the second international workshop on randomization and approximation techniques in computer science (RANDOM'98), Barcelona, Spain, pp 232–247
14. Nonner T (2011) Clique clustering yields a PTAS for max-coloring interval graphs. In: Proceedings of the 38th international colloquium on automata, languages and programming (ICALP2011), Zurich, Switzerland, pp 183–194
15. Pemmaraju S, Raman R, Varadarajan KS (2011) Max-coloring and online coloring with bandwidths on interval graphs. ACM Trans Algorithms 7(3):35
16. Trotter WT (2014) Current research problems: first fit colorings of interval graphs. http://people.math.gatech.edu/~trotter/rprob.html

Online Learning and Optimization

Yaoliang Yu
Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA, USA

Keywords

Convex programming; Hannan consistency; Online learning; Regret; Subgradient descent

Years and Authors of Summarized Original Work

2003; Zinkevich

Problem Definition

Suppose we are going to invest in a stock market. Our neighbor, for mysterious reasons, happens to know how the market evolves. But he cannot change his portfolio (proportions of holding stocks) once committed (to avoid being caught by regulators, say). On the other hand, we, the normal investor, do not have any inside information but can sell and buy at will. If we and our prescient neighbor invest the same amount of money, is there a (computationally feasible) way for us to perform comparably well to our neighbor, without knowing his investing strategy? Surprisingly (as contrary to our real-life experience perhaps), the answer is yes, and we will see it through the lens of online learning. Disclaimer: The reader is at his own risk if he decides to practice the beautiful theoretical results we describe below.

The online learning problem is best described as a multi-round two-person game between the "learner" and the "environment," following the protocol:

The Online Learning Protocol

```
For $t = 1, \dots, T$
    Learner predicts $x_t \in D$;
    Environment responds with a cost function $f_t : D \to \mathbb{R}$;
    Learner suffers an immediate cost $f_t(x_t)$;
    Learner learns some information of $f_t$.
```

Through the multi-round interactions with the environment, the learner tries to learn the behavior of the environment so as to minimize its cumulative cost in the time horizon $t \in [1, T]$, where we could allow the game to continue indefinitely, i.e., $T = \infty$.

The online learning framework is particularly relevant in real applications where (1) *sequential* decisions are needed, (2) *average* good performance is desired, and (3) the process is too *complicated* to be modeled statistically. In our stock example above, the learner will be us (normal stock holder), and the environment will be the market. Each day we submit our portfolio x_t, carefully constructed based on the past information and perhaps also mingled with some randomness (coin tosses for luck). The market responds with rises and falls of the stock prices, represented as the cost function f_t. We suffer the loss $f_t(x_t)$ and learn something about the market (e.g., f_t), and the life moves on to the next day. (If it feels more comfortable, one can negate f and call it *gain*. We shall not do this, because "a true warrior faces her bleak life bravely.") Our adventure ends at day T, which is prefixed. (For $T = \infty$, the adventure never ends.) Of course, the goal is to earn *on average* as much money as possible; it is OK if we lose occasionally. Also, for an average person (us), it is perhaps too complicated to have a clear idea what is exactly going on in that stock market. As mentioned, we would like to compete against our "prescient neighbor." This is formalized as the regret below.

To evaluate the performance of the learner, the following notion of *regret* plays a central role:

$$\mathsf{R}_T(x) := \sum_{t=1}^{T} \Big(f_t(x_t) - f_t(x) \Big), \ \forall x \in C \subseteq D. \tag{1}$$

Intuitively, the learner compares itself with the baseline (e.g., the "prescient" neighbor) that *constantly* predicts $x \in C$ in each round. We are interested in bounding the learner's regret with respect to the "best" competitor in the set C (although our notation drops the dependence on C):

$$\mathsf{R}_T := \sup_{x \in C} \mathsf{E}(\mathsf{R}_T(x)), \tag{2}$$

where the expectation $\mathsf{E}(\cdot)$ is taken with respect to any internal randomization the learner or the environment might use. The learner is said to be (Hannan) consistent if

$$\frac{\mathsf{R}_T}{T} \to 0, \text{ as } T \to \infty, \text{ i.e., } \mathsf{R}_T = o(T).$$

In other words, the learner performs, on average, as well as the best constant competitor in the long run.

We adopted the notion of regret *not* because we believe a constant (unchanging) predictor is the best strategy for our problem. Instead, the regret should be interpreted as a bare minimum requirement: If there does exist a constant predictor that performs reasonably well on our task, it would be unacceptable if our algorithm is not even on par with it. More often than not, we would like to do *better* than any constant predictor, but this can be highly nontrivial (either computationally or statistically).

We have allowed the learner to operate on a larger set D than its "competitors" (which are restricted to C). Of course this buys the learner some advantage, which sometimes is necessary for consistency, particularly when C is a nonconvex set. For instance, consider the game where the sets $C = D = \{0, 1\}$ and the cost functions

$$f_t(x) = \begin{cases} 1, & \text{if } x = x_t \\ 0, & \text{otherwise} \end{cases}. \tag{3}$$

Recall that in our online learning protocol, we have no control on how the environment reacts. In the very worst case, the environment may appear to be completely "hostile." For instance, the cost function f_t in (3) is thus defined to make the learner always suffer unit cost in each round. On the other hand, the best constant competitor in C suffers cost at most $T/2$ in T rounds. Hence, $\frac{\mathsf{R}_T}{T} \geq \frac{1}{2}$ for all T, meaning that any learner that follows our protocol cannot be consistent. The lesson is, of course, that we cannot compete under a very adversarial environment. However, if we allow the learner to *randomize* its decisions and correspondingly pay *expected* cost, then it is again possible to devise consistent learners for this game [8], provided that the environment is oblivious, i.e., it does not adapt to the learner's randomization, thus constraining its "hostility." Intuitively, randomization and averaging *smooth* out the possible worst-case (but oblivious) reactions of the environment. This is also equivalent to allowing the learner to operate on $D = [0, 1]$, the convex hull of $C = \{0, 1\}$. Indeed, for binary x we can interpret the cost function in (3) as $f_t(x) = |x - y_t|$, where in the worse case the environment could happen to choose $y_t = 1 - x_t$ from the set C. In the randomized setting, the learner first picks $x \in D$, the convex hull of C, and then chooses 1 with probability x and 0 otherwise. Provided that the environment still chooses (however adversarial) $y_t \in C$, the *expected* cost the learner suffers is again $f_t(x) = |x - y_t|$, but this time extended to the convex domain D. The claim that there exists a consistent learner under this randomized setting follows from Theorem 2 below. Intuitively, now the learner sits in the middle ($x = 1/2$) and leans toward the better constant predictor fast enough.

The previous example shows that consistency may not always be achievable. Consequently, the interesting questions in online learning include (but are not limited to) the following:

- Identifying settings under which consistency can be achieved
- Determining the correct order of the regret tending to infinity
- Devising computationally efficient and order optimal learners

These questions heavily depend on what the learner can learn in each round. For instance, in the full information setting, the learner observes the entire cost function f_t; in the bandit setting, the learner only observes its incurred cost $f_t(x_t)$, while in the partial monitoring setting, the learner only observes some quantity related to its cost. The geometry of the decision set D and the competitor set C, as well as the structural property (such as convexity, smoothness, etc.) of the cost functions, also play a significant role. In the next section, we will consider a special case where a particularly simple algorithm known as online gradient descent suffices to achieve the optimal regret. For more complete and thorough discussions, please refer to the excellent book [3] and surveys [2, 8].

Online Convex Programming

We further simplify our online learning protocol as follows:

Online Convex Programming (on the real line)

- $D \subseteq \mathbb{R}$ is a closed convex set, with $r = \max_{x,y \in D} |x - y| < \infty$;
- $\forall t \leq T$, f_t is convex and differentiable on some open set containing D;
- The gradient is uniformly bounded: $\sup_{x \in D, t \leq T} |\nabla f_t(x)| \leq M < \infty$;
- The learner gets to observe $\nabla f_t(x_t)$ in round t.

The third condition is satisfied if each f_t is M-Lipschitz continuous, i.e.,

$$\forall x, y \in D,\ |f_t(x) - f_t(y)| \leq M \cdot |x - y|, \tag{4}$$

while the last condition is certainly met if the cost function f_t is revealed to the learner in each round. Under this setting, Zinkevich [9] first analyzed the online learner that simply follows the (projected) gradient update:

$$\forall t \geq 1,\ x_{t+1} = \mathsf{P}_D(x_t - \eta_t \nabla f_t(x_t)), \tag{5}$$

where $\eta_t \geq 0$ is a small step size that we determine later and

$$\mathsf{P}_D(x) = \operatorname*{argmin}_{y \in D} |x - y|, \tag{6}$$

is the (Euclidean) projection of x onto the closed set D, i.e., the closest point in D to x. The projection is needed since the learner's prediction x_{t+1} is restricted to the decision set D.

Before we analyze the regret of the above online gradient algorithm, let us first observe that

$$\begin{aligned}\mathsf{R}_T(x) &= \sum_{t=1}^{T} \Big(f_t(x_t) - f_t(x)\Big) \\ &\leq \sum_{t=1}^{T} \Big(\nabla f_t(x_t) \cdot (x_t - x)\Big) \\ &\leq M \sum_{t=1}^{T} |x_t - x|,\end{aligned} \tag{7}$$

where the first inequality follows from the convexity of f_t. Interestingly, the right-hand side is the worst-case regret for the special case where each f_t is a linear function, say, $w_t x_t$ for some $|w_t| \leq M$. In other words, we could have restricted the game to linear cost functions, instead of the seemingly more general convex functions.

The regret of an online learner can be bounded by analyzing its progress with respect to some *potential* function. Here we choose the familiar quadratic potential. Note that for any $x \in C \subseteq D$, clearly $\mathsf{P}_D(x) = x$; hence,

$$
\begin{aligned}
|x_{t+1} - x|^2 &= |\mathsf{P}_D(x_t - \eta_t \nabla f_t(x_t)) - \mathsf{P}_D(x)|^2 \\
&\le |x_t - \eta_t \nabla f_t(x_t) - x|^2 \\
&\le |x_t - x|^2 - 2\eta_t \nabla f_t(x_t) \\
&\quad \cdot (x_t - x) + \eta_t^2 M^2 \\
&\le |x_t - x|^2 - 2\eta_t (f_t(x_t) \\
&\quad - f_t(x)) + \eta_t^2 M^2, \qquad (8)
\end{aligned}
$$

where the first inequality follows from the 1-Lipschitz continuity of the projection $\mathsf{P}_D(\cdot)$ and the last inequality is due to the convexity of f_t. Dividing (8) by $2\eta_t$, summing the indices from $t = 1$ to $t = T$, and rearranging, we have

$$
\begin{aligned}
\mathsf{R}_T(x) &= \sum_{t=1}^{T} \Big(f_t(x_t) - f_t(x) \Big) \\
&\le \sum_{t=1}^{T} \frac{1}{2\eta_t} (|x_t - x|^2 - |x_{t+1} - x|^2) \\
&\quad + M^2 \sum_{t=1}^{T} \frac{\eta_t}{2} \qquad (9) \\
&\le \frac{1}{2\eta_1} |x_1 - x|^2 + \sum_{t=2}^{T} \left(\frac{1}{2\eta_t} - \frac{1}{2\eta_{t-1}} \right) \\
&\quad |x_t - x|^2 + M^2 \sum_{t=1}^{T} \frac{\eta_t}{2}. \qquad (10)
\end{aligned}
$$

Setting the step size η_t properly leads to our key results, summarized in the next section.

Key Results

If the horizon T is finite and known in advance, then we can use a constant step size $\eta_t \equiv \eta$. Optimizing with respect to $\eta \ge 0$ from (10) yields

Theorem 1 (e.g., [8,9]) *Let $\eta_t \equiv \eta = \frac{c}{M\sqrt{T}}$ for some constant $c > 0$; then the online gradient learner achieves sublinear regret*

$$
\begin{aligned}
\mathsf{R}_T(x) &\le M\sqrt{T} \frac{c^2 + |x - x_1|^2}{2c} \\
&\le M\sqrt{T} \frac{c^2 + r^2}{2c} \qquad (11)
\end{aligned}
$$

for the online convex programming problem.

If the horizon is not know in advance, inspired by the step size in Theorem 1, we can try setting $\eta_t = \frac{c}{M\sqrt{t}}$. Note that η_t is decreasing with respect to t. Continuing from (10):

$$
\begin{aligned}
\mathsf{R}_T(x) &\le r^2 \left(\frac{1}{2\eta_1} + \sum_{t=2}^{T} \Big(\frac{1}{2\eta_t} - \frac{1}{2\eta_{t-1}} \Big) \right) \\
&\quad + cM \sum_{t=1}^{T} \frac{1}{2\sqrt{t}}.
\end{aligned}
$$

Using integration, $\sum_{t=1}^{T} \frac{1}{2\sqrt{t}} \le \int_0^T \frac{1}{2\sqrt{t}} \mathrm{d}t \le \sqrt{T}$. Thus, we have proved

Theorem 2 (Zinkevich [9]) *Let $\eta_t = \frac{c}{M\sqrt{t}}$ for some constant $c > 0$; then the online gradient learner achieves sublinear regret (simultaneously for all T)*

$$
\mathsf{R}_T(x) \le M\sqrt{T} \frac{2c^2 + r^2}{2c} \qquad (12)
$$

for the online convex programming problem.

Comparing to Theorem 1, we only lose a constant 2 in Theorem 2, but the result now holds simultaneously for all T – a property sometimes called *anytime*. Theorems 1 and 2 not only imply the consistency of the online gradient learner but also demonstrate that $\mathsf{R}_T = \mathrm{O}(\sqrt{T})$, since the right-hand sides of (11) and (12) are independent of the competitor x. In fact, this rate is optimal, i.e., there exists an instantiation where no learner (efficient or not) can do better; see, e.g., [6]. Thanks to the convexity assumption on f_t (and the decision set D), the online gradient algorithm can be efficiently implemented if the gradient ∇f_t and the projection $\mathsf{P}_D(\cdot)$ can be efficiently computed.

Doubling Trick When the horizon T is not known in advance, we can also use the doubling trick, which divides the time into exponentially increasing phases

$$\bigcup_{i=1}^{\lceil \log_2(T+1)\rceil} \{2^{i-1}, \ldots, 2^i - 1\},$$

and on the ith phase, we use the constant step size $\eta_i = \mathrm{O}(1/\sqrt{2^{i-1}})$ suggested in Theorem 1. The overall regret is bounded by

$$\sum_{i=1}^{\lceil \log_2(T+1)\rceil} \mathrm{O}(\sqrt{2^{i-1}}) = \frac{\sqrt{2}}{\sqrt{2}-1}\mathrm{O}(\sqrt{T+1}).$$

So asymptotically we only lose a factor of $\frac{\sqrt{2}}{\sqrt{2}-1} \approx 3.41$.

Other Rates It is possible to tighten the regret rate if the cost functions are more "regular." Intuitively, this means the environment is more constrained hence can only be less adversarial. Indeed, if $f_t - \frac{\sigma}{2}|\cdot|^2$ is convex, namely, f_t is σ-strongly convex, Hazan et al. [6] showed that the online gradient learner equipped with a smaller step size $\eta_t \propto \frac{1}{\sigma t}$ suffers only logarithmic regret $\mathrm{O}(\log(T))$ – an exponential improvement compared to Theorem 2. Just like the time horizon, it is possible to achieve the same logarithmic regret without knowing the parameter σ; see [1]. Similarly, if f_t is so-called exponentially concave, a similar logarithmic regret can be achieved using a second-order Newton-type learner [6].

Extension to High Dimensions The above analysis easily extends to high dimensions. In fact, Theorems 1 and 2 hold in any abstract Hilbert space, with virtually the same proof (provided that we replace the absolute value with the Hilbert norm). The cost functions f_t need not be differentiable either; picking an arbitrary subgradient in the subdifferential $\partial f_t(x_t)$ would suffice.

Extension to Composite Functions The regret can be extended to include a penalty function g as follows:

$$\mathsf{R}_T(x) = \sum_{t=1}^{T}(f_t(x_t) + g(x_t) - f_t(x) - g(x)). \tag{13}$$

Our previous definition in (1) corresponds to the setting where $g(x) = 0$ iff $x \in D$ (otherwise the regret is set to ∞). We could simply treat $f_t + g$ as a whole and apply the online gradient algorithm without any modification. A different approach, resulting in a similar regret bound, upgrades the projection to the proximity operator (of g):

$$\mathsf{P}_g^\eta(x) = \operatorname*{argmin}_y \tfrac{1}{2\eta}|x-y|^2 + g(y), \tag{14}$$

where $\eta > 0$ is the step size to be chosen appropriately. The latter approach is not only more general but also leads to more *structured* intermediate predictions [4]. For instance, if $g(x) = \sum_i |x_i|$ is the ℓ_1 norm, then $[\mathsf{P}_g^\eta(x)]_i = \mathrm{sign}(x_i)\cdot \max\{|x_i| - \eta, 0\}$, which would be exactly zero if $|x_i|$ is small and η is large. In contrast, if we apply online gradient descent directly to $f_t + g$, we would almost never get sparse intermediate predictions.

Without Projections The online gradient learner is computationally efficient only when the projection $\mathsf{P}_D(\cdot)$ in (6) (or more generally the proximity operator in (14)) can be efficiently implemented. In some applications, this is unfortunately not the case. Instead, Hazan and Kale [5] proposed a different learner that bypasses the projection step. Basically, the learner iteratively finds the vertexes of the decision set D and then takes suitable convex combinations of them to make progress.

Connection to Stochastic Optimization The regret bound in Theorem 2 is closely related to some results in stochastic optimization, for the following problem [7]:

$$\inf_{x\in D} f(x), \text{ where } f(x) := \mathsf{E}_\xi(F(x,\xi)), \tag{15}$$

and ξ is some random variable. The stochastic (sub)gradient method is a popular iterative algorithm for optimizing (15). In each iteration, it randomly draws an independent sample ξ_t and follows the projected (sub)gradient update:

$$x_{t+1} = \mathsf{P}_D(x_t - \eta_t \nabla_x F(x_t, \xi_t)),$$

for some small step size $\eta_t \geq 0$. The similarity to the online gradient learner is apparent once we identify $f_t(x) := F(x, \xi_t)$. Thus, the regret bound in Theorem 2 implies

$$\begin{aligned}
O\left(\frac{1}{\sqrt{T}}\right) &= \sup_{x \in D} \frac{1}{T} \mathsf{E}\Big[\sum_{t=1}^{T} \big(f_t(x_t) - f_t(x)\big)\Big] \\
&= \sup_{x \in D} \frac{1}{T} \mathsf{E}\Big[\sum_{t=1}^{T} \big(F(x_t, \xi_t) - F(x, \xi_t)\big)\Big] \\
&= \mathsf{E}\left(\frac{1}{T}\sum_{t=1}^{T} f(x_t)\right) - \inf_{x \in D} f(x) \\
&\geq \mathsf{E}\left(f\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right)\right) - \inf_{x \in D} f(x),
\end{aligned}$$

provided that the random sample ξ_t is independent of x_t and $F(\cdot, \xi)$ is convex for (almost) every realization of ξ. In other words, the ergodic mean $\frac{1}{T}\sum_{t=1}^{T} x_t$ approaches, in expectation, the infimum in (15) at the rate $O(1/\sqrt{T})$.

Cross-References

▶ Multi-armed Bandit Problem

Recommended Reading

1. Bartlett PL, Hazan E, Rakhlin A (2007) Adaptive online gradient descent. In: Platt JC, Koller D, Singer Y, Roweis ST (eds) Advances in neural information processing systems 20 (NIPS). Curran Associates, Inc., Vancouver, pp 257–269
2. Bubeck S, Cesa-Bianchi N (2012) Regret analysis of stochastic and nonstochastic multi-armed bandit problems. Found Trends Mach Learn 5(1):1–122
3. Cesa-Bianchi N, Lugosi G (2006) Prediction, learning, and games. Cambridge University Press, New York
4. Duchi JC, Shalev-Shwartz S, Singer Y, Tewari A (2010) Composite objective mirror descent. In: Kalai AT, Mohri M (eds) The 23rd conference on learning theory (COLT). Haifa, pp 14–26
5. Hazan E, Kale S (2012) Projection-free online learning. In: Langford J, Pineau J (eds) The 29th international conference on machine learning (ICML), Edinburgh. Omnipress, pp 521–528
6. Hazan E, Agarwal A, Kale S (2007) Logarithmic regret algorithms for online convex optimization. Mach Learn 69:169–192
7. Nemirovski A, Juditsky A, Lan G, Shapiro A (2009) Robust stochastic approximation approach to stochastic programming. SIAM J Optim 19(4):1574–1609
8. Shalev-Shwartz S (2011) Online learning and online convex optimization. Found Trends Mach Learn 4(2):107–194
9. Zinkevich M (2003) Online convex programming and generalized infinitesimal gradient approach. In: Fawcett T, Mishra N (eds) The 20th international conference on machine learning (ICML), Washington. AAAI Press, pp 928–936

Online List Update

Shahin Kamali
David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada

Keywords

Competitive analysis; Data compression; Online computation; Self-adjusting lists

Years and Authors of Summarized Original Work

1985; Sleator, Tarjan

Problem Definition

List update is one of the classic problems in the context of online computation. The main motivation for the study of the problem is self-adjusting lists. Consider a linear list which represents a dictionary abstract data type. There are three elementary operations in the dictionary, namely, insertion, deletion, and lookup (search). To perform these operations on an item x, an algorithm needs to search for x, i.e., examine the list items, one by one, to find x. For the case of an insertion, all items should be sequentially checked to ensure that the inserted item is not already in the list. A deletion also requires finding the item that is being deleted. In this manner, all operations can be translated into a sequence of lookups or *accesses* to the items in the list. To access an item at index i, an algorithm examines