Robust Quadratic Programming for Price Optimization

Akihiro Yabe, Shinji Ito, Ryohei Fujimaki
NEC Corporation
a-yabe@cq.jp.nec.com, s-ito@me.jp.nec.com, rfujimaki@nec-labs.com

Abstract

The goal of price optimization is to maximize total revenue by adjusting the prices of products, on the basis of predicted sales numbers that are functions of pricing strategies. Recent advances in demand modeling using machine learning raise a new challenge in price optimization, i.e., how to manage statistical errors in estimation. In this paper, we show that uncertainty in recently-proposed prescriptive price optimization frameworks can be represented by a matrix normal distribution. For this particular uncertainty, we propose novel robust quadratic programming algorithms for conservative lower-bound maximization. We offer an asymptotic probabilistic guarantee of conservativeness of our formulation. Our experiments on both artificial and actual price data show that our robust price optimization allows users to determine best risk-return trade-offs and to explore safe, profitable price strategies.

1 Introduction

Product price is the most significant factor in determining sales. Better pricing strategies (i.e., pricing for multiple products) lead to higher total profits, and determination of the best price strategy is one of the most important strategic business decisions. Price optimization has been actively studied in marketing science and revenue management [Kunz and Crone, 2014] and has already achieved great success in such industries as online retail [Ferreira et al., 2015], fast fashion [Caro and Gallien, 2012], hotels [Koushik et al., 2012; Lee, 2011], and airlines [Côté et al., 2003]. Most price optimization frameworks consist of two stages. The first is demand modeling, which reveals such complicated relationships among prices and sales quantities as price elasticity of demand [Marshall, 2009], cross price elasticity (a.k.a. cannibalization) [van Ryzin and Mahajan, 1999], and the law of diminishing marginal utility [Marshall, 2009]. The second derives the best price strategy by maximizing a utility function, typically total revenue or profit. Most existing studies for multi-product price optimization employ mixed-integer programming [Caro and Gallien, 2012; Koushik et al., 2012; Lee, 2011] due to the discrete nature of individual prices.

Recently, [Ito and Fujimaki, 2017] have proposed prescriptive price optimization. They first employ recent advanced regression techniques (e.g., non-linear, sparse) to flexibly model complex relationships between demand and price with respect to multiple products, and they have shown that the profit maximization problem can be naturally expressed by means of a binary quadratic programming (BQP) problem. [Ito and Fujimaki, 2017] has proposed a relaxation method using semi-definite programming (SDP). Although the SDP method is more efficient than existing mixed-integer programming based methods, its scalability is still limited since it theoretically requires \( O(M^6) \) computational time, where \( M \) is the number of products. [Ito and Fujimaki, 2016] has utilized connection between submodularity and the substitute-goods property [Koushik et al., 2012] known in economics and has proposed an extremely fast proximal gradient algorithm with network-flow optimization that, under certain conditions, guarantees the global optimality of the original BQP problem.

A key drawback in previous works is its optimism with respect to its optimization. There exist two types of uncertainty in prescriptive price optimization, i.e., system noise and estimation error. System noise represents stochastic uncertainty in demand. Estimation error occurs due to the stochastic nature of machine learning, i.e., regression coefficients vary due to stochastic changes in sales records. Although previous works have not taken into account these uncertainties, it is known in the area of stochastic optimization that such uncertainties might significantly degrade the quality of "optimal" solutions due to optimistic bias in objective values [Delage and Ye, 2010; Duchi et al., 2016].

This paper proposes a novel robust quadratic method for price optimization. Our key contributions are mainly two-fold. First, we prove that uncertainty in prescriptive price optimization can be represented by a matrix normal distribution when the least square estimation is employed. This provides a natural robust formulation of price optimization as conservative lower-bound maximization. Although it is common for robust optimization to take system noise into account [Bental et al., 2009], our study is unique in the sense that we explicitly model the uncertainty that occurs in estimation made using machine learning (i.e., estimation error). Second, we propose algorithms for robust quadratic optimization consisting of sequential relaxation to a non-robust counterpart that
employs a non-robust algorithm (such as [Ito and Fujimaki, 2016; 2017]) as its sub-routine. The algorithm is guaranteed to obtain a local optimum by \( O(\log(N/\delta)) \) calls of the non-robust optimization oracle, which is much fewer than that of existing algorithms [Tüttüncü and Koenig, 2004]. We have experimentally evaluated our robust pricing strategy for several values of conservativeness parameters, using an artificial price simulator, and are able to show its efficiency in terms of computational time and actual revenue improvement.

2 Problem Settings

2.1 Quadratic Price Optimization

Suppose we have \( M \) products and their prices and sales quantities are denoted by \( x = (x_1, \ldots, x_M) \in X \subseteq \mathbb{R}^M \) and \( y = (y_1, \ldots, y_M) \in \mathbb{R}^M \), respectively, where \( X \) is a closed bounded set. Further, let us assume that the vector \( y \) of the sales quantities follows a multi-dimensional regression with non-linear basis functions as follows:

\[
y = A^*v(x) + \epsilon,
\]

where \( v : X \to \mathbb{R}^N \) is a \( N \)-dimensional non-linear transformation of \( x \), \( A^* \in \mathbb{R}^{M \times N} \) is the true regression coefficient matrix representing cross price elasticity [Marshall, 2009] (this paper assumes \( N \geq M \)), and \( \epsilon \) is system noise over \( \mathbb{R}^M \) that follows a multidimensional normal distribution \( N(0, \Sigma^\ast) \) with zero mean and covariance matrix \( \Sigma^\ast \in \mathbb{R}^{M \times M} \). Without loss of generality, we assume that the first \( M \) elements in \( v(x) \) are linear features, i.e., \( v(x) = (x^T, N - M \text{ non-linear features})^T. \)

The transformation \( v \) allows us to incorporate such non-linear effects as the law of diminishing marginal utility. We here omit non-price features but it is straightforward to incorporate such features as well, as pointed in [Ito and Fujimaki, 2016].

The goal of our price optimization is to maximize gross profit as defined by \( (x - c)^\top y \), where \( c = (c_1, \ldots, c_M) \in \mathbb{R}^M \) is a cost vector. Hereinafter, for notational simplicity, this paper assumes \( c = 0 \) and thus maximizes gross revenue \( x^\top y \). The problem can then be expressed by the following quadratic programming:

\[
\min_{x \in X} v(x)^\top Q^\ast v(x), \quad \text{where} \quad Q^\ast := \begin{pmatrix} -A^\ast & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{N \times N}.
\]

2.2 Link to Prescriptive Price Optimization

This subsection shows that the prescriptive optimization problem [Ito and Fujimaki, 2016; 2017] is reduced to (2), which considers the following general sparse additive models for demand modeling, \( y_i = \sum_{j=1}^M f_{i,j}(x_j) + b_i \), where \( f_{i,j} : P_j \to \mathbb{R} \) is any uni-variate feature transformation function and \( P_j := \{ p_{i,1}, p_{i,2}, \ldots, p_{i,L} \} \subseteq \mathbb{R} \) is a set of price candidates. The price optimization problem can then be formulated as follows:

\[
\max_{x \in A} \sum_{i=1}^M (x_i - c_i) \left( \sum_{j=1}^M f_{i,j}(x_j) + b_i \right),
\]

where \( X := \{ x \mid x_i \in P_i, i = 1, \ldots, M \} \). Again, for simplicity, we here assume \( c = 0 \), and this does not change the discussion that follows.

Without loss of generality, we can assume that there exists a set of basis \( v_k : \mathbb{R} \to \mathbb{R} \) and coefficients \( a'_{i,j,k} \) for \( k = 1, 2, \ldots, K \) such that

\[
f_{i,j}(x_j) = \sum_{n=1}^N a'_{i,j,k} v_k(x_j)
\]

for all \( x_j \in P_j \). The most typical and simplest setting is \( K = 1 \) and \( v_1(x_j) = x_j \) (linear basis), but in general we can represent (4) by having every \( f_{i,j} \) as a basis for \( i, j = 1, \ldots, M \).

Then we have \( y = Fv(x) + b_i \), where \( F \in \mathbb{R}^{M \times KM} \) is defined by \( F_{i,(k-1)M+j} = a'_{i,j,k} \) for \( k = 1, 2, \ldots, K \) and \( i = 1, 2, \ldots, N \). This shows the equivalence of (3) and (2).

3 Robust Price Optimization

3.1 Robust QP under Matrix Normal Uncertainty

Since the true model \( A^\ast \) cannot be obtained in practice, a natural way is to replace it by an estimator \( \hat{A} \). Suppose we have a set of training samples denoted by \( \{ x_{d,1}, y_{d,1} \}_{d=1}^N \). We assume the least square estimator \( \hat{A} \) of \( A^\ast \) defined as follows:

\[
\hat{A} := \arg \min_A \sum_{d=1}^N ||y_d - Av(x_d)||^2.
\]

For \( n \geq 1 \), let \( I_n \) denote the identity matrix of size \( n \). We define matrices \( V \) and \( W \) by

\[
V := (v(x_1), v(x_2), \ldots, v(x_M)) \in \mathbb{R}^{N \times D}
\]

and

\[
W = V V^\top \in \mathbb{R}^{N \times N}.
\]

Then by the polar decomposition, there exists a matrix \( \Gamma \in \mathbb{R}^{N \times D} \) satisfying \( I_N = V \Gamma \) and \( V = W^{1/2} \Gamma \). Let \( U_{m,n} \) denote the random matrix over \( \mathbb{R}^{m \times n} \), where each entry of \( U_{m,n} \) is independently generated by \( N(0,1) \) and \( N \) represents a normal distribution.

The next proposition provides the distribution of \( \hat{A} \).

Proposition 1. Given \( \{ x_{m,d} \}_{m=1}^M \), \( \hat{A} \) is generated by

\[
\hat{A} = A^\ast + \Sigma^{1/2} U_{m,n} W^{-1/2}.
\]

Proposition 1 indicates that \( \hat{A} \) follows the matrix normal distribution [Gupta and Nagar, 1999], denoted by \( \mathcal{MN}(A^\ast, \Sigma^\ast, W) \), as follows:

\[
P(\hat{A} \mid A^\ast, \Sigma^\ast, W) = \exp(-\frac{1}{2} \text{tr}[W^{-1}(\hat{A} - A^\ast)^\top \Sigma^\ast^{-1}(\hat{A} - A^\ast)])
\]

\[
\frac{1}{2\pi |W|^{D/2} |\Sigma^\ast|^{N/2} |W|^{D/2}}.
\]

In order to derive a formulation for robust price optimization, we first consider the confidence region of \( \hat{A} \). Let us define the estimator \( \hat{\Sigma} \) of \( \Sigma^\ast \) and the confidence region \( C_\lambda \) as follows:

\[
\hat{\Sigma} := \frac{1}{D} (Y - \hat{A} W)(Y - \hat{A} W)^\top
\]

\[
C_\lambda := \{ A \mid A = \hat{A} + \hat{\Sigma}^{1/2} U W^{-1/2}, ||U||_F \leq \lambda \},
\]
where \( U \in \mathbb{R}^{M \times N} \), and \( \| U \|_F \) is the Frobenius norm of \( U \).
Then we derive the robust optimization problem with matrix normal uncertainty as
\[
\max_{x \in \mathcal{X}} \min_{A \in \mathcal{C}_a} x^\top A v(x) \iff \min_{Q \in \mathcal{C}_q} \max_{x \in \mathcal{X}} v(x)^\top Q v(x),
\] (12)
where
\[
\mathcal{C}_a := \{ Q = \hat{Q} + L_1^\top U L_2, \quad \| U \|_F \leq \lambda \},
\] (13)
\[
\hat{Q} := \left( \frac{-\hat{A}}{0} \right), \quad L_1 := \left( \begin{array}{cc}
\hat{\Sigma}^2 & 0 \\
0 & 0
\end{array} \right), \quad L_2 := W^{-\frac{1}{2}}.
\]
Here \( \lambda \geq 0 \) is a scalar parameter for controlling the conservativeness (a smaller \( \lambda \) corresponds to a more aggressive strategy).

### 3.2 Probabilistic Guarantees of Robustness

This section presents an asymptotic probabilistic guarantee of our robust formulation, which can be regarded as a specialization of [Duchi et al., 2016, Theorem 6]. It ensures that we can prevent optimistic bias in objective values and shows that our robust optimum value does not asymptotically over-estimate the true optimum revenue in designed probability.

Let \( \tilde{x} \) be a random variable generated by the chi-squared distribution with degree of freedom \( k \).

**Proposition 2.** Assume that the optimum solution of (2) is unique, and that the historical strategies \( x_d \) are generated by some distribution. Then we have
\[
\lim_{D \to \infty} P \left( \max_{Q \in \mathcal{C}_q} v(\tilde{x})^\top Q v(\tilde{x}) \geq v(x)^\top Q^* v(x) \right) = 1 - \frac{1}{2} P(\lambda^2 \geq \lambda^2),
\] (14)
where \( \tilde{x} := \arg\min_{x \in \mathcal{X}} \max_{Q \in \mathcal{C}_q} v(x)^\top Q v(x) \).

### 4 Algorithms

#### 4.1 Tight, Tractable Upper Bounds

Let us define the function \( f : \mathcal{X} \to \mathbb{R} \) as follows:
\[
f(x) = \max_U v(x)^\top Q v(x)
\] (15)
s.t. \( Q = \hat{Q} + L_1^\top U L_2, \quad \| U \|_F \leq \lambda \). (16)

Our goal, then, is to obtain a strategy \( x \in \mathcal{X} \) which minimizes \( f \). We can remove the maximization in \( f \) and obtain another representation as follows:

**Lemma 3.** It holds that
\[
f(x) = v(x)^\top \hat{Q} v(x) + \lambda \| L_1 v(x) \|_2 \| L_2 v(x) \|,\]
\[
\text{where} \quad \| \bullet \|_2 \text{ is the } \ell_2 \text{ norm of } \bullet.
\]

**Proof.** We have
\[
f(x) = v(x)^\top \hat{Q} v(x) + \max_{U: \| U \|_F \leq \lambda} v(x)^\top L_1^\top U L_2 v(x)
\] (17)
\[
= v(x)^\top \hat{Q} v(x) + \lambda \| L_1 v(x) \|_2 \| L_2 v(x) \|.
\] (18)
The last equality is attained when
\[
U = \frac{\lambda (L_1 v(x))^\top \otimes (L_2 v(x))^\top}{\| (L_1 v(x))^\top \otimes (L_2 v(x))^\top \|_F}.\]
(19)

The term \( \| L_1 v(x) \|_2 \| L_2 v(x) \| \) is a product of the square roots of quartic functions w.r.t. \( v(x) \) and is not tractable for optimization. The next proposition provides a tractable upper bound of \( f \).

**Proposition 4.** For any \( x \in \mathcal{X} \) and \( \gamma > 0 \), it holds that
\[
f(x) \leq g(x, \gamma),
\] (20)
where
\[
g(x, \gamma) := v(x)^\top \left( \hat{Q} + \frac{\gamma M_1 + M_2 / \gamma}{2} \right) v(x).
\] (21)

We have \( M_1 := L_1^\top L_1, \quad M_2 := L_2^\top L_2 \). The equality is attained if and only if \( \gamma \| L_1 v(x) \| = \| L_2 v(x) \| \).

**Proof.** For any \( v = v(x) \in \mathbb{R}^D \), we have
\[
\| L_1 v(x) \| \| L_2 v(x) \| = \sqrt{(v^\top M_1 v)(v^\top M_2 v)} / \gamma \leq (v^\top M_1 v + (v^\top M_2 v) / 2) / 2
\] (22)
The equality condition is \( \gamma (v^\top M_1 v) = (v^\top M_2 v) / \gamma \), which is equivalent to \( \gamma \| L_1 v(x) \| = \| L_2 v(x) \| \).

The following corollary shows theoretical tightness of this upper bound, i.e., the optimal solution of the original problem is achievable with an appropriately chosen \( \gamma \).

**Corollary 5.** It holds that
\[
\min_{x \in \mathcal{X}} f(x) = \min_{x \in \mathcal{X}} \inf_{\gamma \in (0, \infty)} g(x, \gamma).
\] (25)

On the basis of Proposition 4 and Corollary 5, the optimization problem can be re-written as follows:
\[
\min_{x, \gamma} g(x, \gamma) \quad \text{s.t.} \quad x \in \mathcal{X}, \gamma > 0.
\] (26)

#### 4.2 Upper Bound Minimization Algorithms

Since robust optimization is essentially more difficult than its non-robust counterpart, we make the following assumption to ensure that the non-robust counterpart is solvable.

**Assumption 6.** For any \( Q \) satisfying \( Q^\top + Q \geq \hat{Q} + \hat{Q}^\top \), we have a non-robust optimization oracle to solve \( \min_{x \in \mathcal{X}} \| v(x) \|_2 \) \( v(x) \).

This assumption immediately holds for convex non-robust counterparts and also holds for certain types of non-convex ones, such as with branch-and-bound methods [Burer and Vandenbussche, 2008], minimum cut methods for unconstrained binary quadratic programming under submodularity conditions [Kolmogorov and Zabin, 2004], and prescriptive price optimization under substitute goods property conditions [Ito and Fujimaki, 2017; 2016]. Note that this assumption is required for our theoretical analysis, and we can employ an approximation algorithm in practice.

For Eq. (21) and a given \( \gamma \), \( Q_\gamma := \hat{Q} + \lambda (\gamma M_1 + M_2 / \gamma) / 2 \), and hence the oracle in Assumption 6 is applicable. For later convenience, we formally summarize this in the following proposition.

\textsuperscript{2}Even if such \( \tilde{x} \) is not unique, the following discussion remains valid for an arbitrary choice of such \( \tilde{x} \), and therefore here we do not consider uniqueness.
Proposition 7. Under Assumption 6, for a given \( \tilde{\gamma} \in (0, \infty) \), we can obtain \( \tilde{x} := \arg \min_{x \in X} g(x, \tilde{\gamma}) \) by means of a single call of the non-robust optimization oracle.

Further, on the basis of Proposition 4, for a given \( \tilde{x} \), the infimum of \( g(x, \tilde{\gamma}) \) with respect to \( \gamma \) is given by
\[
\tilde{\gamma} = \arg \inf_{\gamma} g(\tilde{x}, \gamma) = \|L_2v(\tilde{x})\|/\|L_1v(\tilde{x})\|. \tag{27}
\]

The above consideration indicates that the minimization of \( g \) w.r.t. one of \( x \) or \( \gamma \) is tractable when the other is fixed. This paper proposes two simple algorithms to solve Eq. (26). Let us define a function \( h : (0, \infty) \to \mathbb{R} \) by
\[
h(\gamma) := \min_{x \in \mathcal{X}} g(x, \gamma). \tag{28}
\]

For robust prescriptive price optimization, we can employ algorithms proposed in [Ito and Fujimaki, 2017; 2016] as \( h(\gamma) \).

Algorithm 1 is a golden section search [Kiefer, 1953] method in which we calculate \( h(\gamma) \) by a single call of the oracle. An advantage of Algorithm 1 is a convergence guarantee for a fixed number of calculation of \( h(\gamma) \) that provides theoretical validity to our upper bound minimization (see Section 4.3 for details). Algorithm 2 is a coordinate descent algorithm that alternates optimizations of \( g(x, \gamma) \) w.r.t. \( x \) and \( \gamma \). Though this algorithm offers no guarantee on the number of iterations needed for convergence, our experiments in Section 6 show that Algorithm 2 empirically converges with only a few iterations and is, in practice, much faster than Algorithm 1. Let us emphasize that, though more advanced optimization techniques might be applicable, our empirical results in Section 6 show that these simple algorithms converge fast enough in practice.

4.3 Convergence analysis

For a range \( 0 < \alpha \leq \beta \), let \( h_{[\alpha, \beta]} : [\alpha, \beta] \to \mathbb{R} \) denote the function restricting the domain of \( h \) onto \( [\alpha, \beta] \). Further, let us define an upper bound \( l \) by \( l := (\|L_1\|_F^2 + \|L_2\|_F^2) \max_{x \in \mathcal{X}} \|v(x)\| \geq \max_{x \in \mathcal{X}} \max_{\|v(\cdot)\|_1} \|L_1v(\cdot)\|_1 \|L_2v(\cdot)\|_1 \). The following theorem guarantees the convergence of Algorithm 1 in a fixed number of iterations.

Theorem 8. Suppose that \( v \) is continuous and that there does not exist \( x \in \mathcal{X} \) such that \( L_1v(x) = L_2v(x) = 0 \). Under Assumption 6, for any \( \delta' > 0 \), setting \( \alpha = 1/\beta \), \( \beta = 2\lambda^2/\delta \) and \( \delta = \delta'/\lambda\beta^2 \), Algorithm 1 outputs \( x \) with the following property by \( O(\log(\lambda/\delta')) \) calls of the non-robust optimization oracle: there exists a local optimum \( x^* \in \mathcal{X} \) of \( f \) satisfying \( f(x) \leq f(x^*) + \delta' \).

For comparison with an existing method, as is discussed in the next subsection, we prove in the following proposition that, under a fairly natural assumption, the function \( f \) becomes convex. In this case, \( f \) has a unique minimum, and our algorithm is able to obtain the global minimum. Here \( S_+ \) is a set of positive semidefinite matrices, and \( Q \) is defined by
\[
Q := \{ (Q + Q^T)/2 | Q = Q_1 + L_1^TUL_2, \|U\|_F \leq \lambda \}. \tag{29}
\]

Proposition 9. If \( \mathcal{X} \) is a convex set, \( v \) is affine, and \( Q \subseteq S_+ \), then \( f \) is a convex function.

5 Related Work

Studies of robust optimization with ellipsoidal uncertainty were initiated independently by [Ben-Tal and Nemirovski, 1998] and [Ghaoui and Lebret, 1997]. For robust quadratic programming \( \min_{x \in \mathcal{X}} x^TQx \), [Ben-Tal et al., 2002] have considered the case in which \( A \) of \( Q = A^T A \) has ellipsoidal uncertainty. For the case in which \( Q \) has ellipsoidal uncertainty, which is also the case this paper considers, [Hallodórsson and Tütüncü, 2003] have proposed an interior point method for a general self-concordant convex-concave function given by [Nemirovski, 1999] to this robust optimization. Further detailed information on robust optimization can be found in [Ben-Tal et al., 2009; Bertsimas et al., 2011].

In contrast to [Hallodórsson and Tütüncü, 2003] which is applicable only when the convexity in Proposition 9 holds, our algorithm is applicable to non-convex robust quadratic programming under the existence of a (possibly approximate) algorithm for a non-robust counterpart. If Proposition 9 holds, both Algorithm 1 and [Hallodórsson and Tütüncü, 2003] are applicable. In such a case, the non-robust counterpart is convex quadratic programming, which can be solved by \( \sqrt{N} \log(1/\delta) \) Newton steps, and thus Algorithm 1 requires \( O(\sqrt{N} \log(1/\delta) \log(N/\delta)) \) Newton steps if \( \|v(x)\|_\infty, \|L_1\|_\infty \) and \( \|L_1\|_\infty \) are bounded. If \( N \) is large, this is better than the \( O(N \log(1/\delta)) \) Newton steps of the algorithm in [Hallodórsson and Tütüncü, 2003].

6 Experiments

We employed the state-of-the-art prescriptive price optimization of [Ito and Fujimaki, 2016] to calculate \( h(\gamma) \) defined by Eq. (28), and compared our robust algorithm and non-robust

---

3If \( \|v(x)\|_\infty, \|L_1\|_\infty \) and \( \|L_1\|_\infty \) are bounded, \( O(l) = O(\log N) \) holds.
counterpart. Note that the results with $\lambda = 0$ are reduced to
the results of the algorithm in [Ito and Fujimaki, 2016].

6.1 Evaluation on Artificial Price Simulator

Simulation Data We conducted experiments with an arti-
ficial price simulator to support our robust formulation and
theoretical analysis. We applied a similar simulation data
generation process as that of the original paper [Ito and Fu-
Jimaki, 2016]. The true demand (regression) model followed
(1), where $v(x) = (x_1, x_2, \ldots, x_M, 1)^T$ were linear features
with $N = M + 1$. Then the true coefficient matrix $A^*$ was
generated by $a_{i,i}^* \sim U([-2M, -M])$, $a_{i,j}^* \sim U([0, 2])$ if
$i \neq j$, and $a_{i,N}^* \sim U([M/2, 3M/2])$. We defined the pric-
ing strategies as $X := \{0.6, 0.7, 0.8, 0.9, 1.0\}^M$ where 1.0
was the list price and 0.9 is 10%-off. The distribution of the
non-diagonal entries $a_{i,j}^*$ for $i = 1, 2, \ldots, M$ were chosen
to balance the effect of $i$th product and the other prices on
the sales quantity of $i$th product. The linear term was arranged
to obtain a moderate optimum strategy of $0.8$ when the non-
diagonal entries were zero. The training data $\{(x_d, y_d)\}_{d=1}^2$ were generated by $x_{d, i} = 1.0, 0.9, 0.8, 0.7, 0.6$ with probability
$0.5, 0.2, 0.1, 0.1, 0.1$ respectively for all $i = 1, 2, \ldots, M$,
where the system noise in $y_d$ followed $N(0, 25M)$. Here
$x_{d, i}$ has importance because it is natural to assume that the
product is sold more at the list price than at discounted prices.
In each experiment, we generated $10$ true models randomly
and $100$ training datasets were generated for each dataset.
The results are averages of all runs.

Effect of Robust Formulation Fig. 1 (left) demonstrates how robust solutions changed along with $\lambda$. We observed:

- The revenues were maximized with $\lambda = 2, 3, 4$ in all cases. By appropriately choosing the robustness parameter $\lambda$, we mitigate uncertainty in price optimization and achieve both better revenue and higher robustness.

- With the small training data ($D = 5M$), the non-robust solutions ($\lambda = 0$) result in much lower revenue than robust solutions. This is because the statistical uncertainty in estimation was large with small training data and therefore non-robust solutions easily became poor. As the size of training data increased, the difference became smaller.

- As $\lambda$ increased, the standard deviation decreased (standard deviation is plotted only for $D = 5M$ cases for visualization purposes). This indicates that the solutions became more conservative and thus more stable.

- Even with the best chosen $\lambda$ value, the obtained revenue is 5-10% worse than the true maximal revenue that was obtained with the true demand model. This difference came from estimation error, which we cannot avoid regardless of the choice of optimization algorithms. This result indicates that the true optimal value is not achievable in expectation and we must always undergo a certain reduction in the expected revenue because of the estimation error.

Probabilistic Guarantee with Finite Samples We next demonstrate that the asymptotic probabilistic guarantee on the upper-bound of the objective value holds with a practical number of training samples and reasonably large $\lambda$, and show that our robust optimization mitigates the over-estimation problem. Fig. 1 (middle) shows the result for $D = 100$ and $M = 10, 30, 50$ along with $\lambda$. The vertical axis is the probability of over-estimation $P(\max_{Q \in C_\lambda} v(x_\lambda)^T Q v(x_\lambda) \leq v(x_\lambda)^T v(x_\lambda))$, where $x_\lambda$ is the robust optimum strategy with $\lambda$. We observed:

- The non-robust optimization over-estimated revenue in 70-
80% of the cases, and the optimism of the non-robust opti-
mization was empirically confirmed.

- Although the empirical probabilities (blue, green, red) were larger than the theoretical asymptotic value (pale blue) in this setting ($D = 100$), the difference rapidly decreased with increasing $\lambda$. With a reasonably large $\lambda$ value (e.g. $\lambda \geq 3$), both the absolute value of empirical probabil-

Comparison of Algorithms We compared Algorithm 1 and Algorithm 2, with a sampling algorithm of min-max optimization that is a baseline for the robust optimization al-
gorithm. The 4th step of the sampling problem solves (12) with the replacement of $U$ by finite samples $\{U_j\}_{j=1,...,1-1}$, and we can then add a new $U_j$ that is a maximizer for the current optimum solution. Observe that the sampling algo-

6.2 Evaluation on Real Point-of-Sales Data We applied the proposed method to real sales history of beers$^a$ [Ito and Fujimaki, 2017; Wang et al., 2015]. The data consisted of prices and sales quantities on 50 products over 642 days. We used the price features for the prediction, and we obtained 50 regression models by means of the least

$^a$The data has been provided by KSP-SP Co., LTD, http://
square estimator. For each product, we set lower and upper bounds on the product price as 60% and 100% of its historically maximum price. We then conducted robust optimization for $\lambda = 0, 30, 60, 90$. Table 1 shows the optimized robust pricing strategies for each $\lambda$. We observed:

- For many products (red), non-robust prices ($\lambda = 0$) were drastically discounted from the average prices, and the prices monotonically increased optimal prices to average prices over $\lambda$. With the most conservative setting $\lambda = 90$, the optimized prices were very close to average values. These products are considered to have large price elasticity of demand. This is natural because the average prices and largely-discounted prices can be seen as, respectively, the most “tested” strategies and the most risky.

- For many different products (green), prices changed little over $\lambda$. These products are considered to have small price elasticity of demand and the list prices (average prices) are the safest and the most profitable.

- For some products (blue), non-robust prices were close to averaged prices. They once dropped to discounted prices and then returned to the averaged prices over $\lambda$. These products are mostly high-valued and therefore are considered to have large cross price elasticity. This complex behavior of the optimized prices was caused by the price changes in the other products.

Although we were unable to evaluate these strategies in real retail stores, we have learned here that our algorithm provides a way to simulate scenarios with different risk levels, and the users can explore for the best pricing strategies on the basis of their domain expertise.

### 7 Summary

This paper has proposed a novel robust quadratic optimization framework for prescriptive price optimization. Our statistical observations have revealed that the uncertainty occurring in estimation in machine learning follows a matrix normal distribution, which has lead us to formulate robust quadratic programming as a conservative upper-bound minimization. Our sequential algorithms for robust quadratic programming converge fast, both practically and theoretically, and can be implemented on the basis of existing non-robust price optimization algorithms. Experimental results on both artificial and actual price data showed that our method enables users to obtain both profitable and safe price strategies in prescriptive price optimization.

---

**Table 1:** Real data experiments with sales histories of beers. On the basis of the least square estimate, we calculated robust optimum strategies for several risk-return parameter $\lambda$ values. $P_i$ is the product ID sorted by the ratio of (optimum price at $\lambda = 0$)/(average). Red and blue products indicate, respectively, risky and stable pricing strategy in small $\lambda$ values. Green products indicate large cross price elasticity.

<table>
<thead>
<tr>
<th>ID</th>
<th>$\lambda$</th>
<th>30</th>
<th>60</th>
<th>90</th>
<th>average</th>
<th>ID</th>
<th>$\lambda$</th>
<th>30</th>
<th>60</th>
<th>90</th>
<th>average</th>
<th>ID</th>
<th>$\lambda$</th>
<th>30</th>
<th>60</th>
<th>90</th>
<th>average</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>126</td>
<td>502</td>
<td>519</td>
<td>541</td>
<td>582</td>
<td>P18</td>
<td>1019</td>
<td>1019</td>
<td>1041</td>
<td>1068</td>
<td>1089</td>
<td>P3</td>
<td>255</td>
<td>229</td>
<td>239</td>
<td>251</td>
<td>252</td>
</tr>
<tr>
<td>P2</td>
<td>159</td>
<td>218</td>
<td>243</td>
<td>263</td>
<td>264</td>
<td>P19</td>
<td>55</td>
<td>77</td>
<td>84</td>
<td>90</td>
<td>90</td>
<td>P36</td>
<td>218</td>
<td>218</td>
<td>209</td>
<td>215</td>
<td>216</td>
</tr>
<tr>
<td>P3</td>
<td>326</td>
<td>457</td>
<td>493</td>
<td>537</td>
<td>541</td>
<td>P20</td>
<td>55</td>
<td>69</td>
<td>81</td>
<td>89</td>
<td>90</td>
<td>P37</td>
<td>208</td>
<td>202</td>
<td>200</td>
<td>206</td>
<td>206</td>
</tr>
<tr>
<td>P4</td>
<td>584</td>
<td>130</td>
<td>167</td>
<td>201</td>
<td>306</td>
<td>P21</td>
<td>675</td>
<td>675</td>
<td>675</td>
<td>675</td>
<td>1099</td>
<td>P38</td>
<td>205</td>
<td>283</td>
<td>271</td>
<td>281</td>
<td>282</td>
</tr>
<tr>
<td>P5</td>
<td>148</td>
<td>219</td>
<td>233</td>
<td>244</td>
<td>245</td>
<td>P22</td>
<td>138</td>
<td>138</td>
<td>189</td>
<td>229</td>
<td>222</td>
<td>P39</td>
<td>189</td>
<td>189</td>
<td>183</td>
<td>187</td>
<td>187</td>
</tr>
<tr>
<td>P6</td>
<td>903</td>
<td>1058</td>
<td>1278</td>
<td>1478</td>
<td>1491</td>
<td>P23</td>
<td>138</td>
<td>138</td>
<td>194</td>
<td>219</td>
<td>220</td>
<td>P40</td>
<td>189</td>
<td>189</td>
<td>182</td>
<td>186</td>
<td>187</td>
</tr>
<tr>
<td>P7</td>
<td>743</td>
<td>743</td>
<td>1005</td>
<td>1210</td>
<td>1226</td>
<td>P24</td>
<td>230</td>
<td>285</td>
<td>285</td>
<td>294</td>
<td>294</td>
<td>P41</td>
<td>189</td>
<td>189</td>
<td>180</td>
<td>186</td>
<td>187</td>
</tr>
<tr>
<td>P8</td>
<td>903</td>
<td>1096</td>
<td>1234</td>
<td>1473</td>
<td>1490</td>
<td>P25</td>
<td>167</td>
<td>178</td>
<td>181</td>
<td>187</td>
<td>187</td>
<td>P42</td>
<td>189</td>
<td>189</td>
<td>189</td>
<td>188</td>
<td>187</td>
</tr>
<tr>
<td>P9</td>
<td>153</td>
<td>200</td>
<td>235</td>
<td>251</td>
<td>252</td>
<td>P26</td>
<td>313</td>
<td>313</td>
<td>307</td>
<td>312</td>
<td>312</td>
<td>P43</td>
<td>91</td>
<td>79</td>
<td>82</td>
<td>89</td>
<td>90</td>
</tr>
<tr>
<td>P10</td>
<td>153</td>
<td>229</td>
<td>240</td>
<td>252</td>
<td>252</td>
<td>P27</td>
<td>267</td>
<td>254</td>
<td>256</td>
<td>265</td>
<td>266</td>
<td>P44</td>
<td>180</td>
<td>171</td>
<td>168</td>
<td>177</td>
<td>178</td>
</tr>
<tr>
<td>P11</td>
<td>600</td>
<td>735</td>
<td>881</td>
<td>980</td>
<td>987</td>
<td>P28</td>
<td>265</td>
<td>252</td>
<td>253</td>
<td>264</td>
<td>264</td>
<td>P45</td>
<td>255</td>
<td>232</td>
<td>239</td>
<td>251</td>
<td>252</td>
</tr>
<tr>
<td>P12</td>
<td>107</td>
<td>142</td>
<td>158</td>
<td>174</td>
<td>176</td>
<td>P29</td>
<td>256</td>
<td>235</td>
<td>243</td>
<td>255</td>
<td>255</td>
<td>P46</td>
<td>1124</td>
<td>694</td>
<td>894</td>
<td>1096</td>
<td>1110</td>
</tr>
<tr>
<td>P13</td>
<td>104</td>
<td>140</td>
<td>155</td>
<td>169</td>
<td>171</td>
<td>P30</td>
<td>1600</td>
<td>1364</td>
<td>1467</td>
<td>1584</td>
<td>1592</td>
<td>P47</td>
<td>275</td>
<td>245</td>
<td>253</td>
<td>269</td>
<td>271</td>
</tr>
<tr>
<td>P14</td>
<td>903</td>
<td>903</td>
<td>1156</td>
<td>1463</td>
<td>1483</td>
<td>P31</td>
<td>189</td>
<td>189</td>
<td>184</td>
<td>188</td>
<td>188</td>
<td>P48</td>
<td>1124</td>
<td>697</td>
<td>839</td>
<td>1084</td>
<td>1101</td>
</tr>
<tr>
<td>P15</td>
<td>153</td>
<td>222</td>
<td>237</td>
<td>251</td>
<td>251</td>
<td>P32</td>
<td>188</td>
<td>188</td>
<td>182</td>
<td>187</td>
<td>187</td>
<td>P49</td>
<td>1505</td>
<td>920</td>
<td>985</td>
<td>1438</td>
<td>1470</td>
</tr>
<tr>
<td>P16</td>
<td>114</td>
<td>184</td>
<td>179</td>
<td>186</td>
<td>187</td>
<td>P33</td>
<td>162</td>
<td>143</td>
<td>148</td>
<td>160</td>
<td>161</td>
<td>P50</td>
<td>1288</td>
<td>1062</td>
<td>912</td>
<td>1232</td>
<td>1257</td>
</tr>
<tr>
<td>P17</td>
<td>675</td>
<td>675</td>
<td>855</td>
<td>1091</td>
<td>1107</td>
<td>P34</td>
<td>162</td>
<td>151</td>
<td>152</td>
<td>161</td>
<td>161</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
References


