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

Elad Hazan
Technion

Satyen Kale
Yahoo

Manfred K. Warmuth
UCSC

Open problem at COLT 2010
June 28, 2010

Synopsis of open problems

- Since COLT 03, **eight open problems**

Synopsis of open problems

- Since COLT 03, **eight open problems**
- **None solved**

Synopsis of open problems

- Since COLT 03, **eight open problems**
- **None solved**
- Had to solve **one myself** (this COLT)

Synopsis of open problems

- Since COLT 03, **eight open problems**
- **None solved**
- Had to solve **one myself** (this COLT)
  - with some **brilliant co-authors**

Synopsis of open problems

- Since COLT 03, **eight open problems**
- **None solved**
- Had to solve **one myself** (this COLT)
  - with some **brilliant co-authors**
- So far - **no money lost**

Canonical on-line learning problem [FS]

For trial $t = 1, \ldots,$

- Predict with a distribution $\mathbf{w}_{t-1}$ over n experts
- Receive loss vector $\ell_t \in [0,1]^n$
- Incur loss $\mathbf{w}_{t-1} \cdot \ell_t$

Canonical on-line learning problem [FS]

For trial $t = 1, \ldots,$

- Predict with a distribution $\mathbf{w}_{t-1}$ over n experts
- Receive loss vector $\boldsymbol{\ell}_t \in [0,1]^n$
- Incur loss $\mathbf{w}_{t-1} \cdot \boldsymbol{\ell}_t$

Hedge algorithm

$$w_{t,i} = \frac{e^{-\eta \ell_{<t,i}}}{\sum e^{-\eta \ell_{<t,i}}}$$

Canonical on-line learning problem [FS]

For trial $t = 1, \ldots,$

- Predict with a distribution $\mathbf{w}_{t-1}$ over n experts
- Receive loss vector $\boldsymbol{\ell}_t \in [0,1]^n$
- Incur loss $\mathbf{w}_{t-1} \cdot \boldsymbol{\ell}_t$

Hedge algorithm

$$w_{t,i} = \frac{e^{-\eta \ell_{<t,i}}}{\sum e^{-\eta \ell_{<t,i}}}$$

Regret bounds

$$\sum_t \mathbf{w}_{t-1} \cdot \boldsymbol{\ell}_t - \inf_i \boldsymbol{\ell}_{\le t,i} \quad \le \quad \sqrt{2\ell^* \ln n} + \ln n$$

Loss matrices instead of loss vectors

- Symmetric loss matrices $\mathbf{L}$
- Experts replaced by dyads $\mathbf{u}\mathbf{u}^\top$
- Loss of $\mathbf{u}\mathbf{u}^\top$ is $\mathrm{tr}(\mathbf{u}\mathbf{u}^\top\mathbf{L}) = \mathbf{u}\mathbf{L}\mathbf{u}^\top$ which variance of random variable with covariance $\mathbf{L}$ in direction $\mathbf{u}$

Loss matrices instead of loss vectors

- Symmetric loss matrices $\mathbf{L}$
- Experts replaced by dyads $\mathbf{u}\mathbf{u}^\top$
- Loss of $\mathbf{u}\mathbf{u}^\top$ is $\mathrm{tr}(\mathbf{u}\mathbf{u}^\top\mathbf{L}) = \mathbf{u}\mathbf{L}\mathbf{u}^\top$ which variance of random variable with covariance $\mathbf{L}$ in direction $\mathbf{u}$
- Uncertainty over experts is probability distribution
- Uncertainty over dyads is density matrix $\mathbf{W}$,
  i.e. eigenvalues nonnegative and sum to one

Matrix version of same problem [KW]

For trial $t = 1, \ldots,$

- Predict with an n-dimensional density matrix $\mathbf{W}_{t-1}$
- Receive loss matrix $\mathbf{L}_t$ which is symmetric and has eigenvalues in [0,1]
- Incur loss $\mathbf{W}_{t-1} \cdot \mathbf{L}_t$

Matrix version of same problem [KW]

For trial $t = 1, \ldots,$

- Predict with an n-dimensional density matrix $\mathbf{W}_{t-1}$
- Receive loss matrix $\mathbf{L}_t$ which is symmetric and has eigenvalues in [0,1]
- Incur loss $\mathbf{W}_{t-1} \cdot \mathbf{L}_t$

Matrix Hedge algorithm

$$W_t = \frac{\mathbf{exp}(-\eta \mathbf{L}_{<t})}{\mathrm{tr}(\mathbf{exp}(-\eta \mathbf{L}_{<t}))}$$

Matrix version of same problem [KW]

For trial $t = 1, \ldots,$

- Predict with an n-dimensional density matrix $\mathbf{W}_{t-1}$
- Receive loss matrix $\mathbf{L}_t$ which is symmetric and has eigenvalues in [0,1]
- Incur loss $\mathbf{W}_{t-1} \cdot \mathbf{L}_t$

Matrix Hedge algorithm

$$W_t = \frac{\mathbf{exp}(-\eta \mathbf{L}_{<t})}{\mathrm{tr}(\mathbf{exp}(-\eta \mathbf{L}_{<t}))}$$

Regret bounds

$$\sum_t \mathrm{tr}(\mathbf{W}_{t-1}\mathbf{L}_t) - \inf_{\mathbf{u}} \mathbf{u}^\top \mathbf{L}_{\leq T} \mathbf{u} \quad \leq \quad \sqrt{2\ell^* \ln n} + \ln n$$

Expert setting recovered when all matrices $\mathbf{L}_t$ are diagonal

The issue of time

Matrix Hedge implementation requires eigendecomposition of current total loss matrix $\mathbf{L}_{<t}$:

	input size	update time
Vector Hedge	n	$O(n)$
Matrix Hedge	n^2	$O(n^3)$

The issue of time

Matrix Hedge implementation requires eigendecomposition of current total loss matrix $\mathbf{L}_{<t}$:

	input size	update time
Vector Hedge	n	$O(n)$
Matrix Hedge	n^2	$O(n^3)$

OPEN: Is there an algorithm for the matrix case with

- $O(n^2)$ **update time**
- **Regret remains** $\sqrt{2\ell^* \ln n} + \ln n$**?**

When rank of loss matrices small, then many tricks

Can we use FPL?

Vector case

- Add random loss vector $\mathbf{r}$ to $\ell_{<t}$
- Predict with $\text{argmin}_i(\ell_{<t,i} + \mathbf{r})$ [KV]
- With proper choice of $\mathbf{r}$, FPL simulates Vector Hedge: [K,KW]

$$\mathbf{E}(\operatorname*{argmin}_i \ell_{<t,i} + \mathbf{r}) = \frac{e^{-\eta \ell_{<t,i}}}{\sum e^{-\eta \ell_{<t,i}}}$$

Can we use FPL?

Vector case

- Add random loss vector $\mathbf{r}$ to $\ell_{<t}$
- Predict with $\text{argmin}_i(\ell_{<t,i} + \mathbf{r})$ [KV]
- With proper choice of $\mathbf{r}$, FPL simulates Vector Hedge: [K,KW]

$$\mathbf{E}(\operatorname*{argmin}_i \ell_{<t,i} + \mathbf{r}) = \frac{e^{-\eta \ell_{<t,i}}}{\sum e^{-\eta \ell_{<t,i}}}$$

Matrix case:

- Add random loss matrix $\mathbf{R}$ to $\mathbf{L}_{<t}$
- Predict with $\text{argmin}_{\mathbf{u}}\, \mathbf{u}^\top(\mathbf{L}_{<t} + \mathbf{R})\mathbf{u}$
- $\text{argmin}_{\mathbf{u}}\, \mathbf{u}^\top(\mathbf{L}_{<t} + \mathbf{R})\mathbf{u}$ is eigenvector with minimum eigenvalue
- Takes $O(n^2)$ time to compute argmin

What $\mathbf{R}$?

- Decompose $L_{<t} = \mathbf{UDU}^\top$
- Add right vector perturbation $\mathbf{r}$ to diagonal, i.e. choose $\mathbf{R} \; = \; \mathbf{U}\,\mathrm{diag}(\mathbf{r})\,\mathbf{U}^\top$:

$$\mathbf{L}_{<t} + \mathbf{R} = \mathbf{U}\,(\mathbf{D} + \mathbf{r})\,\mathbf{U}^\top$$

What $\mathbf{R}$?

- Decompose $L_{<t} = \mathbf{UDU}^\top$
- Add right vector perturbation $\mathbf{r}$ to diagonal,
  i.e. choose $\mathbf{R} = \mathbf{U}\,\mathrm{diag}(\mathbf{r})\,\mathbf{U}^\top$:

$$\mathbf{L}_{<t} + \mathbf{R} = \mathbf{U}\,(\mathbf{D} + \mathbf{r})\,\mathbf{U}^\top$$

  Can simulate Matrix Hedge
- Regret $\sqrt{2\ell^* \ln n} + \ln n$
- But $O(n^3)$ time to choose $\mathbf{R}$
- No advantage over Matrix Hedge **:-(**

A suboptimal choice of $\mathbf{R}$

- Pick $\mathbf{R} = \mathbf{U}\ \mathrm{diag}(\mathbf{r})\ \mathbf{U}^\top$,
  - where $\mathbf{U}$ is random orthogonal
  - $\mathbf{r}$ exponentially distributed entries
- $O(n^3)$ preprocessing
- Use same perturbation $\mathbf{R}$ in each trial
  - $O(n^2)$ per trial
- Suboptimal regret $\sqrt{\ell^* n}$

Open problem

Is there a different per trial choice of random matrix R such that

- **R can be found in $O(n^2)$ time**
- **Predict with minimum eigendirection of perturbed total loss matrix**
- **Regret remains $\sqrt{2\ell^* \ln n} + \ln n$**

Open problem

Is there a different per trial choice of random matrix R such that

- **R can be found in $O(n^2)$ time**
- **Predict with minimum eigendirection of perturbed total loss matrix**
- **Regret remains $\sqrt{2\ell^* \ln n} + \ln n$**

What is needed:

- Perturbation matrix **R** must "listen" to spectrum of total loss matrix
- But no time to decompose

Would speed up all applications of Matrix Exponentiated Gradient algorithm

$100 for solving it