming problem. The global minimum of the optimization problem (9) can be found using a cutting plane method (Horst & Thoai, 1999), but that method only converges in the limit and does not admit known algorithmic convergence guarantees.[3] There exists also a branch-and-bound algorithm with exponential convergence for DC-programming (Horst & Thoai, 1999) for finding the global minimum. Nevertheless, in (Tao & An, 1997), it is pointed out that this type of combinatorial algorithms fail to solve real-world DC-programs in high dimensions. In fact, our implementation of this algorithm shows that the convergence of the algorithm in practice is extremely slow for even moderately high-dimensional problems. Another attractive solution for finding the global solution of a DC-programming problem over a polyhedral convex set is the combinatorial solution of Hoang Tuy (Tuy, 1964). However, casting our problem as an instance of that problem requires explicitly specifying the slope and offsets for the piecewise linear function corresponding to a sum of L_γ losses, which admits an exponential cost in time and space.

An alternative consists of using the DC algorithm, a primal-dual sub-differential method of Dinh Tao and Hoai An (Tao & An, 1998), (see also (Tao & An, 1997) for a good survey). This algorithm is applicable when u and v are proper lower semi-continuous convex functions as in our case. When v is differentiable, the DC algorithm coincides with the CCCP algorithm of Yuille and Rangarajan (Yuille & Rangarajan, 2003), which has been used in several contexts in machine learning and analyzed by (Sriperumbudur & Lanckriet, 2012).

The general proof of convergence of the DC algorithm was given by (Tao & An, 1998). In some special cases, the DC algorithm can be used to find the global minimum of the problem as in the trust region problem (Tao & An, 1998), but, in general, the DC algorithm or its special case CCCP are only guaranteed to converge to a critical point (Tao & An, 1998; Sriperumbudur & Lanckriet, 2012). Nevertheless, the number of iterations of the DC algorithm is relatively small. Its convergence has been shown to be in fact linear for DC-programming problems such as ours (Yen et al., 2012). The algorithm we are proposing goes one step further than that of (Tao & An, 1998): we use DCA to find a local minimum but then restart our algorithm with a new seed that is guaranteed to reduce the objective function. Unfortunately, we are not in the same regime as in the trust region problem of Dinh Tao and Hoai An (Tao & An, 1998) where the number of local minima is linear in the size of the input. Indeed, here the number of local minima can be exponential in the number of dimensions of the feature space and it is not clear to us how the combinatorial structure of the problem could help us rule out some local minima faster and make the optimization more tractable.

In the following, we describe more in detail the solution we propose for solving the DC-programming problem (9). The functions v and V are not differentiable in our context but they admit a sub-gradient at all points. We will denote by $\delta V(\mathbf{w})$ an arbitrary element of the sub-gradient $\partial V(\mathbf{w})$, which coincides with $\nabla V(\mathbf{w})$ at points $\mathbf{w}$ where V is differentiable. The DC algorithm then coincides with CCCP, modulo the replacement of the gradient of V by $\delta V(\mathbf{w})$. It consists of starting with a weight vector $\mathbf{w}_0 \leq \Lambda$ and of iteratively solving a sequence of convex optimization problems obtained by replacing V with its linear approximation giving $\mathbf{w}_t$ as a function of $\mathbf{w}_{t-1}$, for $t = 1, \ldots, T$: $\mathbf{w}_t \in \operatorname{argmin}_{\|\mathbf{w}\| \leq \Lambda} U(\mathbf{w}) - \delta V(\mathbf{w}_{t-1}) \cdot \mathbf{w}$. This problem

[3] Some claims of (Horst & Thoai, 1999), e.g., Proposition 4.4 used in support of the cutting plane algorithm, are incorrect (Tuy, 2002).

DC Algorithm

```
$\mathbf{w} \leftarrow \mathbf{w}_0$                ▷ initialization
for $t \geq 1$ do
    $\mathbf{w}_t \leftarrow$ DCA($\mathbf{w}$)    ▷ DCA algorithm
    $\mathbf{w} \leftarrow$ OPTIMIZE(objective, fixed direction $\mathbf{w}_t/\|\mathbf{w}_t\|$)
end for
```

Figure 5. Pseudocode of our DC-programming algorithm.

can be rewritten in our context as the following:

$$\min_{\|\mathbf{w}\| \leq \Lambda, \mathbf{s}} \sum_{i=1}^{m} s_i - \delta V(\mathbf{w}_{t-1}) \cdot \mathbf{w} \tag{10}$$

subject to $(s_i \geq -\mathbf{w} \cdot \mathbf{x}_i) \wedge \Big[s_i \geq \frac{1}{\gamma}\big(\mathbf{w} \cdot \mathbf{x}_i - (1+\gamma) b_i^{(1)}\big)\Big]$.

The problem is equivalent to a QP (quadratic-programming) problem since the quadratic constraint can be replaced by a term of the form $\lambda\|\mathbf{w}\|^2$ in the objective and thus can be tackled using any standard QP solver. We propose an algorithm that iterates along different local minima, but with the guarantee of reducing the function at every change of local minimum. The algorithm is simple and is based on the observation that the function L_γ is positive homogeneous. Indeed, for any $\eta > 0$ and $(r, \mathbf{b})$,

$$L_\gamma(\eta r, \eta \mathbf{b}) = -\eta b^{(2)} 1_{\eta r < \eta b^{(2)}} - \eta r 1_{\eta b^{(2)} \leq \eta r \leq \eta b^{(1)}} + \frac{\eta r - (1+\gamma)\eta b^{(1)}}{\gamma} 1_{\eta b^{(1)} < \eta r < \eta(1+\gamma) b^{(1)}} = \eta L_\gamma(r, \mathbf{b}).$$

Minimizing the objective function of (9) in a fixed direction $\mathbf{u}$, $\|\mathbf{u}\| = 1$, can be reformulated as follows: $\min_{0 \leq \eta \leq \Lambda} \sum_{i=1}^m L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i)$. Since for $\mathbf{u} \cdot \mathbf{x}_i \leq 0$ the function $\eta \mapsto L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i)$ is constant equal to $-b_i^{(2)}$ the problem is equivalent to solving

$$\min_{0 \leq \eta \leq \Lambda} \sum_{\mathbf{u} \cdot \mathbf{x}_i > 0} L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i).$$

Furthermore, since L_γ is positive homogeneous, for all $i \in [1, m]$ with $\mathbf{u} \cdot \mathbf{x}_i > 0$, $L_\gamma(\eta \mathbf{u} \cdot \mathbf{x}_i, \mathbf{b}_i) = (\mathbf{u} \cdot \mathbf{x}_i) L_\gamma(\eta, \mathbf{b}_i/(\mathbf{u} \cdot \mathbf{x}_i))$. But $\eta \mapsto (\mathbf{u} \cdot \mathbf{x}_i) L_\gamma(\eta, \mathbf{b}_i/(\mathbf{u} \cdot \mathbf{x}_i))$ is a v-function and thus the problem can efficiently optimized using the combinatorial algorithm for the no-feature case (Section 4.1). This leads to the optimization algorithm described in Figure 5. The last step of each iteration of our algorithm can be viewed as a *line search* and this is in fact the step that reduces the objective function the most in practice. This is because we are then precisely minimizing the objective function even though this is for some fixed direction. Since in general this line search does not find a local minimum (we are likely to decrease the objective value in other directions that are not the one in which the line search was performed) running DCA helps us find a better direction for the next iteration of the line search.

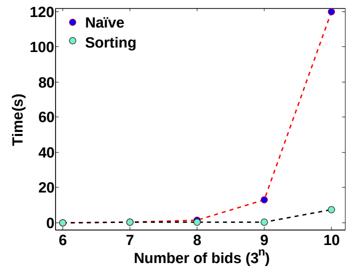


Figure 6. Running-time of our combinatorial algorithm (sorting) compared to the naïve algorithm in log-scale.

5. Experiments

Here, we report the results of some preliminary experiments demonstrating the benefits of our algorithm. All our experiments were carried out using synthetic data. While experiments with data from online auctions have been reported in the literature (Cui et al., 2011), due to confidentiality reasons, the corresponding data is not available to the public. There are other sources of auction data (`http://modelingonlineauctions.com/datasets`), however, these data sets do not include features. To the best of our knowledge, there is no publicly available data set for online auctions including features that could be readily used with our algorithm.

We first tested the speed of our combinatorial algorithm in the simple no-feature case. Figure 6 shows the computational time of that algorithm for finding the optimal solution compared to the naïve approach of evaluating the loss at each point on a 4-Core 2.6 GHz AMD processor with 7GB of RAM. The time our algorithm took to solve the problem with 50,000 points was less than a second, whereas the naïve approach required more than 2 minutes to find the solution. This shows the potential for scalability of our algorithm. Running our algorithm to solve the problem using 200,000 points required 1.87 seconds.

Our experiments were set up as follows. We sampled vectors $\mathbf{x}_i$ in $\mathbb{R}^{200}$ from a standard Gaussian distribution. A *labeling* vector $\mathbf{w} \in \mathbb{R}^{200}$, also sampled from a standard Gaussian, was used to generate the bid vectors $\mathbf{b}_i = (|\mathbf{w} \cdot \mathbf{x}_i|, \frac{1}{2}|\mathbf{w} \cdot \mathbf{x}_i|)$. Absolute values were used to make the dependency between features and bids non-linear.

We are not aware of any published learning algorithm using features to tackle the same problem. In the absence of a baseline, we instead compared the performance of our algorithm with some potential alternatives. One possible algorithm consists of the regularized minimization of the convex surrogate loss L_α of Figure 1(b) parametrized by $\alpha \in [0, 1]$ and defined by

$$L_\alpha(r, \mathbf{b}) = \begin{cases} -r & \text{if } r < b^{(1)} + \alpha(b^{(2)} - b^{(1)}) \\ \Big(\frac{(1-\alpha)b^{(1)} + \alpha b^{(2)}}{\alpha(b^{(1)} - b^{(2)})}\Big)(r - b^{(1)}) & \text{otherwise.} \end{cases}$$

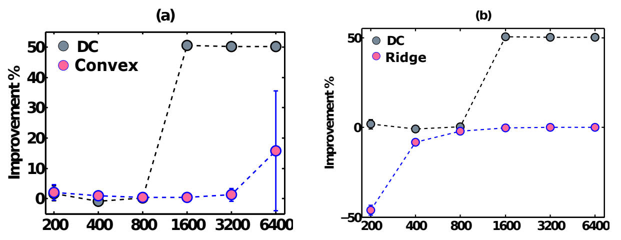


Figure 7. Comparison of the performance of our algorithm (DC) with that of other reserve price optimization techniques: (a) regularized minimization of a convex surrogate loss (CONVEX); (b) ridge regression (RIDGE). The results are reported as percentages of revenue improvement over the no-feature method. The error bars are not indicated since they are too tiny to be discernible at the scale of the plot.

A second alternative consists of using ridge regression to estimate the first bid and use its prediction as the reserve price. A third algorithm consists of minimizing the loss while ignoring the feature vectors $\mathbf{x}_i$, i.e., solving the problem $\min_{r \leq \Lambda} \sum_{i=1}^n L(r, \mathbf{b}_i)$. It is worth mentioning that this third approach is very similar to what advertisement exchanges currently use to suggest reserve prices to publishers. By Equation (1), this is equivalent to estimating the empirical distribution of bids and optimizing the expected revenue with respect to this empirical distribution as in (Ostrovsky & Schwarz, 2011) and (Cesa-Bianchi et al., 2013).

For all our experiments, the parameters Λ, γ and α were tuned via 10-fold cross-validation. The test set was a collection of 20,000 examples drawn from the same distribution. The experiment was repeated 20 times. Figure 7 shows the mean revenue increase obtained for each algorithm over the method using no feature. Since our DC-programming algorithm can converge to a local minimum, the choice of a good starting vector is crucial. For our experiments, it was selected via cross-validation from random starts. Another starting point considered in the cross-validation was the solution to $\min_{\|w\| \leq \Lambda} \sum_{i=1}^n L_\alpha(\mathbf{w} \cdot \mathbf{x}_i, \mathbf{b}_i)$.

Figure 7 shows the results of our experiments. The performance gain achieved by our algorithm is substantial and clearly superior to that of a regularized minimization of a convex surrogate loss or the no-feature algorithm, which is the current state-of-the-art. Since the square loss used in ridge regression is not calibrated with respect to L (it is symmetric around $b^{(1)}$ whereas L is not), we could not expect a high performance using that algorithm. As can be seen from Figure 7, its performance is in fact the worst among the four algorithms tested.

Finally, to test the performance of our algorithm in the presence of noise, we sampled the feature vectors $\mathbf{x}_i$ and $\mathbf{w}$ as before but generated bids as follows:

$$b_i^{(1)} = \max\left((|\mathbf{w} \cdot \mathbf{x}_i| + \sigma\epsilon)_+, (0.5|\mathbf{w} \cdot \mathbf{x}_i| + \sigma\epsilon)_+\right)$$
$$b_i^{(2)} = \min\left((|\mathbf{w} \cdot \mathbf{x}_i| + \sigma\epsilon)_+, (0.5|\mathbf{w} \cdot \mathbf{x}_i| + \sigma\epsilon)_+\right),$$

where $z_+ := \max(z, 0)$, $\epsilon \sim \mathcal{N}(0, 1)$ is a Gaussian random variable, and σ takes values in the set $\{.5, 1, 1.5, 2\}$. We trained our algorithm on a sample of 8,000 points and tested it on a sample of same size, and measured its performance as a function of the noise added to the bids. Table 5 shows the mean revenue improvement achieved over the no-feature algorithm using our algorithm and the regularized minimization of a convex surrogate loss which was the only competitive algorithm in the previous experiment.

Table 1. Comparison of the performance for different noise settings. The results reported are percentages of revenue gained over using no feature using our algorithm (DC) or the regularized minimization based on a convex surrogate loss L_α (CONV).

σ	0.5	1.0	1.5	2.0
DC	$33.59 \pm .65$	$26.43 \pm .56$	$18.38 \pm .57$	$10.68 \pm .65$
CONV	$1.13 \pm .16$	$-.08 \pm .13$	$-1.95 \pm .07$	$-3.54 \pm .07$

Of course, as expected from all learning algorithm, the performance deteriorates as the noise parameter increases. But, while the performance of our algorithm becomes smaller it remains non negligible for even relatively high values of σ. In contrast, we observe that the performance of the surrogate convex loss minimization algorithm decreases rapidly under an even moderate amount of noise. This is likely to be related to the lack of calibration of convex surrogate. Note that this algorithm is even quickly outperformed by the straightforward no-feature approach in the presence of noise.

6. Conclusion

We presented a comprehensive theoretical and algorithmic analysis of the learning problem of revenue optimization in second-price auctions with reserve. The specific properties of the loss function for this problem required a new analysis and new learning guarantees. The algorithmic solutions we presented are practically applicable to revenue optimization problems for this type of auctions in most realistic settings. Our experimental results further demonstrate their effectiveness. Much of the analysis and algorithms presented, in particular our study of calibration questions, can also be of interest in other learning problems.

Acknowledgements

We thank Afshin Rostamizadeh and Umar Syed for several discussions about the topic of this work and ICML reviewers for useful comments. This work was partly funded by the NSF award IIS-1117591.

References

Amin, Kareem, Kearns, Michael, Key, Peter, and Schwaighofer, Anton. Budget optimization for sponsored search: Censored learning in MDPs. In *UAI*, pp. 54–63, 2012.

Balcan, Maria-Florina, Blum, Avrim, Hartline, Jason D., and Mansour, Yishay. Reducing mechanism design to algorithm design via machine learning. *J. Comput. Syst. Sci.*, 74(8):1245–1270, 2008.

Bartlett, Peter L, Jordan, Michael I, and McAuliffe, Jon D. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Blum, Avrim, Kumar, Vijay, Rudra, Atri, and Wu, Felix. Online learning in online auctions. *Theor. Comput. Sci.*, 324(2-3):137–146, 2004.

Cesa-Bianchi, Nicolò, Gentile, Claudio, and Mansour, Yishay. Regret minimization for reserve prices in second-price auctions. In *SODA*, pp. 1190–1204, 2013.

Cui, Ying, Zhang, Ruofei, Li, Wei, and Mao, Jianchang. Bid landscape forecasting in online ad exchange marketplace. In *KDD*, pp. 265–273, 2011.

Debreu, Gerard and Koopmans, Tjalling C. Additively decomposed quasiconvex functions. *Mathematical Programming*, 24, 1982.

Easley, David A. and Kleinberg, Jon M. *Networks, Crowds, and Markets - Reasoning About a Highly Connected World*. Cambridge University Press, 2010.

Horst, R and Thoai, Nguyen V. DC programming: overview. *Journal of Optimization Theory and Applications*, 103(1):1–43, 1999.

Koltchinskii, V. and Panchenko, D. Empirical margin distributions and bounding the generalization error of combined classifiers. *Ann. Statist.*, 30(1):1–50, 2002.

Langford, John, Li, Lihong, Vorobeychik, Yevgeniy, and Wortman, Jennifer. Maintaining equilibria during exploration in sponsored search auctions. *Algorithmica*, 58 (4):990–1021, 2010.

Ledoux, Michel and Talagrand, Michel. *Probability in Banach spaces*. Classics in Mathematics. Springer-Verlag, Berlin, 2011. Isoperimetry and processes, Reprint of the 1991 edition.

Milgrom, P.R. and Weber, R.J. A theory of auctions and competitive bidding. *Econometrica: Journal of the Econometric Society*, pp. 1089–1122, 1982.

Mohri, Mehryar, Rostamizadeh, Afshin, and Talwalkar, Ameet. *Foundations of machine learning*. MIT Press, Cambridge, MA, 2012.

Muthukrishnan, S. Ad exchanges: Research issues. *Internet and network economics*, pp. 1–12, 2009.

Myerson, R.B. Optimal auction design. *Mathematics of operations research*, 6(1):58–73, 1981.

Nisan, Noam, Roughgarden, Tim, Tardos, Éva, and Vazirani, Vijay V. (eds.). *Algorithmic game theory*. Cambridge University Press, Cambridge, 2007.

Ostrovsky, Michael and Schwarz, Michael. Reserve prices in internet advertising auctions: a field experiment. In *ACM Conference on Electronic Commerce*, pp. 59–60, 2011.

Pollard, David. *Convergence of stochastic processes*. Springer Series in Statistics. Springer-Verlag, New York, 1984.

Riley, J.G. and Samuelson, W.F. Optimal auctions. *The American Economic Review*, pp. 381–392, 1981.

Sriperumbudur, Bharath K. and Lanckriet, Gert R. G. A proof of convergence of the concave-convex procedure using Zangwill's theory. *Neural Computation*, 24(6): 1391–1407, 2012.

Tao, Pham Dinh and An, Le Thi Hoai. Convex analysis approach to DC programming: theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

Tao, Pham Dinh and An, Le Thi Hoai. A DC optimization algorithm for solving the trust-region subproblem. *SIAM Journal on Optimization*, 8(2):476–505, 1998.

Tuy, Hoang. Concave programming under linear constraints. *Translated Soviet Mathematics*, 5:1437–1440, 1964.

Tuy, Hoang. Counter-examples to some results on D.C. optimization. Technical report, Institute of Mathematics, Hanoi, Vietnam, 2002.

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

Yen, Ian E.H., Peng, Nanyun, Wang, Po-Wei, and Lin, Shou-De. On convergence rate of concave-convex procedure. In *Proceedings of the NIPS 2012 Optimization Workshop*, 2012.

Yuille, Alan L. and Rangarajan, Anand. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.