LINEAR PROGRAMMING WITH ONLINE LEARNING

TATSIANA LEVINA, YURI LEVIN, JEFF MCGILL, AND MIKHAIL NEDIAK

SCHOOL OF BUSINESS, QUEEN'S UNIVERSITY, 143 UNION ST., KINGSTON, ON, K7L 3N6, CANADA

E-MAIL:{TLEVIN,YLEVIN,JMCGILL,MNEDIAK}@BUSINESS.QUEENSU.CA

Abstract. We propose online decision strategies for *time-dependent* sequences of linear programs which use no distributional and minimal geometric assumptions about the data. These strategies are obtained through Vovk's aggregating algorithm which combines recommendations from a given strategy pool. We establish an average-performance bound for the resulting solution sequence.

Keywords: Linear programming; on-line learning; aggregating algorithm.

1. Introduction.

In LP implementations, the problem of data uncertainty is typically addressed through multiple runs, sensitivity analysis, or other techniques, but it is generally recognized that optimal solutions are, at best, improvements over decisions that might have been reached without the model. Less haphazard approaches can be used when a modeler can make assumptions about the statistical properties of the data. Such situations have been studied for many years (see, for example, [7]) and are addressed extensively in the field of *stochastic programming*. Unfortunately, in many cases the distributional properties of the input data may be as difficult to obtain as the unknown parameters themselves. More recently, work in *robust optimization* avoids the need for distributional assumptions by focusing on the optimal worst-case performance of the system when the data belong to a given set (see [2]). In both stochastic programming and robust optimization, all possible realizations of the data need to be specified and considered in advance. Moreover, the success of a solution depends on exploiting the stochastic or geometric structure of possible data realizations.

In practice, many LP models are solved repeatedly over time; indeed, applications in stock-cutting, scheduling, revenue management, logistics, and many other areas can involve solutions of LP's multiple times per day. While the structure of the model typically remains constant over successive runs, the model's input parameters can vary significantly and are typically not known exactly before a decision must be made. However, in some cases it may be possible to observe parameter values *after* the decision. While this can be true for any of a model's parameters,

Corresponding author: Yuri Levin. E-mail: ylevin@business.queensu.ca. Address: School of Business, Queen's University, 143 Union St.,Kingston, ON, K7L 3N6, CANADA.

it is particularly true for the objective function coefficients since their true values are often determined by random and uncontrollable external market price and cost factors that, nonetheless, can be observed accurately *ex post*.

In this paper, we consider the solution of sequences of LP problems in which the problem structure remains constant, but the objective function coefficients (henceforth, cost vector) may vary over time. In particular, we study *decision strategies* for such sequences. We make no distributional and only minimal geometric assumptions about the data; thus, these strategies do not require explicit consideration of all possible future realizations and only use past observations of the problem data to choose a decision in the current time interval. Our objective is to find, within a family of possible decision strategies, a strategy that will perform, under any circumstances, almost as well as the best decision strategy in the family for those circumstances. Our approach to this problem falls within the general class of *online methods*. Generally, an online method is an algorithm which does not have immediate access to the entire input of the problem. Instead, the problem is revealed to the algorithm incrementally, and in response to each incomplete portion of the input, the algorithm must take an irreversible action without access to the future input. This framework is frequently used to model problems which evolve over time. There is a vast literature on online methods and their *competitive analysis*, "whereby the quality of an online algorithm on each input sequence is measured by comparing its performance to that of an optimal offline algorithm" (see [4] and also [8, 10]). This literature includes a significant number of publications on online discrete optimization problems.

The prior work on the subject of online linear optimization includes [13] and [18]. The work [13] considers a randomized "follow the expected leader" algorithm, while [18] utilizes a deterministic algorithm based on the expected gradient ascent. The analysis of each algorithm is based on an additive performance criterion, and the resulting (arithmetic) average performance of the decision sequences proposed by these algorithms trails the average performance of a fixed optimal solution chosen in hindsight by no more than $O(1/\sqrt{T})$ (where T is the number of stages in the online problem). In this paper, the performance criterion is multiplicative, and the performance is measured in terms of the geometric mean which is natural in certain situations as demonstrated later in the paper. We show that our performance bound compares favorably to the existing bounds in terms of the number of stages T.

The online strategies developed in this paper are inspired by two streams of previous work: 1) work on universalization of strategies for portfolio selection, which began with Cover's work on constant rebalanced portfolios, [6], and 2) more general work on universalization of online algorithms [12], and [16]. The methodology of the Aggregating Algorithm (AA) discussed in [16] is a very general form of the weighted majority algorithm. For applications of the weighted majority algorithm to metrical task systems and other general frameworks see [3].

In Section 2 we describe the decision problem in terms of time-dependent families of linear programs with varying cost vectors and show how AA can be applied to such problems. In Section 3, we derive a performance bound when the reference family of strategies consists of all (fixed) feasible solutions to linear programs. We also show that the strategy produced by the AA in this case passes the 'reality test' of weak convergence in situations when the linear program is, in fact, fixed, but the algorithm is not aware of that; that is, the value of the AA's solution converges to the optimal value for a fixed cost vector.

2. Problem Description.

Consider a *Decision Maker* (DM) who needs to make periodic decisions modeled by a family of linear programming (LP) problems

$$\max\{\mathbf{c}_t^\top \mathbf{x} : \ \mathbf{A}\,\mathbf{x} \le \mathbf{b}\}, \tag{1}$$

where the values $\mathbf{c}_t \in \Omega \subseteq \mathbb{R}^n$ may vary with time, and are determined by the outcome of *Reality* at time t, $t = 1, 2, ...$ Our only assumption about the cost vector sequence is that each of its elements belongs to Ω. There are no other statistical or geometric assumptions about the sequence (distribution, independence, etc). At time t, DM has a memory of the *past* cost vectors $\mathbf{c}_1, \dots, \mathbf{c}_{t-1}$, but has to choose the solution vector before the outcome of Reality at time t becomes known. Therefore, the decision can only use $\mathbf{c}_1, \dots, \mathbf{c}_{t-1}$.

We denote by X the feasible set of (1), i.e $X = \{\mathbf{x} : \mathbf{A}\,\mathbf{x} \le \mathbf{b}\}$, and assume that X is bounded, and the sets X and Ω are such that

$$\mathbf{c}^\top \mathbf{x} \ge 0 \ \text{ for all } \ \mathbf{x} \in X \ \text{ and } \ \mathbf{c} \in \Omega. \tag{2}$$

The family of linear programming problems which we consider here is quite general. Indeed, a linear program in the most general form (with equality and inequality constraints and arbitrary bounds on the variables) can be reduced to the form (1) by a series of elementary transformations. Thus, the real restrictions are the requirement that X is bounded, which often arises in the theoretical analysis of linear programming, and condition (2). The latter will hold, for example, when the set Ω is a cone, and X is a subset of $-\Omega^\circ$ (negative of the polar cone of Ω). Even if (2) does not hold for certain linear programs, there may exist a simple transformation of the problem such that the condition does hold. The following statement describes when (2) is not restrictive in this sense (the proof of this as well as other statements in the article can be found in the online Appendix [15]):

Lemma 1. *If Ω is a subset of a cone K such that its polar cone K° has a nonempty interior, then there exists a linear transformation of variables from X to a polyhedral set X' (defined by a system of inequalities with the same matrix A) such that (2) is satisfied in the new variables.*

At time t, the Decision Maker chooses a solution $\mathbf{x}_t$ based on the outcomes of reality in the previous time steps and, perhaps, some additional information. A problem of this generality is unlikely to be solvable. Therefore we assume that additional information may only come in the form of recommended solutions from a *Pool* of strategies. In general, Pool may take various forms such as the set of all feasible solutions, a collection of complex algorithms based on external information, or even all realizations of some stochastic process (such strategy pools were considered, for example, by [14]). Also, the performance of DM's actual strategy will be compared against the Pool.

Mathematically, we must only require that Pool is a measurable space: a set Θ with a σ-algebra. At each time t, the Pool makes a recommendation $\boldsymbol{\xi}_t$; that is, the recommendation of the pool element $\theta \in \Theta$ is $\boldsymbol{\xi}_t(\theta) \in X$. We will require that the function $\boldsymbol{\xi}_t : \Theta \to X$ is measurable. Measurability requirements for Θ and $\boldsymbol{\xi}_t$ are essentially the only restrictions on the type of strategy pools that may be considered in this approach. These requirements are trivially satisfied when Θ is finite, but are essential for treatment of any infinite Θ. Note also that defining a probability measure on Θ would turn it into a measure space and $\boldsymbol{\xi}_t$ into a random variable. At this point, however, we do not define this measure. It will only appear as a part of the algorithm which uses Pool's recommendations.

The online problem the Decision Maker faces is modelled as the following T-stage perfect-information game between DM, Pool, and Reality. Consistent with some learning and game-theoretic literature (e.g. [9]), the game participants do not have to be adversarial. For example, while Reality chooses its stage t move after DM and has a knowledge of DM's move, it does not have to choose the worst-case cost vector for DM. The algorithm that we describe in the end of this section will be applicable regardless of whether game participants are adversarial or not. The game proceeds according to the following

Protocol:

FOR $t = 1, 2, \ldots, T$:

Pool recommends $\boldsymbol{\xi}_t$

DM views Pool's recommendation and chooses $\mathbf{x}_t$ in X.

Reality chooses $\mathbf{c}_t$ from Ω.

The average performance of DM in the game is measured according to the geometric mean:

$$G_T(DM) := \left(\prod_{t=1}^{T} \mathbf{c}_t^\top \mathbf{x}_t \right)^{\frac{1}{T}}. \tag{3}$$

Similarly, the average performance of any strategy $\theta \in \Theta$ is measured according to

$$G_T(\theta) := \left(\prod_{t=1}^{T} \mathbf{c}_t^\top \boldsymbol{\xi}_t(\theta) \right)^{\frac{1}{T}}. \tag{4}$$

In this paper, we describe how one can use Vovk's aggregating algorithm (AA) to find a solution strategy for DM. In the case that Pool consists of fixed solution strategies ($\Theta = X$), we obtain a bound on the average performance of the AA relative to the performance of any strategy. The corollary of our result, described in the next section, is that the long-term average performance of the AA is as good as the long-term average performance of any fixed feasible solution.

In the following example we illustrate the advantage of the geometric mean as a performance criterion and explain why one generally needs to compare the performance of an algorithm to the one of an arbitrary feasible solution.

Example 1. Let T be a multiple of 2, and the dimension of the polytope $n = 2$. The observed cost vectors are

$$\mathbf{c}_t = \begin{cases} (1, 0), & t \text{ odd}, \\ (0, 1), & t \text{ even}, \end{cases}$$

and the polytope $X = \{(x_1, x_2) : x_1 + x_2 = 1,\ x_1 \geq 0,\ x_2 \geq 0\}$. The solution optimal in hindsight with respect to the geometric mean performance criterion is given by the optimization problem

$$\begin{aligned} \max \quad & \left(((0,1)\cdot \mathbf{x})^{T/2} ((1,0)\cdot \mathbf{x})^{T/2} \right)^{\frac{1}{T}} = \sqrt{x_1 x_2} \\ \text{s.t.} \quad & x_1 + x_2 = 1,\ \ x_1, x_2 \geq 0. \end{aligned}$$

The unique optimal solution $x_1^* = x_2^* = \frac{1}{2}$ with the value $\frac{1}{2}$ has an attractive symmetry property which appears to be quite natural for this example. If, instead, we used an arithmetic average performance criterion then an optimal solution would not be identified uniquely. All points of X would be optimal in hindsight including the vertices which only have value 0 if the geometric mean is used.

Generally, the geometric mean criterion is more likely to result in the unique optimal solution (in hindsight), since it is strictly log-concave whenever the observed cost vectors span the linear hull of X. The arithmetic mean criterion does not necessarily result in the unique solution even if the above condition is satisfied because of the possibility of dual degeneracy. We also point out that geometric average is a natural criterion in financial applications with linear objectives representing the "rate of return" as, for example, in the work of [6].

To simplify the algorithm it may also be tempting to choose the set of vertices of X as strategies and use one of the existing methods for combining expert advice, see, for example, [5]. However, the performance guarantees of the method in this case would be in terms of the vertex that is optimal *ex post*. Again, this example shows why a stronger result in terms of the geometric mean performance might be achieved if we consider all feasible solutions.

Informally, the aggregating algorithm works as follows. In the beginning, each strategy $\theta \in \Theta$ is assigned some initial weight. This is achieved by specifying a *prior distribution* on Θ. If there is no prior information about the strategies, one would like this prior distribution to be *noninformative* in the language of Bayesian statistics. For example, when Θ is a bounded set in a finite-dimensional space, we can let the prior distribution be uniform on Θ. Such a distribution will contain no prior information about the strategies. We do need the prior distribution to be *proper*, however, in the usual sense of integrating to 1 over Θ.

Subsequently, every strategy in the Pool selects a solution and presents it to DM. At this point, DM has to select his own solution. He does this by merging or *aggregating* the solutions recommended by the strategies for that trial, considering both the solution recommendations and the current weights of the strategies. Then Reality chooses some outcome and DM can calculate his payoff at a particular trial. Then the aggregating algorithm starts over: DM updates strategy weights to reflect their performance (i.e. the larger the gain of the strategy θ the more sharply its weight increases), strategies make their selections, then DM makes his selection by merging the strategies' selections, and so on.

The algorithm has the following three parameters: a *learning rate* $\eta > 0$, a prior distribution P_0 on the pool Θ, and an *aggregation functional* Σ (Vovk uses the term "substitution function"). The learning rate $\eta > 0$, determines how fast the aggregating algorithm learns. The second parameter P_0 specifies the initial weights assigned to the strategies. The last parameter, the aggregation functional $\Sigma = \Sigma(\boldsymbol{\xi}, \mathsf{P})$, merges recommendations of the strategies $\boldsymbol{\xi}(\theta)$, $\theta \in \Theta$ according to the probability distribution $\mathsf{P}(d\theta)$ on Θ. In our analysis, we use the aggregation functional Σ which is the average with respect to P

$$\Sigma(\boldsymbol{\xi}, \mathsf{P}) = \int_{\Theta} \boldsymbol{\xi}(\theta)\mathsf{P}(d\theta); \tag{5}$$

however, one could also consider a more general form of the aggregation functional. We remark that, in the case of (5), $\Sigma(\boldsymbol{\xi}, \mathsf{P}) \in X$ is guaranteed due to the convexity of X. In general, the AA is applicable to any aggregation functional which guarantees $\Sigma(\boldsymbol{\xi}, \mathsf{P}) \in X$. The algorithm maintains current weights of the strategies in the form of the (unnormalized) measure $\mathsf{P}_t(d\theta)$ and uses a normalized measure (probability distribution) $\mathsf{P}_t^*(d\theta)$ in the aggregation functional. We summarize how the aggregating algorithm works in Figure 1.

Note that, through the operation of the AA, at time T we have:

$$\mathsf{P}_T(\Theta) = \int_{\Theta} \prod_{t=1}^{T} (\mathbf{c}_t^\top \boldsymbol{\xi}_t(\theta))^\eta \mathsf{P}_0(d\theta). \tag{6}$$

3. Analysis for the case of "fixed solution" strategies.

In this section, we consider theoretical performance of the aggregating algorithm when Pool consists of "fixed solution" strategies; that is, Pool is identified with all feasible solutions: $\Theta = X$ and $\boldsymbol{\xi}_t(\theta) = \theta$. The following technical lemma is used in the analysis and applies to any Pool of strategies (not necessarily $\Theta = X$):

Lemma 2. *For any reality outcome* $\mathbf{c} \in \Omega$*, learning rate* $0 < \eta \leq 1$*, probability measure* P*, and pool of strategies* Θ*,*

$$(\mathbf{c}^\top \Sigma(\boldsymbol{\xi}, \mathsf{P}))^\eta \geq \int_\Theta (\mathbf{c}^\top \boldsymbol{\xi}(\theta))^\eta \mathsf{P}(d\theta)\,. \tag{7}$$

The subsequent claims describe the performance of the aggregating algorithm. We denote by $\mathrm{AA}(\eta, \mathsf{P}_0, \Sigma)$ the aggregating algorithm with a learning rate $\eta > 0$, a prior distribution P_0, and a aggregation functional Σ. The analysis of the AA's performance in a variety of situations, including online portfolio selection, online statistics, prediction with expert advice, is based on the result of Vovk (see, for example [16]) which we reformulate here in terms of the geometric mean. The proposition holds for any set of strategies Θ.

Proposition 1. *For any reality outcomes* $\mathbf{c}_1, \ldots \mathbf{c}_T \in \Omega$*, learning rate* $0 < \eta \leq 1$*, probability measure* P_0*, and pool of strategies* Θ

$$(G_T(AA(\eta, \mathsf{P}_0, \Sigma)))^{\eta T} \geq \mathsf{P}_T(\Theta). \tag{8}$$

Theorem 1. *Consider a sequence of solutions produced by an aggregating algorithm (AA) with the following parameter settings applied to the pool of strategies* $\Theta = X$ *with* $\boldsymbol{\xi}_t(\theta) = \theta$*:*

- *the learning rate* $\eta \in (0, 1]$*,*
- *the probability measure* P_0 *is uniform on* X*, and*
- *the aggregation functional* Σ *is given by (5).*

Then there exists a constant $B \geq 0$ *(not depending on* η *and* T*) such that for any feasible solution* $\boldsymbol{\delta} \in X$ *and any sequence of objective coefficients* $\mathbf{c}_1, \ldots, \mathbf{c}_T \in \Omega$*,*

$$G_T\left(AA(\eta, \mathsf{P}_0, \Sigma)\right) \geq G_T(\boldsymbol{\delta}) \exp\left\{-\frac{1}{T}\left(\frac{k - 1/2}{\eta} \ln(\eta T + k) + \frac{B}{\eta}\right)\right\}, \tag{9}$$

where k *is the number of vertices of* X*.*

Moreover, the optimal learning rate $\eta = 1$*.*

An interpretation of the above result is that under the assumptions of Theorem 1, the sequence of solutions produced by the aggregating algorithm achieves long-term (geometric) time-average performance which is as good as the long-term (geometric) time-average performance of any feasible solution in X.

The resulting quotient in the performance guarantee is of the form $1-O\left(\ln(T+k)/T\right)$ and has a more favorable dependence on T than the $O(1/\sqrt{T})$ term in the additive performance bounds of [18] and [13].

Suppose that the cost vector does not change from one stage to the next. While the learner observes the cost vector after each stage, it is impossible for him to know whether the cost vector will change in the future. We now show that when the cost vector is constant but this is unknown, the values of solutions provided by the Aggregating Algorithm converge to the true optimum.

Again, let the pool of strategies consist of all feasible solutions, i.e. $\Theta = X$, and the prior distribution P_0 be uniform. We denote the constant value of the cost vector by $\mathbf{c}$. Then the stage t selection of the aggregating algorithm is

$$\mathbf{x}_t = \frac{\int_X \mathbf{x}(\mathbf{c}^\top\mathbf{x})^{\eta t}\mathsf{P}_0(d\mathbf{x})}{\int_X (\mathbf{c}^\top\mathbf{x})^{\eta t}\mathsf{P}_0(d\mathbf{x})},$$

and the cost value for this selection is

$$\mathbf{c}^\top\mathbf{x}_t = \frac{\int_X (\mathbf{c}^\top\mathbf{x})(\mathbf{c}^\top\mathbf{x})^{\eta t}\mathsf{P}_0(d\mathbf{x})}{\int_X (\mathbf{c}^\top\mathbf{x})^{\eta t}\mathsf{P}_0(d\mathbf{x})}.$$

Let $z = \max_{\mathbf{x}\in X} \mathbf{c}^\top\mathbf{x}$. We show the following:

Proposition 2. $\lim_{t\to\infty} \mathbf{c}^\top\mathbf{x}_t = z$.

Acknowledgment

The research of the second and the third authors was supported by Natural Sciences and Engineering Research Council of Canada (grant numbers 261512-04 and 138093-2001).

References

[1] M. Abramowitz and I. Milton, editors. *Handbook of mathematical functions, with formulas, graphs and mathematical tables.* Applied Mathematics Series, Vol. 55. United States Department of Commerce, Washington, D.C., 1970.

[2] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Oper. Res. Lett.*, 25(1):1–13, 1999.

[3] A. Blum and C. Burch. On-line learning and the metrical task system problem. *Machine Learning*, 39:35–58, 2000.

[4] A. Borodin and R. El-Yaniv. *Online computation and competitive analysis.* Cambridge University Press, 1998.

[5] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth. How to use expert advice. *Journal of the Association for Computing Machinery*, 44:427–285, 1997.

[6] T. Cover. Universal portfolios. *Mathematical Finance*, 1:1–29, 1991.

[7] G. Dantzig. Linear programming under uncertainty. *Management Sci.*, 1:197–206, 1955.

[8] A. Fiat and G. Woeginger, editors. *Online algorithms: The state of the art*, volume 1442 of *Lecture Notes in Computer Science*. Springer, 1998.

[9] Y. Freund and R. Schapire. Adaptive game playing using multiplicative weights. *Games Econom. Behav.*, 29(1-2):79–103, 1999.

[10] M. Grötschel, S. Krumke, and J. Rambau, editors. *Online optimization of large scale systems*. Springer, 2001.

[11] G. Hardy, J. Littlewood, and G. Polya. *Inequalities*. Cambridge University Press, Cambridge, 1967.

[12] M. Herbster and M. K. Warmuth. Tracking the best expert. *Journal of Machine Learning*, 32:151–178, 1998.

[13] A. Kalai and S. Vempala. Efficient algorithms for online decision problems. In *Proceedings of the 16th Annual Conference on Computational Learning Theory (COLT'03)*, pages 26–40, 2003.

[14] T. Levina. *Using the aggregating algorithm for portfolio selection*. PhD thesis, Rutgers University, 2004.

[15] T. Levina, Y. Levin, J. McGill and M. Nediak. *Online Appendix to "Linear Programming with Online Learning"*. http://web.business.queensu.ca/faculty/ylevin/papers/LP-learning-Appendix.pdf.

[16] V. Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, 35:247–282, 1999.

[17] V. Vovk. Private communications, 2005.

[18] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML'03)*, pages 928–936, 2003.

Aggregating Algorithm

Inputs: Learning rate $\eta > 0$, measure P_0 on Θ, aggregation functional Σ

FOR $t = 1, 2, \ldots$

Strategy Selections: Receive strategy selections $\boldsymbol{\xi}_t(\theta)$, $\theta \in \Theta$

DM Selection: Select $\mathbf{x}_t$ by applying the aggregation functional Σ

$$\mathbf{x}_t := \Sigma(\boldsymbol{\xi}_t, \mathsf{P}^*_{t-1}), \quad \text{where} \quad \mathsf{P}^*_{t-1}(d\theta) = \frac{\mathsf{P}_{t-1}(d\theta)}{\mathsf{P}_{t-1}(\Theta)}$$

Reality outcome: Receive true outcome $\mathbf{c}_t$

Strategies' Weights Update:

$$\mathsf{P}_t(d\theta) := (\mathbf{c}_t^\top \boldsymbol{\xi}_t(\theta))^\eta \mathsf{P}_{t-1}(d\theta)$$

END FOR

Figure 1. Aggregating Algorithm