Online Linear Programming

Yinyu Ye

Department of Management Science and Engineering

Stanford University

Stanford, CA 94305, U.S.A.

http://www.stanford.edu/~yyye

(Conic) Linear Programming Examples and Reviews

$$
\begin{array}{lll}
\text{(LP)} & \text{minimize} & 2x_1 + x_2 + x_3 \\
& \text{subject to} & x_1 + x_2 + x_3 = 1, \\
& & (x_1; x_2; x_3) \geq \mathbf{0};
\end{array}
$$

$$
\begin{array}{lll}
\text{(SOCP)} & \text{minimize} & 2x_1 + x_2 + x_3 \\
& \text{subject to} & x_1 + x_2 + x_3 = 1, \\
& & \sqrt{x_2^2 + x_3^2} \leq x_1.
\end{array}
$$

$$
\begin{array}{lll}
\text{(SDP)} & \text{minimize} & 2x_1 + x_2 + x_3 \\
& \text{subject to} & x_1 + x_2 + x_3 = 1, \\
& & \begin{pmatrix} x_1 & x_2 \\ x_2 & x_3 \end{pmatrix} \succeq \mathbf{0},
\end{array}
$$

Linear Programming and its Dual

Consider the classical linear program in standard form, called the primal problem,

$$\begin{array}{lll} (LP) & \text{minimize} & \mathbf{c}^T\mathbf{x} \\ & \text{subject to} & A\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\ (\in K), \end{array}$$

where $\mathbf{x} \in \mathcal{R}^n$. The dual problem can be written as:

$$\begin{array}{lll} (LD) & \text{maximize} & \mathbf{b}^T\mathbf{y} \\ & \text{subject to} & A^T\mathbf{y} + \mathbf{s} = \mathbf{c},\ \mathbf{s} \geq \mathbf{0}\ (\in K^*), \end{array}$$

where $\mathbf{y} \in \mathcal{R}^m$ and $\mathbf{s} \in \mathcal{R}^n$. The components of $\mathbf{s}$ are called dual slacks.

Applying Farkars' lemma: If either one is infeasible and the other is feasible, then the other is also unbounded.

LP, SOCP, and SDP Examples

$$\begin{array}{ll} \text{min} & 2x_1 + x_2 + x_3 \\ \text{s. t.} & x_1 + x_2 + x_3 = 1, \\ & (x_1; x_2; x_3) \geq \mathbf{0}. \end{array} \qquad \begin{array}{ll} \text{max} & y \\ \text{s.t.} & \mathbf{e} \cdot y + \mathbf{s} = (2;\ 1;\ 1), \\ & (s_1; s_2; s_3) \geq \mathbf{0}. \end{array}$$

$$\begin{array}{ll} \text{min} & 2x_1 + x_2 + x_3 \\ \text{s.t.} & x_1 + x_2 + x_3 = 1, \\ & x_1 - \sqrt{x_2^2 + x_3^2} \geq 0. \end{array} \qquad \begin{array}{ll} \text{max} & y \\ \text{s.t.} & \mathbf{e} \cdot y + \mathbf{s} = (2;\ 1;\ 1), \\ & s_1 - \sqrt{s_2^2 + s_3^2} \geq 0. \end{array}$$

For the SOCP case: $2 - y \geq \sqrt{2(1-y)^2}$. Since $y = 1$ is feasible for the dual, $y^* \geq 1$ so that the dual constraint becomes $2 - y \geq \sqrt{2}(y - 1)$ or $y \leq \sqrt{2}$. Thus, $y^* = \sqrt{2}$, and there is no duality gap.

$$
\begin{array}{ll}
\text{minimize} & \begin{pmatrix} 2 & .5 \\ .5 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 & x_2 \\ x_2 & x_3 \end{pmatrix} \\
\text{subject to} & \begin{pmatrix} 1 & .5 \\ .5 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 & x_2 \\ x_2 & x_3 \end{pmatrix} = 1, \\
& \begin{pmatrix} x_1 & x_2 \\ x_2 & x_3 \end{pmatrix} \succeq \mathbf{0},
\end{array}
$$

$$
\begin{array}{ll}
\text{maximize} & y \\
\text{subject to} & \begin{pmatrix} 1 & .5 \\ .5 & 1 \end{pmatrix} y + \mathbf{s} = \begin{pmatrix} 2 & .5 \\ .5 & 1 \end{pmatrix}, \\
& \mathbf{s} = \begin{pmatrix} s_1 & s_2 \\ s_2 & s_3 \end{pmatrix} \succeq \mathbf{0}.
\end{array}
$$

LP Duality Theories

Theorem 1 *(LP Weak Duality Theorem) Let feasible regions $\mathcal{F}_p$ and $\mathcal{F}_d$ be non-empty. Then,*

$$\mathbf{c}^T\mathbf{x} \geq \mathbf{b}^T\mathbf{y} \quad \textit{where} \quad \mathbf{x} \in \mathcal{F}_p,\ (\mathbf{y}, \mathbf{s}) \in \mathcal{F}_d.$$

$$\mathbf{c}^T\mathbf{x} - \mathbf{b}^T\mathbf{y} = \mathbf{c}^T\mathbf{x} - (A\mathbf{x})^T\mathbf{y} = \mathbf{x}^T(\mathbf{c} - A^T\mathbf{y}) = \mathbf{x}^T\mathbf{s} \geq 0.$$

This theorem shows that a feasible solution to either problem yields a bound on the value of the other problem. We call $\mathbf{c}^T\mathbf{x} - \mathbf{b}^T\mathbf{y}$ the duality gap.

From this we have important implication: if we have $\mathbf{c}^T\mathbf{x} = \mathbf{b}^T\mathbf{y}$ where $\mathbf{x}$ is feasible for LP and $\mathbf{y}$ is feasible for LD, then they are optimal for LP and LD respectively.

Is the reverse also true?

LP Strong Duality Theorem

Theorem 2 *(LP Strong Duality Theorem) Let $\mathcal{F}_p$ and $\mathcal{F}_d$ be non-empty. Then, $\mathbf{x}^*$ is optimal for (LP) if and only if the following conditions hold:*

i) $\mathbf{x}^* \in \mathcal{F}_p$;

ii) *there is* $(\mathbf{y}^*, \mathbf{s}^*) \in \mathcal{F}_d$ *such that*

iii) $\mathbf{c}^T \mathbf{x}^* = \mathbf{b}^T \mathbf{y}^*$.

Given $\mathcal{F}_p$ and $\mathcal{F}_d$ being non-empty, we like to prove that there is $\mathbf{x}^* \in \mathcal{F}_p$ and $(\mathbf{y}^*, \mathbf{s}^*) \in \mathcal{F}_d$ such that $\mathbf{c}^T \mathbf{x}^* \leq \mathbf{b}^T \mathbf{y}^*$, or to prove that

$$A\mathbf{x} = \mathbf{b},\ A^T\mathbf{y} \leq \mathbf{c},\ \mathbf{c}^T\mathbf{x} - \mathbf{b}^T\mathbf{y} \leq 0,\ \mathbf{x} \geq \mathbf{0}$$

has a feasible solution if both LP and LD are feasible – using Farkas' lemma again.

Theorem 3 *(LP Primal-Dual Theorem) If (LP) and (LD) both have feasible solutions then both problems have optimal solutions and the optimal objective values of the objective functions are equal.*

If one of (LP) or (LD) has no feasible solution, then the other is either unbounded or has no feasible solution. If one of (LP) or (LD) is unbounded then the other has no feasible solution.

The above theorems show that if a pair of feasible solutions can be found to the primal and dual problems with equal objective values, then these are both optimal. The converse is also true; there is no "gap."

Optimality Conditions:

$$\left\{ (\mathbf{x}, \mathbf{y}, \mathbf{s}) \in (\mathcal{R}^n_+, \mathcal{R}^m, \mathcal{R}^n_+) : \begin{array}{rcl} \mathbf{c}^T\mathbf{x} - \mathbf{b}^T\mathbf{y} & = & \mathbf{0} \\ A\mathbf{x} & = & \mathbf{b} \\ -A^T\mathbf{y} - \mathbf{s} & = & -\mathbf{c} \end{array} \right\},$$

which is a system of linear inequalities and equations. Now it is easy to verify whether or not a pair $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ is optimal.

LP Complementarity Condition

For feasible $\mathbf{x}$ and $(\mathbf{y}, \mathbf{s})$, $\mathbf{x}^T\mathbf{s} = \mathbf{x}^T(\mathbf{c} - A^T\mathbf{y}) = \mathbf{c}^T\mathbf{x} - \mathbf{b}^T\mathbf{y}$ is called the complementarity gap.

Since both $\mathbf{x}$ and $\mathbf{s}$ are nonnegative, $\mathbf{x}^T\mathbf{s} = 0$ implies that $x_j s_j = 0$ for all $j = 1, \ldots, n$, where we say $\mathbf{x}$ and $\mathbf{s}$ are complementary to each other.

$$
\begin{aligned}
x_j s_j &= 0, \ \forall j, \\
A\mathbf{x} &= \mathbf{b}, \ \mathbf{x} \geq \mathbf{0} \\
A^T\mathbf{y} + \mathbf{s} &= \mathbf{c}, \ \mathbf{s} \geq \mathbf{0}.
\end{aligned}
$$

Theorem 4 *(LP Strict-Complementarity Theorem) For any pair of LP and LD where both are feasible, there exists an optimal or complementarity solution pair such that*

$$x_j + s_j > 0, \ \forall j.$$

Resource Allocation LP

$$
(LP) \quad \begin{array}{lllll}
\text{maximize} & x_1 & +2x_2 & & \\
\text{subject to} & x_1 & & \leq 1 & \\
 & & x_2 & \leq 1 & \\
 & x_1 & +x_2 & \leq 1.5 & \\
 & x_1, & x_2 & \geq 0. &
\end{array}
$$

$$
\max \ \mathbf{p}^T\mathbf{x} \quad \text{s.t.} \quad A\mathbf{x} \leq \mathbf{r}, \quad \mathbf{x} \geq \mathbf{0}
$$

where

- $\mathbf{p}$: profit margin vector
- A: resources consumption rate matrix
- $\mathbf{r}$: available resource vector
- $\mathbf{x}$: allocation decision vector

Dual Interpretation of Resource Allocation: Liquidation Pricing

$$(LD)\quad \begin{array}{lllll} \text{minimize} & y_1 & +y_2 & +1.5y_3 & \\ \text{subject to} & y_1 & & +y_3 & \geq 1 \\ & & y_2 & +y_3 & \geq 2 \\ & y_1, & y_2, & y_3 & \geq 0. \end{array}$$

$$\min\ \mathbf{r}^T\mathbf{y} \quad \text{s.t.} \quad A^T\mathbf{y} \geq \mathbf{p}, \quad \mathbf{y} \geq \mathbf{0}$$

where

- $\mathbf{y}$: the fair price vector
- $A^T\mathbf{y} \geq \mathbf{p}$: competitiveness
- $\mathbf{y} \geq \mathbf{0}$: positivity
- $\min\ \mathbf{r}^T\mathbf{y}$: minimize the total liquidation cost

A Combinatorial Auction Pricing Problem

Given the m different states that are mutually exclusive and exactly one of them will be true at the maturity. A contract on a state is a paper agreement so that on maturity it is worth a notional $\$1$ if it is on the winning state and worth $\$0$ if is not on the winning state. There are n orders betting on one or a combination of states, with a price limit and a quantity limit.

Order Data: The jth order is given as $(\mathbf{a}_j \in R^m_+,\ \pi_j \in R_+,\ q_j \in R_+)$: $\mathbf{a}_j$ is the combination betting vector where each component is either 1 or 0

$$\mathbf{a}_j = \begin{pmatrix} a_{1j} \\ a_{2j} \\ \ldots \\ a_{mj} \end{pmatrix},$$

where 1 is winning and 0 is non-winning; π_j is the price limit for one such a contract share, and q_j is the maximum number of shares the better like to buy.

World Cup Information Market

Order:	#1	#2	#3	#4	#5
Argentina	1	0	1	1	0
Brazil	1	0	0	1	1
Italy	1	0	1	1	0
Germany	0	1	0	1	1
France	0	0	1	0	0
Bidding Prize: π	0.75	0.35	0.4	0.95	0.75
Quantity limit: $\mathbf{q}$	10	5	10	10	5
Order fill: $\mathbf{x}$	x_1	x_2	x_3	x_4	x_5

Parimutuel Call Auction Mechanism I

Let x_j be the number of contracts awarded to the jth order. Then, the jth better will pay the amount

$$\pi_j \cdot x_j$$

and the total collected amount is

$$\sum_{j=1}^{n} \pi_j \cdot x_j = \pi^T \mathbf{x}$$

If the ith state is the winning state, then the auction organizer need to pay back

$$\left(\sum_{j=1}^{n} a_{ij} x_j \right)$$

The question is, how to decide $\mathbf{x} \in R^n$.

Parimutuel Call Auction Mechanism II

$$\begin{array}{lrl} \max & \pi^T \mathbf{x} - \max_j \{\mathbf{a}_{\cdot j}^T \mathbf{x}\} & \\ \text{s.t.} & \mathbf{x} & \leq \mathbf{q}, \\ & \mathbf{x} & \geq \mathbf{0}. \end{array}$$

$$\begin{array}{lrl} \max & \pi^T \mathbf{x} - \max(A^T \mathbf{x}) & \\ \text{s.t.} & \mathbf{x} & \leq \mathbf{q}, \\ & \mathbf{x} & \geq \mathbf{0}. \end{array}$$

This is NOT a linear program.

Parimutuel Call Auction Mechanism III: Risk-Free LP model

$$
\begin{array}{lrl}
\max & \pi^T \mathbf{x} - z & \\
\text{s.t.} & A\mathbf{x} - \mathbf{e} \cdot z & \leq \mathbf{0}, \\
 & \mathbf{x} & \leq \mathbf{q}, \\
 & \mathbf{x} & \geq 0;
\end{array}
$$

where $\mathbf{e}$ is the vector of all ones.

$\pi^T \mathbf{x}$: the optimistic amount can be collected. z: the worst-case amount need to pay back.

Parimutuel Call Auction Mechanism IV: The Dual

$$\begin{array}{lrl} \min & \mathbf{q}^T\mathbf{y} & \\ \text{s.t.} & A^T\mathbf{p}+\mathbf{y} & \geq \pi, \\ & \mathbf{e}^T\mathbf{p} & = 1, \\ & (\mathbf{p},\mathbf{y}) & \geq 0. \end{array}$$

$\mathbf{p}$ represents the state price.

What is $\mathbf{y}$?

Price information gaps/differentials/slacks where their weighted sum we like to minimize.

Parimutuel Call Auction Mechanism V: Complementarity Condition

$x_j > 0$	$\mathbf{a}_j^T\mathbf{p} + y_j = \pi_j$ and $y_j \geq 0$ so that $\mathbf{a}_j^T\mathbf{p} \leq \pi_j$
$0 < x_j < q_j$	$y_j = 0$ so that $\mathbf{a}_j^T\mathbf{p} = \pi_j$
$x_j = q_j$	$y_j > 0$ so that $\mathbf{a}_j^T\mathbf{p} < \pi_j$
$x_j = 0$	$\mathbf{a}_j^T\mathbf{p} + y_j > \pi_j$ and $y_j = 0$ so that $\mathbf{a}_j^T\mathbf{p} > \pi_j$

The price is Fair:

$$\mathbf{p}^T(A\mathbf{x} - \mathbf{e}\cdot z) = 0 \quad \text{implies} \quad \mathbf{p}^T A\mathbf{x} = \mathbf{p}^T\mathbf{e}\cdot z = z;$$

that is, the worst case cost equals the worth of total shares. Moreover, if a lower bid wins the auction, so does the higher bid on any same type of bids.

World Cup Information Market Result

Order:	$\#1$	$\#2$	$\#3$	$\#4$	$\#5$	State Price
Argentina	1	0	1	1	0	0.2
Brazil	1	0	0	1	1	0.35
Italy	1	0	1	1	0	0.2
Germany	0	1	0	1	1	0.25
France	0	0	1	0	0	0
Bidding Price: π	0.75	0.35	0.4	0.95	0.75	
Quantity limit: $\mathbf{q}$	10	5	10	10	5	
Order fill: $\mathbf{x}^*$	5	5	5	0	5	

Question 1: The uniqueness of dual prices?

Nonlinear Convex Programming Mechanism/Regularization

To value the uncertain revenue s_i between the worst-case cost and the actual cost when state i is realized:

$$\begin{array}{rll} \max & \pi^T \mathbf{x} - z + U(\mathbf{s}) & \\ \text{s.t.} & A^T \mathbf{x} - \mathbf{e} \cdot z + \mathbf{s} & = \mathbf{0}, \\ & \mathbf{x} & \leq \mathbf{q}, \\ & \mathbf{x} & \geq \mathbf{0}. \end{array}$$

where $U(\cdot)$ is a concave (risk aversion) and increasing value function for the possible slack revenues $\mathbf{s} = \mathbf{e} \cdot z - A^T \mathbf{x}$. For example,

$$U(\mathbf{s}) = \min_i (\mathbf{s}), \quad \text{or} \quad U(\mathbf{s}) = \sum_i u(s_i).$$

If $u(\cdot)$ is a strictly concave function, then the state price vector is unique.

Question 2: Online Auction?

Online Combinatorial Auction Mechanism

- Traders come one by one with an order $(\mathbf{a}, \pi, q)$.
- Market maker has to make an order-fill decision as soon as an order arrives – may need to accept bets that do not have a matching bet yet.
- Market maker still hopes: i) to pay the winners almost completely from the stakes of losers, and ii) to update state prices reflect the traders' aggregated belief on outcome states

Market Scoring Rules

Market Scoring Rule: Traders report their beliefs/prices, $\mathbf{p}$, on outcome states directly; then payment is determined by a scoring rule, $s_i(\mathbf{p})$, on reported probability vector $\mathbf{p}$.

For example,

$$s_i(\mathbf{p}) = b\log(p_i) + 1, \ \forall i.$$

Suppose constant $b = 0.5$ and you bet the distribution

$$\mathbf{p} = (0.2, 0.3, 0.2, 0.25, 0.05)$$

over the five teams. Then, if Brazil wins, your profit for each share under (LMSR) is

$$0.5\log(0.3) + 1 = 0.398.$$

But if France wins, your profit for each share is

$$0.5\log(0.05) + 1 = -0.498.$$

Online Combinatorial Auction: Sequential Convex Programming Mechanism

Given the previous $(t-1)$ order-fills $\bar{x}_1, \ldots, \bar{x}_{t-1}$ on input $\{\pi_j, \mathbf{a}_j, q_j\}_{j=1}^{t-1}$ until time t, the t^{th} order-fill decision is to choose x_t such that,

$$
\begin{array}{rll}
\max & \pi_t x_t - z + U(\mathbf{s}) + \sum_{j=1}^{t-1} \pi_j \bar{x}_j & \\
\text{s.t.} & \mathbf{a}_t x_t - \mathbf{e} \cdot z + \mathbf{s} + \sum_{j=1}^{t-1} \mathbf{a}_j \bar{x}_j & = \mathbf{0}, \\
& x_t & \le q_t, \\
& x_t & \ge 0.
\end{array}
$$

$(\pi_t, \mathbf{a}_t, q_t)$: the newly arrived bidding data.

z: the new worst-case cost.

$\sum_{j=1}^{t-1} \pi_j \bar{x}_j$: collected revenue before the new arrival.

$\sum_{j=1}^{t-1} \mathbf{a}_j \bar{x}_j$: outstanding shares in each state before the new arrival.

Online Combinatorial Auction: Theorem and Initial Prices

Theorem 5 *The SCPM with a concave and increasing value function is equivalent to choosing x_t in order to minimize a convex risk measure on random return revenue. Moreover, any convex risk measure can be used to construct an SCPM model with a corresponding concave value function.*

Initial Prices and Shares:

$$\begin{aligned} \max \quad & -z + U(\mathbf{s}) \\ \text{s.t.} \quad & -\mathbf{e} \cdot z + \mathbf{s} \quad = \mathbf{0}, \end{aligned}$$

or

$$\max \quad -z + U(\mathbf{e}z)$$

KKT condition: let z^0 be the optimizer: $\mathbf{e}^T \nabla U(z^0) = 1,$ so that z^0 can be viewed as the initial outstanding shares in each state, and $\nabla U(z^0)$ contains initial prices for each state!

Online Combinatorial Auction: Simplified Formulation and Computation

$$\begin{array}{llll} \max & \pi_t x_t - z + U(\mathbf{s}) & & \\ \text{s.t.} & \mathbf{a}_t x_t - \mathbf{e} z + \mathbf{s} & = -\mathbf{b}^{t-1}, \\ & x_t & \leq q_t, \\ & x_t & \geq 0, \end{array}$$

where $\mathbf{b}^{t-1} = \sum_{j=1}^{t-1} \mathbf{a}_j \bar{x}_j$–outstanding shares in each state. Or

$$\begin{array}{ll} \max & \pi_t x_t - z + U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) \\ \text{s.t.} & x_t \leq q_t, \\ & x_t \geq 0. \end{array}$$

This is actually a two-variable problem.

KKT or Optimality Conditions for the Simplified Formulation

$$\begin{array}{ll} \max & \pi_t x_t - z + U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) \\ \text{s.t.} & x_t \le q_t, \\ & x_t \ge 0, \end{array}$$

$$\begin{aligned} \pi_t - \mathbf{a}^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) - \lambda &\le 0,\ \lambda \ge 0, \\ -1 + \mathbf{e}^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) &= 0 \\ x_t(\pi_t - \mathbf{a}^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) - \lambda) &= 0 \\ \lambda(q_t - x_t) &= 0. \end{aligned}$$

The equality $\mathbf{e}^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) = 1$ implies that z is an implicit function of x_t.

Let $(\bar{x}_t,\ \bar{z}^t)$ be the optimizer. Then new outstanding shares $\mathbf{b}^t = \mathbf{b}^{t-1} + \mathbf{a}_t \bar{x}_t$ and $\mathbf{p}^t := \nabla U(\mathbf{e}\bar{z}^t - \mathbf{b}^t)$ represents the state price for each state after the tth order is optimally filled.

Online KKT Implications

$\bar{x}_t = 0$	$\lambda = 0$ so that $\pi_t < \mathbf{a}_t^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1})$
$\bar{x}_t = q_t$	$\lambda > 0$ so that $\pi_t > \mathbf{a}_t^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1})$
$0 < \bar{x}_t < q_t$	$\lambda = 0$ so that $\pi_t = \mathbf{a}_t^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1})$

In the first case: $\bar{z}^t = \bar{z}^{t-1}$, $\mathbf{b}^t = \mathbf{b}^{t-1}$ and $\mathbf{p}^t = \mathbf{p}^{t-1}$.

In the third case: $(\bar{z}^t, \bar{x}_t)$ is the solution of two equations

$$\mathbf{e}^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) = 1 \quad \text{and} \quad \mathbf{a}_t^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) = \pi_t.$$

Sequential Convex Programming Mechanism/Algorithm

- Step 1: if $\pi_t < \mathbf{a}_t^T \mathbf{p}^{t-1}$, $\bar{x}_t = 0$, $\mathbf{b}^t = \mathbf{b}^{t-1}$ and $\mathbf{p}^t = \mathbf{p}^{t-1}$; otherwise go to Step 2;
- Step2: Solve

$$\mathbf{e}^T \nabla U(\mathbf{e}z - \mathbf{a}_t q_t - \mathbf{b}^{t-1}) = 1$$

  and let $\bar{z}^t$ be the root. If $\pi_t > \mathbf{a}_t^T \nabla U(\mathbf{e}y - \mathbf{a}_t q_t - \mathbf{b}^{t-1})$, $\bar{x}_t = q_t$, $\mathbf{b}^t = \mathbf{b}^{t-1} + \mathbf{a}^t q_t$ and $\mathbf{p}^t = \nabla U(\mathbf{e}\bar{z}^t - \mathbf{b}^t)$; otherwise go to Step 3;
- Step 3: Solve for $(\bar{z}^t, \bar{x}_t)$ from

$$\mathbf{e}^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) = 1 \quad \text{and} \quad \mathbf{a}_t^T \nabla U(\mathbf{e}z - \mathbf{a}_t x_t - \mathbf{b}^{t-1}) = \pi_t.$$

  Let $\mathbf{b}^t = \mathbf{b}^{t-1} + \mathbf{a}^t \bar{x}_t$ and $\mathbf{p}^t = \nabla U(\mathbf{e}\bar{z}^t - \mathbf{b}^t)$.

Sequential Convex Programming Mechanism Example

Consider the five teams playing for the world cup. Let the vaule function $U(\mathbf{s}) = \sum_i u(s_i)$ and $u(s_i) = 0.2 \cdot \log(s_i)$; and the first bid comes as

$$\pi_1 = 0.75,\ \mathbf{a}_1 = (1;\ 1;\ 0;\ 0;\ 0),\ \text{and } q_1 = 2.5.$$

We see $\mathbf{p}^0 = (1/5; 1/5; 1/5; 1/5; 1/5)$ and $\mathbf{b}^0 = (1; 1; 1; 1; 1)$.

Step 1: $\mathbf{a}_1^T \mathbf{p}^0 = 0.4 < 0.75 = \pi_1$, so that we go to Step 2;

Step 2: We solve the equation

$$2\frac{0.2}{y-3.5} + 3\frac{0.2}{y-1} = 1, \Rightarrow\ \bar{y}^1 = 4.$$

But

$$\sum_i a_{it} u'(4 - a_{it}2.5 - 1) = \frac{0.2}{0.5} + \frac{0.2}{0.5} = 0.8 > 0.75 = \pi_1,$$

so that we go to Step 3;

Step 3:We solve the two equations

$$2\frac{0.2}{y - x_1 - 1} + 3\frac{0.2}{y - 1} = 1 \text{ and } 2\frac{0.2}{y - x_1 - 1} = 0.75$$

so that the root $\bar{y}^1 = 17/5$ and $\bar{x}_1 = 28/15$. Then

$$\mathbf{p}^1 = (3/8; 3/8; 1/12; 1/12; 1/12), \text{ and } \mathbf{b}^1 = (43/15; 43/15; 1; 1; 1).$$

Market Scoring Rules and SCPM

Theorem 6 *Every scoring rule has a concave and increasing value function representation in the Convex Programming Mechanism/Regularization model. Conversely, every concave and increasing value function induces a scoring rule that can be truthfully implemented. Furthermore, the properties of the value function and its derivatives, such as boundedness, smoothness, span, etc, determine other desired or undesired properties of the mechanism, such as the worst-case loss, properness, risk-attitude, etc.*

- Exponential [Hanson, 2003]: $u(s_i) = b \cdot (1 - \exp(-s_i/b)),$ for some positive constant b.
- Logarithmic [Peters et al. 2007]: $u(s_i) = b \cdot \log(s_i),$ for some positive constant b.
- Quadratic [Chen and Pennock 2007]:

$$u(s_i) = \begin{cases} b \cdot (1 - (1 - s_i/b)^2) & 0 \leq s_i \leq b \\ b & s_i \geq b \end{cases}$$

for some positive constant b.

General Offline and Online Resource Allocation Linear Programming

Now consider a more general resource allocation linear program:

$$
\begin{array}{lll}
\text{maximize}_{\mathbf{x}} & \sum_{t=1}^{n} \pi_t x_t & \\
\text{subject to} & \sum_{t=1}^{n} a_{it} x_t \leq b_i, & \forall i = 1, ..., m \\
 & 0 \leq x_t \leq 1, & \forall t = 1, ..., n
\end{array}
$$

Each order t requests up to one unit of a bundle of m goods, and is willing to pay π_t for it.

In real applications, data/information is revealed sequentially, and one has to make decisions sequentially based on what is known. That is, we only know $\mathbf{b}$ at the start, but

- the constraint matrix is revealed column by column sequentially along with the corresponding objective coefficient.
- an irrevocable decision must be made as soon as an order arrives without observing or knowing the future data.

An Example

	order 1($t = 1$)	order 2($t = 2$)	.....	Inventory($\mathbf{b}$)
Price(π_t)	\$100	\$30	...	
Decision	x_1	x_2	...	
Pants	1	0	...	100
Shoes	1	0	...	50
T-shirts	0	1	...	500
Jacket	0	0	...	200
Socks	1	1	...	1000

Sequential Convex Programming Mechanism?

$$
\begin{array}{lll}
\text{(SCPM):} & \text{maximize}_{x_t,\mathbf{s}} & \pi_t x_t + u(\mathbf{s}) \\
 & \text{s.t.} & \mathbf{a}_t x_t + \mathbf{s} = \mathbf{b} - \sum_{j=1}^{t-1} \mathbf{a}_j \bar{x}_j, \\
 & & 0 \le x_t \le 1, \ \mathbf{s} \ge \mathbf{0}.
\end{array}
$$

$\sum_{j=1}^{t-1} \mathbf{a}_j \bar{x}_j$: allocated resource vector before the new arrival.

Possible Concave Value Functions:

- Exponential: $u(s_i) = b \cdot (1 - \exp(-s_i/b)),$ for some positive constant b.
- Logarithmic: $u(s_i) = b \cdot \log(s_i),$ for some positive constant b.
- Quadratic:

$$
u(s_i) = \begin{cases} b \cdot (1 - (1 - s_i/b)^2) & 0 \le s_i \le b \\ b & s_i \ge b \end{cases} \quad \text{for some positive constant } b.
$$

Pros and Cons?

More on the Online Linear Programming Model

Main Assumptions

- The columns $\mathbf{a}_t$ arrive in a random order.
- We know the total number of columns n a priori.

Other technical assumptions

- $0 \leq a_{it} \leq 1$, for all (i, t);
- $\pi_t \geq 0$ for all t

The algorithm/mechanism quality is evaluated on the expected performance over all the permutations comparing to the offline optimal solution, i.e., an algorithm $\mathcal{A}$ is c-competitive if and only if

$$E_\sigma \left[\sum_{t=1}^{n} \pi_t x_t(\sigma, \mathcal{A}) \right] \geq c \cdot OPT(A, \pi).$$

Comments on the Online Model

- The online approach is distribution-free. It allows for great robustness in practical problems. If the columns or arrivals are drawn $i.i.d.$ from a certain distribution (either known or unknown to the decision maker), then the first assumption is automatically met.
- The second assumption is necessary for one to obtain a near optimal solution. However, it can be relaxed to an approximate knowledge of n or the length of decision horizon.
- Both assumptions are reasonable and standard in many operations research and computer science applications.

Main Theorems of Online Linear Programming Mechanism

Theorem 7 *For any fixed* $0 < \epsilon < 1$, *there is no online algorithm for solving the linear program with competitive ratio* $1 - \epsilon$ *if*

$$B < \frac{\log(m)}{\epsilon^2}.$$

Theorem 8 *For any fixed* $0 < \epsilon < 1$, *there is a* $1 - \epsilon$ *competitive online algorithm for solving the linear program if*

$$B \geq \Omega\left(\frac{m \log(n/\epsilon)}{\epsilon^2}\right).$$

Agrawal, Wang and Y [Operations Research 2014]

Comments on the Main Theorems

- The condition of B to hold the main result is independent of the size of $OPT(A, \pi)$ or the objective coefficients, and is also independent of any possible distribution of input data. Therefore, it's checkable.
- The condition on sample size $1/\epsilon^2$ is necessary as it is common in many learning-based algorithm.
- The condition is proportional only to $\log(n)$ so that it is way below to satisfy everyone's demand.

Key Ideas to Prove Negative Result

- Consider $m = 1$ and inventory level B, one can construct an instance where $OPT = B$, and there will be a loss of $\sqrt{B}$ with a high probability, which give an approximation ratio $1 - \frac{1}{\sqrt{B}}$.
- Consider general m and inventory level B for each good. We are able to construct an instance to decompose the problem into $\log(m)$ separable problems, each of which has an inventory level $B/\log(m)$ on a composite "single good" and $OPT = B/\log(m)$.
- Then, with hight probability each "single good" case has a loss of $\sqrt{B/\log(m)}$ and the total loss of $\sqrt{B \cdot \log(m)}$. Thus, approximation ratio is at best $1 - \frac{\sqrt{\log(m)}}{\sqrt{B}}$.

Key Ideas to Prove Positive Result

The proof of the positive result is constructive and based on a learning policy.

- There is no distribution known so that any type of stochastic optimization models is not applicable.
- Unlike dynamic programming, the decision maker does not have full information/data so that a backward recursion can not be carried out to find an optimal sequential decision policy.
- Thus, the online algorithm needs to be learning-based, in particular, learning-while-doing.

But what to lean?

Itemized Pricing Method

The problem would be easy if there is an ”ideal price” vector:

	Bid 1($t = 1$)	Bid 2($t = 2$)	.....	Inventory($\mathbf{b}$)	$\mathbf{p}^*$
Bid(π_t)	$100	$30	...		
Decision	x_1	x_2	...		
Pants	1	0	...	100	$45
Shoes	1	0	...	50	$45
T-shirts	0	1	...	500	$10
Jackets	0	0	...	200	$55
Hats	1	1	...	1000	$15

One-Time Learning Algorithm

We start with a simple

- Set $x_t = 0$ for all $1 \le t \le \epsilon n$;
- Solve the ϵ portion of the problem

$$\begin{array}{lll} \text{maximize}_{\mathbf{x}} & \sum_{t=1}^{\epsilon n} \pi_t x_t & \\ \text{subject to} & \sum_{t=1}^{\epsilon n} a_{it} x_t \le (1-\epsilon)\epsilon b_i & i = 1, ..., m \\ & 0 \le x_t \le 1 & t = 1, ..., \epsilon n \end{array}$$

  and get the optimal dual solution $\hat{\mathbf{p}}$;

- Determine the future allocation x_t as:

$$x_t = \begin{cases} 0 & \text{if } \pi_t \le \hat{\mathbf{p}}^T \mathbf{a}_t \\ 1 & \text{if } \pi_t > \hat{\mathbf{p}}^T \mathbf{a}_t \end{cases}$$

  as long as $a_{it} x_t \le b_i - \sum_{j=1}^{t-1} a_{ij} x_j$ for all i; otherwise, set $x_t = 0$.

One-Time Learning Algorithm Result

Theorem 9 *For any fixed* $\epsilon > 0$*, the one-time learning algorithm is* $(1 - \epsilon)$ *competitive for solving the linear program when*

$$B \geq \Omega\left(\frac{m \log (n/\epsilon)}{\epsilon^3}\right)$$

Outline of the Proof

- With high probability, we clear the market;
- With high probability, the revenue is near-optimal if we include the initial ϵ portion revenue;
- With high probability, the first ϵ portion revenue, a learning cost, doesn't contribute too much.

Then, we prove that the one-time learning algorithm is $(1-\epsilon)$ competitive under condition $B \geq \frac{6m\log(n/\epsilon)}{\epsilon^3}$.

But this is one ϵ factor higher than the lower bound...

Dynamic Price Updating Algorithm

In the dynamic price learning algorithm, we update the price at time ϵn, $2\epsilon n$, $4\epsilon n$, ..., till $2^k\epsilon \geq 1$.

At time $\ell \in \{\epsilon n, 2\epsilon n, ...\}$, the price vector is the optimal dual solution to the following linear program:

$$
\begin{array}{lll}
\text{maximize}_{\mathbf{x}} & \sum_{t=1}^{\ell} \pi_t x_t & \\
\text{subject to} & \sum_{t=1}^{\ell} a_{it} x_t \leq (1 - h_\ell)\frac{\ell}{n} b_i & i = 1, ..., m \\
 & 0 \leq x_t \leq 1 & t = 1, ..., \ell
\end{array}
$$

where

$$h_\ell = \epsilon \sqrt{\frac{n}{\ell}};$$

and this price vector is used to determine the allocation for the next immediate period.

Dynamic Price Updating Algorithm

In the dynamic price updating algorithm, we update the price at time ϵn, $2\epsilon n$, $4\epsilon n$ ... At time $\ell \in \{\epsilon n, 2\epsilon n, ...\}$, the price is the optimal dual solution to the following linear program:

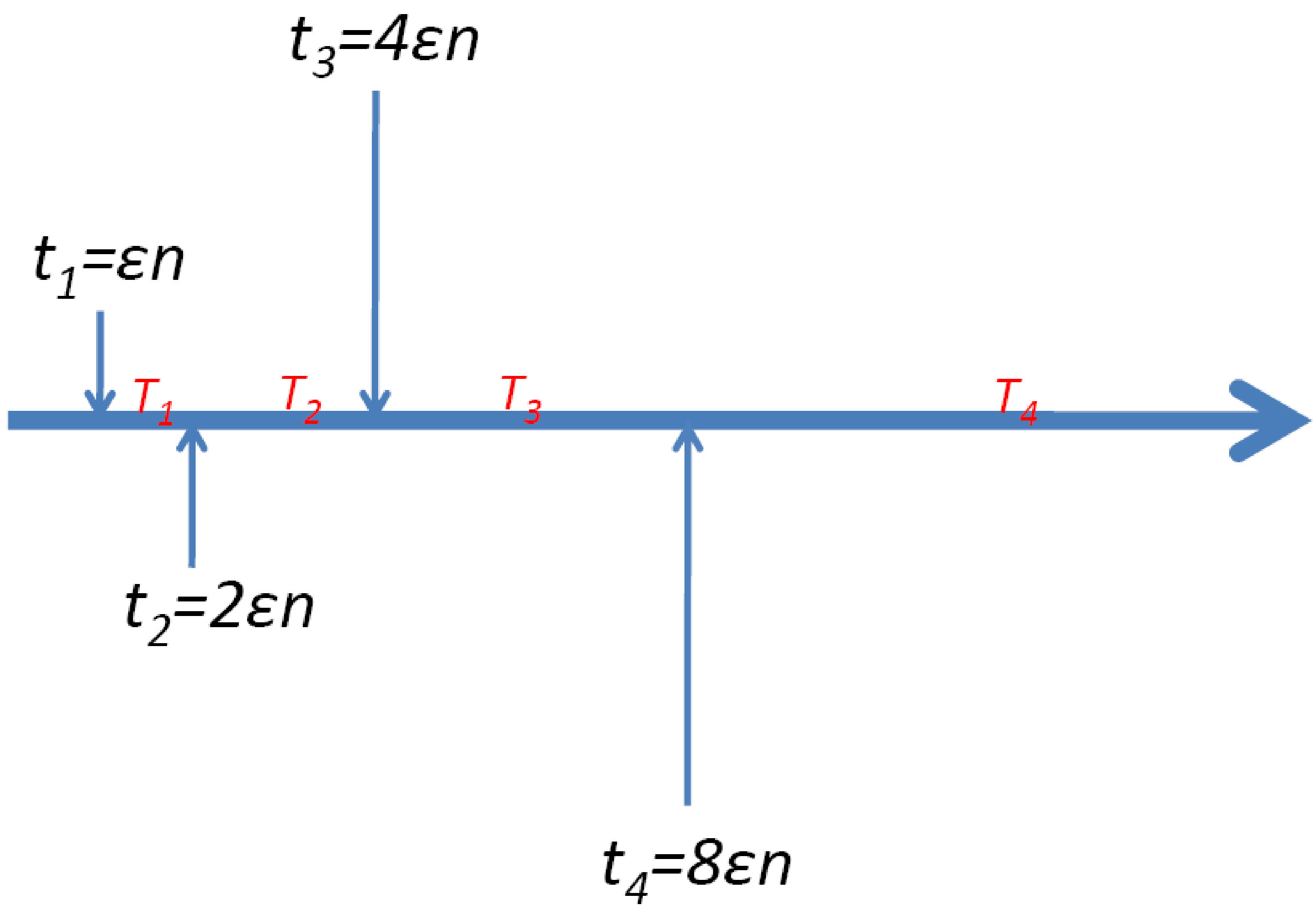
$$\begin{array}{lll} \text{maximize}_{\mathbf{x}} & \sum_{t=1}^{\ell} \pi_t x_t & \\ \text{subject to} & \sum_{t=1}^{\ell} a_{it} x_t \leq (1-h_\ell)\frac{\ell}{n} b_i & i = 1, ..., m \\ & 0 \leq x_t \leq 1 & t = 1, ..., \ell \end{array}$$

where

$$h_\ell = \epsilon \sqrt{\frac{n}{\ell}}$$

And this price is used to determine the allocation for the next immediate period.

Geometric Pace/Grid of Price Updating



Comments on Dynamic Learning Algorithm

- In the dynamic algorithm, we update the prices $\log_2(1/\epsilon)$ times during the entire time horizon.
- The numbers h_ℓ play an important role in improving the condition on B in the main theorem. It basically balances the probability that the inventory ever gets violated and the lost of revenue due to the factor $1 - h_\ell$.
- Choosing large h_ℓ (more conservative) at the beginning periods and smaller h_ℓ (more aggressive) at the later periods, one can now control the loss of revenue by an ϵ order while the required size of B can be weakened by an ϵ factor.

Related Ongoing Work on Random-Permutation

	Sufficient Condition	Learning
Kleinberg [2005]	$B \geq \frac{1}{\epsilon^2}$, for $m = 1$	Dynamic
Devanur et al [2009]	$OPT \geq \frac{m^2 \log(n)}{\epsilon^3}$	One-time
Feldman et al [2010]	$B \geq \frac{m \log n}{\epsilon^3}$ **and** $OPT \geq \frac{m \log n}{\epsilon}$	One-time
Agrawal et al [2010]	$B \geq \frac{m \log n}{\epsilon^2}$ **or** $OPT \geq \frac{m^2 \log n}{\epsilon^2}$	Dynamic
Molinaro/Ravi [2013]	$B \geq \frac{m^2 \log m}{\epsilon^2}$	Dynamic
Kesselheim et al [2014]	$B \geq \frac{\log m}{\epsilon^2}$	Dynamic*
Gupta/Molinaro [2014]	$B \geq \frac{\log m}{\epsilon^2}$	Dynamic*
Agrawal/Devanur [2014]	$B \geq \frac{\log m}{\epsilon^2}$	Dynamic*

Table 1: Comparison of several existing results

Online Resource Allocation with Production Costs

One may consider more general resource allocation problems with production costs:

$$
\begin{array}{ll}
\text{maximize}_{\mathbf{x}} & \sum_{j=1}^{n} (\pi_j x_j - \sum_k c_{ijk} y_{ijk}) \\
\text{s.t.} & \sum_k y_{ijk} = a_{ij} x_j; \ \forall i, j, \\
 & \sum_{i,j} y_{ijk} \le c_k; \ \forall k, \\
 & 0 \le x_j \le 1, \ \forall\, j = 1, ..., n;
\end{array}
$$

where c_{ijk} is the cost allocate good/resource i, which is produced by producer $k = 1, ..., K$, to bidder j; and c_k is the production capacity of producer k.

Price-Post Learning

- Selling a good in a fixed horizon T, and there is no salvage value for the remaining quantities after the horizon.
- The production lead time is long so that the inventory B is fixed and can not be replenished during the selling season.
- Demand arrives in a Poisson process, where the arrival rate $\lambda(p)$ depends only on the instantaneous price posted by the seller.
- Objective is to maximize the expected revenue.

Historically, researchers mostly consider the case where the demand function $\lambda(p)$ is known.

Unknown Demand Function: Parametric and Non-parametric Learning

In this case, the seller has to learn the demand function "on the fly".

- Parametric learning approach is to make the demand function $\lambda(p)$ satisfy a parametric family (e.g., $\lambda(p) = b - ap$ or $\lambda(p) = e^{-ap}$).
- In the parametric case, a dynamic programming with Bayesian update is usually considered.
- Sometimes the demand function doesn't belong to any function form (or one doesn't know which form it belongs to), so that considering a wrong demand family may be costly.
- Non-parametric approach only poses few requirements on the demand function thus is very robust to model uncertainty.
- In a non-parametric learning algorithm, more price experimentations have to be made and the question is how to reduce the learning cost.

Evaluation of the Learning Algorithm: Asymptotic Regret I

For any pricing policy/algorithm π, denote its expected revenue by $J^\pi(B, T; \lambda)$. Also denote the optimal expected revenue as $J^*(B, T; \lambda)$. Then, we consider the regret

$$R^\pi(B, T; \lambda) = 1 - \frac{J^\pi(B, T; \lambda)}{J^*(B, T; \lambda)}$$

Since no one knows which λ is realized, so we consider the worst regret

$$\sup_{\lambda \in \Gamma} R^\pi(B, T; \lambda)$$

where Γ is a general family of functions which we will define later.

- However it is still very hard to evaluate the regret for a low volume.
- Therefore we consider a high-volume regime where the inventory B, together with the demand rate λ, grows proportionally (multiplied in an positive integer n) and consider the asymptotic behavior of $R^\pi(n \cdot B, T; n \cdot \lambda)$.
- This type of evaluation criterion is widely adopted.

Prior Best Results of Learning Algorithms

- For the parametric case, the best algorithm achieves a regret of $O(n^{-1/3})$, while for the non-parametric case, it achieves a regret of $O(n^{-1/4})$ (By Besbes and Zeevi, 2009).
- There is a lower bound showing that no algorithm can do better than $O(n^{-1/2})$, for both parametric and non-parametric case.
- The algorithms for both cases use one-time learning, that is, learning first and doing second. In the learning period, a number of prices are tested and the best one is selected to be implemented in the doing period.
- As presented earlier, under the auction model, the best learning algorithm can achieve an asymptotic regret of $O(n^{-1/2})$.

Could we close the gaps?

Assumptions on the Demand Function and Main Result

- $\lambda(p)$ is bounded
- $\lambda(p)$ (and $r(p) = p\lambda(p)$) is Lipschitz continuous. Also there exists an inverse demand function $\gamma(\lambda)$ that is also Lipschitz continuous.
- $r(\lambda) = \lambda\gamma(\lambda)$ is (strictly) concave
- $r''(\lambda)$ exists and $r''(\lambda) \leq -\alpha < 0$ for a fixed positive number α.

Theorem 10 *(Wang, Deng and Y 2014, Operations Research) Let the above assumptions hold. Then, there exists an admissible pricing policy π, such that for all $n \geq 1$,*

$$\sup_{\lambda \in \Gamma} R_n^{\pi_\delta}(n \cdot B, T; n \cdot \lambda) \leq C(\log n)^{4.5} \cdot n^{-1/2}$$

for some constant C that only depends on Γ, B and T.

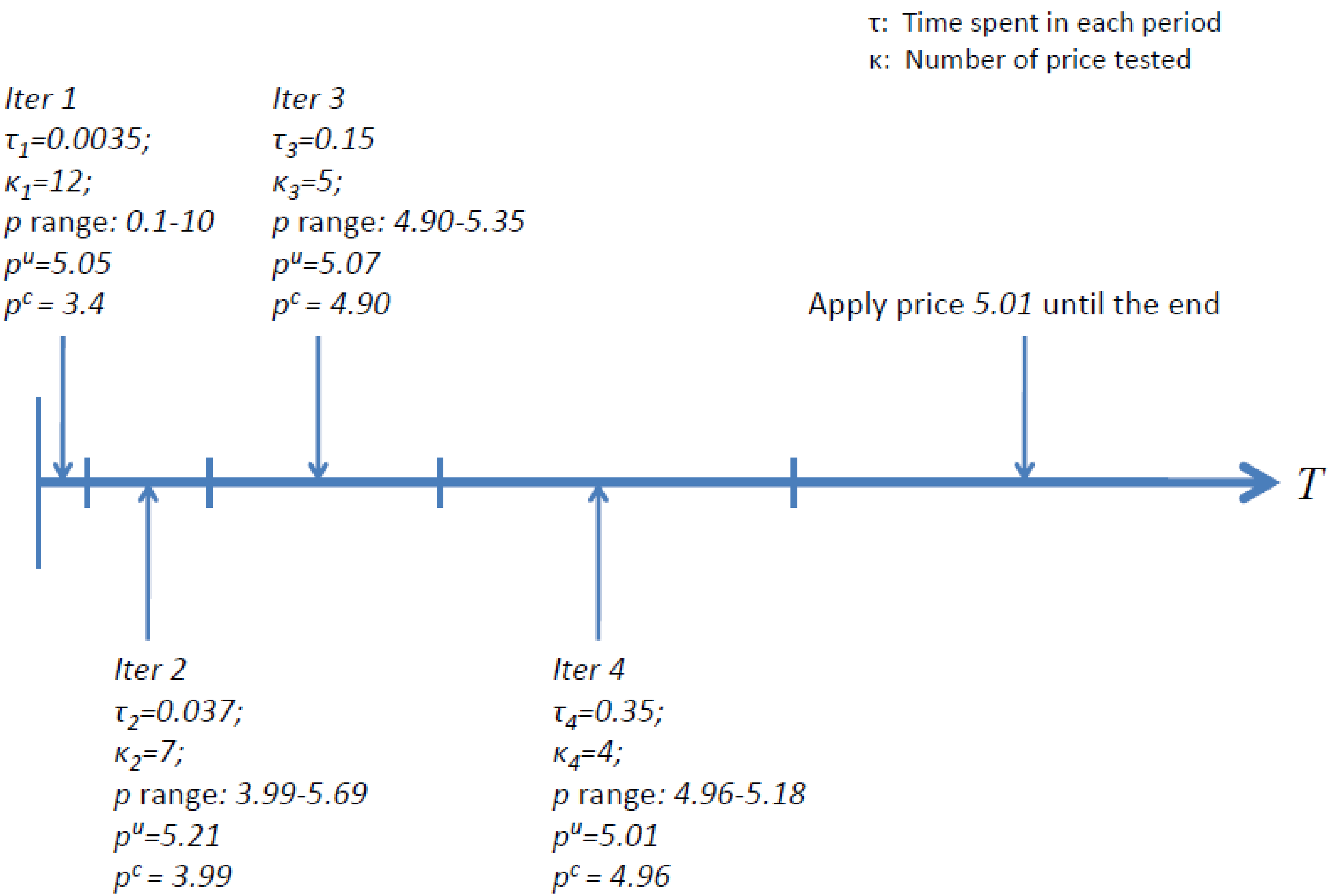
Description of the Algorithm

The algorithm is a dynamic pricing algorithm, where we integrate the "learning" and "doing" periods. Specifically, we

- Divide the time into geometric intervals
- Keep a shrinking admissible price range
- Perform and apply price experimentation in each time interval within the current price range
- Find the optimal price, update the price range for the next time interval

The key is to balance and interplay demand learning (exploration) and near-optimal pricing (exploitation).

Geometric Pace of Price Testing:


τ: Time spent in each period
κ: Number of price tested
Iter 1
τ_1=0.0035;
κ_1=12;
p range: 0.1-10
p^u=5.05
p^c = 3.4
Iter 3
τ_3=0.15
κ_3=5;
p range: 4.90-5.35
p^u=5.07
p^c = 4.90
Apply price 5.01 until the end
T
Iter 2
τ_2=0.037;
κ_2=7;
p range: 3.99-5.69
p^u=5.21
p^c = 3.99
Iter 4
τ_4=0.35;
κ_4=4;
p range: 4.96-5.18
p^u=5.01
p^c = 4.96

Summary and Future Questions on OLP

- $B = \frac{\log m}{\epsilon^2}$ is now a necessary and sufficient condition (differing by a constant factor).
- Thus, they are near-optimal online algorithms for a very general class of online linear programs.
- The algorithms are distribution-free and/or non-parametric, thereby robust to distribution/data uncertainty.
- The dynamic learning has the feature of "learning-while-doing", and is provably better than one-time learning by a factor.
- Buy-and-sell or double market?
- Price-Posting multi-good model?
- Online Utility Formulation for Resource Allocation?