On-line variance minimization in $O(n^2)$ per trial?

Elad Hazan
IBM Almaden
650 Harry Road
San Jose, CA 95120
ehazan@cs.princeton.edu

Satyen Kale
Yahoo! Research
4301 Great America Parkway
Santa Clara, CA 95054
skale@yahoo-inc.com

Manfred K. Warmuth*
Department of Computer Science
UC Santa Cruz
manfred@cse.ucsc.edu

Consider the following canonical online learning problem with matrices [WK06]: In each trial t the algorithm chooses a density matrix $\mathbf{W}_t \in \mathcal{R}^{n\times n}$ (i.e., a positive semi-definite matrix with trace one). Then nature chooses a symmetric loss matrix $\mathbf{L}_t \in \mathcal{R}^{n\times n}$ whose eigenvalues lie in the interval $[0,1]$ and the algorithms incurs loss $\mathrm{tr}(\mathbf{W}_t\mathbf{L}_t)$. The goal is to find algorithms that for any sequence of trials have small regret against the best dyad chosen in hindsight. Here a *dyad* is an outer product $\mathbf{u}\mathbf{u}^\top$ of a unit vector $\mathbf{u}$ in $\mathcal{R}^n$. More precisely the regret after T trials is defined as follows:

$$\sum_{t=1}^{T} \mathrm{tr}(\mathbf{W}_t\mathbf{L}_t) - L^*, \quad \text{where } L^* = \inf_{\mathbf{u}:\|\mathbf{u}\|=1} \mathrm{tr}\left(\mathbf{u}\mathbf{u}^\top \mathbf{L}_{\le T}\right) \text{ with } \mathbf{L}_{\le T} = \sum_{t=1}^{T} \mathbf{L}_t.$$

Instead of choosing a density matrix $\mathbf{W}_t$, the algorithm may eigendecompose $\mathbf{W}_t$ as $\sum_i \sigma_i \mathbf{u}_i\mathbf{u}_i^\top$ and choose the eigendyad[1] $\mathbf{u}_i\mathbf{u}_i^\top$ with probability σ_i. If the loss matrix $\mathbf{L}_t$ is a covariance matrix of a random variable, then $\mathbf{u}_i^\top\mathbf{L}_t\mathbf{u}_i$ is the variance in direction $\mathbf{u}_i$ and $\mathrm{tr}(\mathbf{W}_t\mathbf{L}_t)$ the expected variance / loss with respect to $\mathbf{W}_t$.

Good regret bounds are achieved by a matrix version of the Hedge algorithm [FS97] predicting with:

$$\mathbf{W}_t = \mathbf{exp}(-\eta\mathbf{L}_{<t}) \;/\; \mathrm{tr}(\mathbf{exp}(-\eta\mathbf{L}_{<t})),$$

where $\mathbf{exp}()$ is the matrix exponential and η a nonnegative learning rate. When η is chosen as $\sqrt{2\frac{\ln n}{\widehat{L}}}$, where $\widehat{L} \ge L^*$, then the *Matrix Hedge* algorithm achieves a regret bound of $\sqrt{2}\sqrt{\widehat{L}\ln n} + \ln n$ and $\sqrt{2}$ is the best known constant before the leading $\sqrt{\widehat{L}\ln n}$ term.

Note that when the initial matrix $\mathbf{W}_1$ is the identity matrix and the loss matrices are all diagonal, then Matrix Hedge maintains a distribution over the n unit dyads $\mathbf{e}_i\mathbf{e}_i^\top$ (often called "experts" in this case) and becomes the original Hedge algorithm [FS97] written with diagonal matrices instead of probability and loss vectors. The problem with Matrix Hedge is that it takes $O(n^3)$ time per trial, because the matrix exponential is typically computed by decomposing the matrices and exponentiating the eigenvalues.

Open problem: Is there an $O(n^2)$ per trial algorithm with a regret bound of $O(\sqrt{\widehat{L}\ln n})$?
Why is this a natural problem? Note that for the standard expert setting, the running time of the essentially optimal Hedge algorithm is **linear** in the number of experts n. For the matrix version, the size of all matrices involved is n^2 and we want an $O(n^2)$ per trial algorithm.

An approach based on Follow the Perturbed Leader algorithm. For the original expert setting there is an alternative algorithm to Hedge: Add a vector $\mathbf{r} \in \mathcal{R}^n$ of perturbations to the current total loss $\ell_{<t} \in \mathcal{R}^n_{\ge 0}$ of all n experts and predicts at trial t with the expert $\mathrm{argmin}_i\, \ell_{<t,i} + r_i$ of minimum perturbed loss. When r_i is the log of a suitably chosen exponential random variable, then this *Follow the Perturbed Leader* (FPL) algorithm simulates the Hedge algorithm for experts and thus obtains essentially the optimal regret bound [KW05, Kal05].

It is natural to consider matrix versions of FPL for our matrix problem. Now the perturbation is an $n \times n$ matrix $\mathbf{R}_t$ that is added to the loss matrix $\mathbf{L}_{<t}$. Computing a best expert corresponds to finding the minimum eigendyad (i.e. the one corresponding to the minimum eigenvalue) of the perturbed matrix $\mathbf{L}_{<t} + \mathbf{R}_t$, which can be approximately done in $O(n^2)$ time. Thus *Matrix FPL* essentially takes $O(n^2)$ time provided that the perturbation matrix at trial t can be sampled in $O(n^2)$ time. This speedup would be very important because it would open the path for implementing the Matrix Exponentiated Gradient algorithm [TRW05] in $O(n^2)$ time,

*Supported by NSF grant IIS-0917397

[1] A dyad $\mathbf{u}\mathbf{u}^\top$, where $\mathbf{u}$ is a unit eigenvector.

bypassing the use of decompositions, and hence have other applications, such as the ones given in [AK07] for efficiently approximating combinatorial problems.

If $\mathbf{R}_t$ is always set to a pre-selected random perturbation matrix $\mathbf{R}$, and hence does not depend on the current loss matrix $\mathbf{L}_{<t}$, and the adversary is non-adaptive, i.e. the sequence of loss matrices is fixed in advance, then we can allow $O(n^3)$ time or greater for computing $\mathbf{R}$, because this is only a preprocessing step. In each round a minimum eigendyad can then be approximated in $O(n^2)$ time.

However if $\mathbf{L}_{<t}$ and $\mathbf{R}_t$ do not have a similar eigensystem, then $\mathbf{R}_t$ may not perturb the top eigenvalues of $\mathbf{L}_{<t}$ very much. So for achieving good regret bounds it seems necessary that the perturbation matrix $\mathbf{R}_t$ adapts to $\mathbf{L}_{<t}$. If we allow the algorithm ample $O(n^3)$ time for choosing its perturbation matrix $\mathbf{R}_t$, then it is trivial to simulate Matrix Hedge with FPL: Simply decompose $\mathbf{L}_{<t}$ in $O(n^3)$ time per trial and then add log exponential perturbations to the eigenvalues as done in [KW05, Kal05] (This corresponds to choosing $\mathbf{R}_t$ to have the same eigenbasis as $\mathbf{L}_{<t}$ with eigenvalues chosen from the log exponential distribution); finally, predict with minimum eigendyad of the perturbed loss matrix. However, this $O(n^3)$ per trial implementation of Matrix FPL is not interesting, because you might as well just use the original Matrix Hedge algorithm that requires the same time.

If we ignore the optimum dependence on the dimension, then by choosing a fixed perturbation with an exponential spectral perturbation and a randomly chosen eigenbasis, we can get an $O(n^2)$ per trial algorithm and $O(n^3)$ preprocessing time. The following theorem can be proved along the same lines as in [HKW10]:

Theorem 1 *For appropriately chosen ε, the expected regret of the algorithm given below is bounded by $O(\sqrt{\widehat{L}r\log n})$, where r is an upper bound on the rank of the loss matrices.*

1: Sample n real numbers $\sigma_1, \sigma_2, \ldots, \sigma_n$ independently from the Laplace distribution with mean 0 and scale $1/\varepsilon$, i.e. the two-sided exponential probability density function $f(x) = \frac{\varepsilon}{2}\exp(-\varepsilon|x|)$.
2: Sample a random orthogonal matrix $\mathbf{U}$ uniformly from the Haar measure.
3: Define $\mathbf{R} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^\top$, where $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$.
4: **for** $t = 1$ to T **do**
5: Let $\mathbf{W}_t = \mathbf{u}_t\mathbf{u}_t^\top$, the minimum eigendyad of the matrix $\mathbf{L}_{<t} + \mathbf{R}$.
6: Predict $\mathbf{W}_t$ and observe the actual loss matrix $\mathbf{L}_t$. Incur loss $\mathrm{tr}(\mathbf{u}_t\mathbf{u}_t^\top\mathbf{L}_t)$.
7: **end for**

Notice that this already resolves the open problem for loss matrices of rank one (or constant rank). This suggests the following direction: The so-called **"unit rule"** in the usual expert setting says that the worst possible sequence of losses for the experts in a Hedge-type algorithm are the ones where only a single expert incurs loss in each trial. If the analogous statement were true for the FPL-type algorithms suggested above, in the sense that the worst sequence of loss matrices were all rank one, then our open problem would be solved.

Unfortunately, however, it is easy to concoct examples of matrices where for a fixed perturbation matrix, the loss on a rank 2 loss matrix is more than the loss on a sequence of two rank 1 loss matrices. The unit-rule might still be true in an expected sense, but we have been unable to prove such a statement.

References

[AK07] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *STOC*, pages 227–236, 2007.

[FS97] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

[HKW10] E. Hazan, S. Kale, and M. K. Warmuth. Learning rotations with little regret. In *COLT*, 2010.

[Kal05] A. Kalai. A perturbation that makes Follow the Leader equivalent to Randomized Weighted Majority. Private communication, December 2005.

[KW05] D. Kuzmin and M. K. Warmuth. *Optimum Follow the Leader Algorithm*, volume 3559, pages 684–686. Springer Verlag, 2005.

[TRW05] K. Tsuda, G. Rätsch, and M. K. Warmuth. Matrix exponentiated gradient updates for on-line learning and Bregman projections. *Journal of Machine Learning Research*, 6:995–1018, June 2005.

[WK06] M. K. Warmuth and D. Kuzmin. Online variance minimization. In *Proceedings of the 19th Annual Conference on Learning Theory (COLT '06)*, Pittsburg, June 2006. Springer-Verlag.