Online Semidefinite Programming

Noa Elad

Technion

Joint work: Satyen Kale and Seffi Naor

Talk Outline

- Semidefinite programming
  - Online semidefinite programming
- Back to basics - Set Cover
  - Online set cover
  - Generalizing set cover to an O.S.P
- Additional results
  - O.S.P. with Box constraints
  - Randomized rounding

Positive Semidefinite Matrices

- $A \in SYM^{m \times m}$ is *positive semidefinite* $A \succcurlyeq 0$ iff

$$v^t A v \geq 0 \quad \forall v \in \mathbb{R}^m$$

  - Equivalently: all eigenvalues of A are ≥ 0

- Partial order: $A \succcurlyeq B$ iff $A - B \succcurlyeq 0$

- $A \succcurlyeq 0$ iff $A \cdot B = \sum_{i,j} A_{i,j} B_{i,j} \geq 0$ for all $B \succcurlyeq 0$

- $A \succcurlyeq C$ iff $A \cdot B \geq C \cdot B$ for all $B \succcurlyeq 0$

Semidefinite programming

- Semidefinite program and its dual

$$\begin{aligned} &\min cx \\ s.t.\ &A_1x_1 + \cdots + A_nx_n \succcurlyeq B \\ &x \geq 0 \end{aligned} \qquad \begin{aligned} &\max Y \cdot B \\ s.t.\ &A_i \cdot Y \leq c_i \ \ \forall i \\ &Y \succcurlyeq 0 \end{aligned}$$

- Weak Duality

$$cx = \sum_i c_i x_i \geq \sum_i (A_i \cdot Y) x_i = \left(\sum_i A_i x_i \right) \cdot Y \geq B \cdot Y$$

Y feasible
x non-negative

x feasible
Y P.S.D

Our work: Online Semidefinite Programming

- Constraint matrix B arrives online

$$\begin{aligned} &\min cx \\ s.t.\ &A_1x_1 + \cdots + A_nx_n \succcurlyeq B_t \\ &x \geq 0 \end{aligned} \qquad \begin{aligned} &\max Y \cdot B_t \\ s.t.\ &A_i \cdot Y \leq c_i \ \ \forall i \\ &Y \succcurlyeq 0 \end{aligned}$$

- Evolution of matrix B: $B_t \succcurlyeq B_{t-1}$
- Irrevocability requirement: x, Y can only increase

Online Algorithms

- Standard (offline) algorithms:
  - Input arrives all at once
  - Compute an optimal solution
  - Objective: minimize running time
- Online algorithms:
  - Input arrives iteratively
  - Compute a partial solution
  - Decisions are irrevocable
  - Objective: minimize competitive ratio
  - Running time usually not a strong consideration

Competitive ratio

$$Competitive\ Ratio = \max_{Input\ I}{}^{*} \frac{Online\ Algorithm(I)}{OPT(I)}$$

Worst-case over all inputs

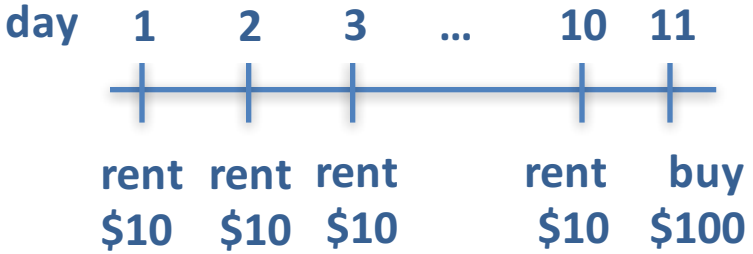
*W.L.o.G. minimization problem

Example: Ski Rental Problem

- Length of ski vacation is unknown.
- Each day, decide between:
  - Ski rental $10 per day
  - Ski purchase $100 one time
- Objective: minimize cost.
- Optimal offline: purchase iff vacation is longer than 10 days.
- What competitive ratio can we achieve?

Example: Ski Rental Problem

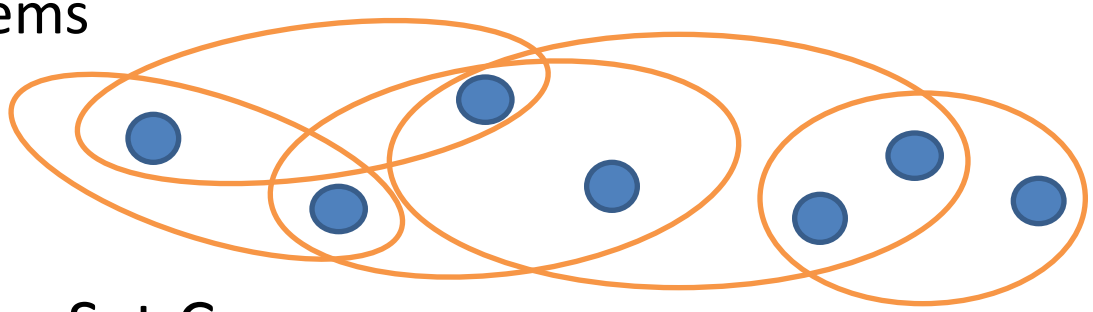
- ***Algorithm:***
  - If today <= day 10: **rent**
  - Otherwise: **buy**



- ***Analysis:***
  - If vacation ends by day 10, we paid the same as OPT.
  - If vacations ends after day 10, we spent $200 = 2*OPT.
  - **CR = 2** (worst case).
- Better ratios can be achieved ($e/e-1 \approx 1.58$)

Stepping back: Online Set Cover

- Classical Set Cover
  - Input: collection of items $1, \ldots, n$ and groups containing those items (a subset of 2^n), with costs.
  - Output: min-cost collection of sets needed to cover all items



- Online Set Cover
  - Items arrive one at a time
  - When new item arrives – must choose a set that contains it (if not already covered).
  - Sets that were chosen cannot be unchosen.

Linear Programming

- Finite set of variables: $x_1, x_2, \ldots, x_n$
- Linear objective function:

$$\min c_1 x_1 + \cdots + c_n x_n \qquad \min cx$$

- Finite number of linear constraints:

$$a_{1,1} x_1 + \cdots + a_{1,n} x_n \geq b_1$$

$$\vdots \qquad Ax \geq b$$

$$a_{m,1} x_1 + \cdots + x_{m,n} x_n \geq b_m$$

- Non-negativity constraints

$$x_i \geq 0 \quad \forall i \in [n] \qquad x \geq 0$$

Duality

- Primal and dual problem pair:

$$(\boldsymbol{P}) \quad \begin{aligned} &\min cx \\ &s.t. \;\; Ax \geq b \\ &x \geq 0 \end{aligned} \qquad\qquad (\boldsymbol{D}) \quad \begin{aligned} &\max by \\ &s.t. A^T y \leq c \\ &y \geq 0 \end{aligned}$$

- Weak Duality:

$$(\boldsymbol{P}) \geq (\boldsymbol{D})$$

- Strong Duality:

$$\boldsymbol{OPT}(\boldsymbol{P}) = \boldsymbol{OPT}(\boldsymbol{D})$$

Generalizing Set Cover to Semidefinite Covering

2 elements, 3 sets:

$$\min c_1 x_1 + c_2 x_2 + c_3 x_3$$

$$s.t. \quad x_1 + x_2 \geq 1$$

$$x_1 + x_3 \geq 1$$

Generalizing Set Cover to Semidefinite Covering

2 elements, 3 sets:

$$\min c_1 x_1 + c_2 x_2 + c_3 x_3$$

$$s.t. \quad 1x_1 + 1x_2 + 0x_3 \geq 1$$

$$1x_1 + 0x_2 + 1x_3 \geq 1$$

Generalizing Set Cover to Semidefinite Covering

2 elements, 3 sets:

$$\min c_1 x_1 + c_2 x_2 + c_3 x_3$$

$$s.t. \quad \begin{pmatrix} 1x_1 + 1x_2 + 0x_3 \geq 1 \\ 1x_1 + 0x_2 + 1x_3 \geq 1 \end{pmatrix} \geq \begin{pmatrix} \\ \end{pmatrix}$$

Characteristic matrix of the set

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix} x_1 + \begin{pmatrix} 1 \\ 0 \end{pmatrix} x_2 + \begin{pmatrix} 0 \\ 1 \end{pmatrix} x_3 \geq \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Generalizing Set Cover to Semidefinite Covering

Set Cover (Hypergraph covering)

$$\begin{pmatrix}1\\ \\ 1\end{pmatrix}x_1+\begin{pmatrix}1\\ \\ 0\end{pmatrix}x_2+\cdots+\begin{pmatrix}0\\ \\ 1\end{pmatrix}x_n\geq\begin{pmatrix}1\\ \\ 1\end{pmatrix}$$

Semidefinite Covering

$$A_1x_1+A_2x_2+\cdots+A_nx_n\succcurlyeq B$$

$0\preccurlyeq A_i\ \ \forall i$

B evolves online

Not neccesarily diagonal – matrices are P.S.D. (are in different bases)

Set Cover and Set Packing

Indicator if set S was chosen

$$(P) \quad \min \sum_S c_s x_s$$

Every element should be covered

$$s.t. \quad \forall e \sum_{S:e \in S} x_S \geq 1$$

$$x \geq 0$$

Relaxation from $x \in [0,1]^S$

$$(D) \quad \max \sum_e y_e$$

$$s.t. \quad \forall S \sum_{e \in S} y_e \leq c_s$$

$$y \geq 0$$

Online Set Cover and Set Packing

Online version

- Elements, e.g. primal constraints (and dual variables) arrive one by one
- Irrevocability: variables can only be increased

$$(P)\quad \min \sum_{s} c_s x_s$$

$$s.t.\quad \forall e \sum_{S:e\in S} x_S \geq 1$$

$$x \geq 0$$

$$(D)\quad \max \sum_{e} y_e$$

$$s.t.\quad \forall S \sum_{e\in S} y_e \leq c_s$$

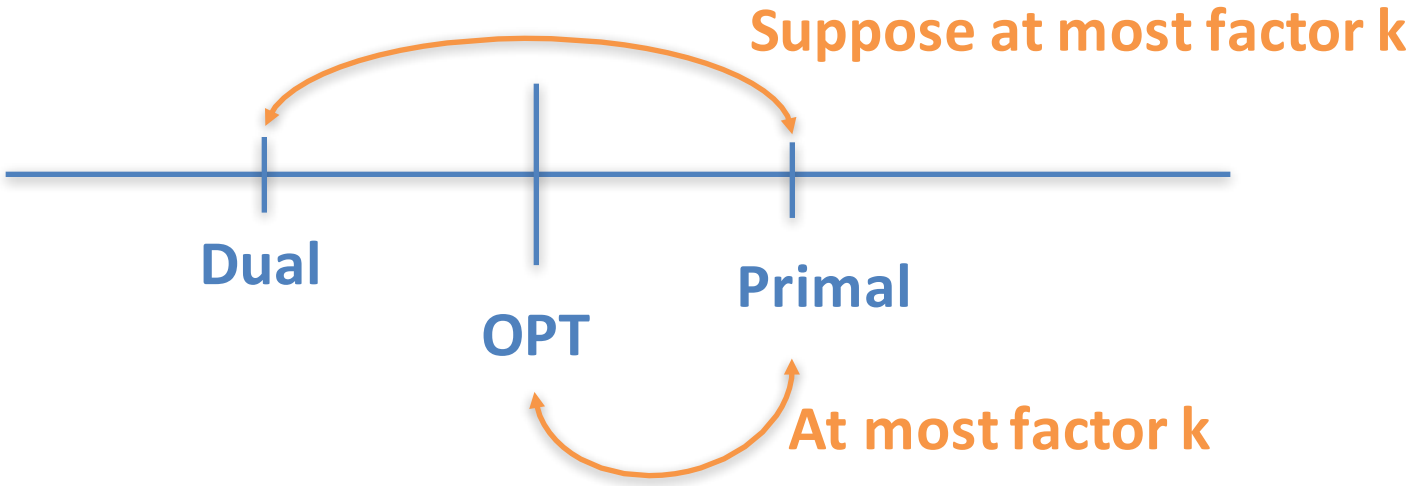
$$y \geq 0$$

Designing online algorithms using primal-dual method:

- Maintain feasible primal and dual solutions
- Update them at each online step
  - Maintain ratio between objective functions

$$P \leq kD$$

  - Then k will be the competitive ratio



Online Set Covering-Packing

$$\min \sum_S c_s x_S \quad s.t. \sum_{S:e\in S} x_S \geq 1 \ \forall e \quad x \geq 0$$

$$\max \sum_e y_e \quad s.t. \sum_{e\in S} y_e \leq c_S \ \forall S \quad y \geq 0$$

Algorithm outline:

While new element e is uncovered $\sum_{S:e\in S} x_S < 1$:

continuously increase y_e

for each S: $e \in S$ let: $x_S = x_S^{old} e^{\left(\frac{a}{c_S} y_e\right)}$

This ensures that the primal-dual ratio remains a:

Primal change $= \frac{\partial}{\partial y_e} \sum_S c_S x_S = \sum_{S:e\in S} c_S \frac{a}{c_S} x_S$

$= a \sum_{S:e\in S} x_S < a = a \cdot$ Dual change

Online Set Covering-Packing

$$\min \sum_S c_s x_S \quad s.t. \sum_{S:e\in S} x_S \geq 1 \ \forall e \quad x \geq 0$$

$$\max \sum_e y_e \quad s.t. \sum_{e\in S} y_e \leq c_S \ \forall S \quad y \geq 0$$

Feasibility:

Primal: we increase until new element is covered.

Monotonicity ⇒ once covered, will not be uncovered.

Dual: we never need to increase x_S to more than 1:

$$x_S = x_S^{init} e^{\left(\frac{a}{c_s}\sum_{e\in S} y_e\right)} \leq 1$$

$$\Rightarrow a \sum_{e\in S} y_e \leq c_s \log \frac{1}{x_S^{init}}$$

Let $x_S^{init} = 1/n$, then for $a = \log n$ we have feasibility

Online Semidefinite Programming

$$\min cx \quad s.t. \sum_i A_i x_i \succcurlyeq B_t \quad x \geq 0$$

$$\max Y \cdot B_t \quad s.t.\ A_i \cdot Y \leq c_i \ \forall i \quad Y \succcurlyeq 0$$

Algorithm outline:

While $\sum_i A_i x_i \not\succcurlyeq B_t$

find V that: $(\sum_i A_i x_i) \cdot V < B_t \cdot V$

- V is a witness to the violation of the semidefinite constraint.
  - $V = vv^t$, (v is an eigenvector of $(\sum_i A_i x_i) - B_t$)
- V defines a violated linear constraint
  - We know how to fix that :)

Online Semidefinite Programming

$$\min cx \quad s.t. \sum_i A_i x_i \succcurlyeq B_t \quad x \geq 0$$

$$\max Y \cdot B_t \quad s.t.\ A_i \cdot Y \leq c_i \ \forall i \quad Y \succcurlyeq 0$$

Algorithm outline:

While $\sum_i A_i x_i \not\succcurlyeq B_t$

find V that: $(\sum_i A_i x_i) \cdot V < B_t \cdot V$

Use V as a progress direction:

$$Y = Y^{old} + V\delta \qquad x_i = x_i^{old} e^{\left(\frac{a}{c_i} A_i \cdot V\delta\right)}$$

continuously increase δ until $(\sum_i A_i x_i) \cdot V \geq B_t \cdot V$

Online Semidefinite Programming

$$\min cx \quad s.t. \sum_i A_i x_i \succcurlyeq B_t \quad x \geq 0$$

$$\max Y \cdot B_t \quad s.t.\ A_i \cdot Y \leq c_i \ \forall i \quad Y \succcurlyeq 0$$

Algorithm outline:

$$\left(\sum_i A_i x_i\right) \cdot V < B_t \cdot V$$

$$x_i = x_i^{old} e^{\left(\frac{a}{c_i} A_i \cdot V \delta\right)}$$

Online Semidefinite Programming

$$\min cx \quad s.t. \sum_i A_i x_i \succcurlyeq B_t \quad x \geq 0$$

$$\max Y \cdot B_t \quad s.t.\ A_i \cdot Y \leq c_i \ \forall i \quad Y \succcurlyeq 0$$

Algorithm outline:

$$\left(\sum_i A_i x_i\right) \cdot V < B_t \cdot V$$

$$x_i = x_i^{old} e^{\left(\frac{a}{c_i} A_i \cdot V \delta\right)}$$

This ensures that the primal-dual ratio remains a:

Primal change $= \frac{\partial}{\partial \delta} \sum_i c_i x_i = \sum_i c_i \frac{a}{c_i} A_i \cdot V x_i$

$= a\left(\sum_i A_i x_i\right) \cdot V < a B_t \cdot V = a \cdot$ Dual change

Online Semidefinite Programming

What about feasibility?

- **Primal**: we increase until each violation we find $(\sum_i A_i x_i) \cdot V < B \cdot V$ is satisfied. Because of monotonicity, it will remain satisfied.

  (Will we eventually satisfy all possible violations?)

Online Semidefinite Programming

What about feasibility?

- **Dual**: use some boundary x^{max} on x:

$$x_i = x_i^{init} e^{\left(\frac{a}{c_i} A_i \cdot Y\right)} \leq x_i^{max}$$

$$\Rightarrow aA_i \cdot Y \leq \log \frac{x_i^{max}}{x_i^{init}} c_i$$

Let $a = \log \frac{x_i^{max}}{x_i^{init}}$ to obtain feasibility.

Online Semidefinite Programming

How to initialize?

Assume our objective is in the range $(\alpha/2, \alpha)$ ✔

Initialize: $x_i^{init} = \frac{\alpha}{2nc_i}$ $\left(\sum_i c_i x_i = \alpha/2\right)$

And we can bound x_i by $x_i^{max} = \frac{\alpha}{c_i}$

Therefore $a = \log \frac{x_i^{max}}{x_i^{init}} = \log 2n$

Online Semidefinite Programming

How to initialize?

Assume our objective is in the range $(\alpha/2, \alpha)$ ✔

Run binary search on α:

- Start with $\alpha < OPT$, for instance a trivial solution to the dual problem.
- Either we find a feasible primal solution $\leq \alpha$.
- Or we get a feasible dual solution whose value is $\geq \alpha/a$. Then we double our guess $\alpha \leftarrow 2\alpha$ and try again (we'll be off by a constant factor).

Online Semidefinite Programming

Loose end: primal feasibility

We increase until each violation we find $((\sum_i A_i x_i) \cdot V < B \cdot V)$ is satisfied.

Will we eventually satisfy all possible violations?

Not necessarily! There are infinitely many choices of V.

Our Solution

Increase so that each violation is over-satisfied: $(\sum_i A_i x_i) \cdot V \geq 2B \cdot V$.

This implies at least one variable was doubled, which can happen at most n times.

We only pay a factor 2 in the competitive ratio.

What next?

Can we do better than $O(\log n)$?

Problems

- Initializing the primal solution to a non-zero value introduces a dependence on n - the number of variables.
- But without the guess-and-double framework, we have no boundary x^{max} .
  - Needed for proving dual feasibility.

Box Constraints

Change the rules!

$$\min cx$$
$$s.t.\ A_1x_1 + \cdots + A_nx_n \succcurlyeq B_t$$
$$x \geq 0$$
$$x_i \leq u_i$$

$$\max Y \cdot B_t - \sum_{i=1}^{n} u_i z_i$$
$$s.t.\ A_i \cdot Y - u_i z_i \leq c_i \quad \forall i$$
$$Y \succcurlyeq 0$$

W.l.o.g: assume $u_i = 1$, scale A_i, c_i accordingly.

Box Constraints

$$\min cx$$
$$s.t.\ A_1x_1 + \cdots + A_nx_n \succcurlyeq B$$
$$x \geq 0$$
$$x_i \leq 1$$

$$\max Y \cdot B - \sum_{i=1}^{n} z_i$$
$$s.t. A_i \cdot Y - z_i \leq c_i \ \ \forall i$$
$$Y \succcurlyeq 0$$

Sparsity

$$F^* = \max_{t,S} \min_{V} \frac{\sum_{i \notin S} A_i \cdot V}{\left(B_t - \sum_{i \in S} A_i\right) \cdot V}$$

Box Constraints

$$\begin{array}{l} \min cx \\ s.t. \sum_{i=1}^{n} A_i x_i \succcurlyeq B_t \\ 0 \le x \le 1 \end{array} \qquad \begin{array}{l} \max Y \cdot B_t - \sum_{i=1}^{n} z_i \\ s.t.\ A_i \cdot Y - z_i \le c_i \ \forall i \\ Y \succcurlyeq 0 \end{array}$$

Sparsity

$$F^* = \max_{t,S} \min_{V} \frac{\sum_{i \notin S} A_i \cdot V}{\left(B_t - \sum_{i \in S} A_i\right) \cdot V}$$

The covering potential from unsaturated variables

A projection

Online step

Set of saturated variables: $S \subseteq [n]$

Any witness for which the denominator is positive $(B_t - \sum_{i \in S} A_i) \cdot V > 0$

The remainder left to cover

The covering we get from saturated variables

How much we "over-cover" by any set of variables

Box Constraints

$$\begin{array}{l} \min cx \\ s.t. \sum_{i=1}^{n} A_i x_i \succcurlyeq B_t \\ 0 \le x \le 1 \end{array} \qquad \begin{array}{l} \max Y \cdot B_t - \sum_{i=1}^{n} z_i \\ s.t.\ A_i \cdot Y - z_i \le c_i \ \forall i \\ Y \succcurlyeq 0 \end{array}$$

For Set-Cover

The number of sets containing j

$$F^* = \max_{t,S} \min_{V} \frac{\sum_{i \notin S} A_i \cdot V}{\left(B_t - \sum_{i \in S} A_i\right) \cdot V}$$

$(B_t - \sum_{i \in S} A_i) \cdot V > 0$

Look at witnesses $V = \mathbb{E}_{j,j}$ such that element j is uncovered

$= 1$ if j needs to be covered

$\le$ max # sets an element is contained in = row sparsity

Box Constraints

$$\begin{array}{l} \min cx \\ s.t. \sum_{i=1}^{n} A_i x_i \succcurlyeq B_t \\ 0 \leq x \leq 1 \end{array} \qquad \begin{array}{l} \max Y \cdot B_t - \sum_{i=1}^{n} z_i \\ s.t.\ A_i \cdot Y - z_i \leq c_i \ \forall i \\ Y \succcurlyeq 0 \end{array}$$

Sparsity

$$F^* = \max_{t,S} \min_{V} \frac{\sum_{i \notin S} A_i \cdot V}{\left(B_t - \sum_{i \in S} A_i\right) \cdot V}$$

We obtain an $\boldsymbol{O(\log F^*)}$ competitive algorithm.

Online Semidefinite Programming w/ Box Constraints

$$\min cx \quad s.t. \sum_{i=1}^{n} A_i x_i \succcurlyeq B_t \quad 0 \le x \le 1$$

$$\max Y \cdot B_t - \sum_{i=1}^{n} z_i \quad s.t.\ A_i \cdot Y - z_i \le c_i \ \forall i \quad Y \succcurlyeq 0$$

Algorithm outline:

While $\sum_i A_i x_i \not\succcurlyeq B$

find V that: $\sum_i A_i x_i \cdot V < B \cdot V$

$$\Rightarrow \sum_{i \notin S} A_i x_i \cdot V < B \cdot V - \sum_{i \in S} A_i \cdot V$$

$S = \{i : x_i = 1\}$

We use V as a progress direction:

$$Y = Y^{old} + V\delta$$

If $x_i \in S$: $z_i = z_i^{old} + A_i \cdot V\delta$

otherwise: $x_i + \frac{1}{F^*} = \left(x_i^{old} + \frac{1}{F^*}\right) e^{\frac{a}{c_i}(A_i \cdot V\delta)}$

+ Our definition of F^*

continuously increase δ until $\left(\sum_i A_i x_i\right) \cdot V \ge B \cdot V$

This ensures that the primal-dual ratio remains a:

Primal change $= \frac{\partial}{\partial \delta} \sum_{i \notin S} c_i x_i = \sum_{i \notin S} c_i \frac{a}{c_i} A_i \cdot V \left(x_i + \frac{1}{F^*}\right)$

$= a\left[\sum_{i \notin S} A_i x_i \cdot V + \frac{1}{F^*} \sum_{i \notin S} A_i \cdot V\right] < 2a[B \cdot V - \sum_{i \in S} A_i \cdot V] = 2a$ Dual change

Online Semidefinite Programming with Box Constraints

Feasibility

- **Primal**: same as before.
  We need a similar over-satisfying of constraints - omitted in previous slides for brevity.
- **Dual**: Since $x_i \leq 1$, we have:

$$x_i = \frac{1}{F^*}\left[e^{\left(\frac{a}{c_i}A_i \cdot Y\right)} - 1\right] \leq 1$$

$$\Rightarrow aA_i \cdot Y \leq \log(F^* + 1)\, c_i$$

Let $a = \log(F^* + 1)$ to obtain feasibility.

Randomized Rounding

- Suppose we want to round our primal solution $x \in \mathbb{R}^n$ to an integer solution $\hat{x} \in \mathbb{N}^n$.
- [Ahlswede Winter] and [Tropp] showed that a randomized solution where $\mathrm{E}[\hat{x}] = \rho x$ is feasible w.h.p., for $\rho = O(\log m)$.
- We translate this to an online randomized rounding procedure, by rounding on every increase with probability such that $\mathrm{E}[\hat{x}] = \rho x$.
- The expected cost is $\rho = O(\log m)$ times the fractional cost, i.e. we pay a factor of $O(\log m)$ for the rounding.

Tail bound for random matrix sums

Let $\{X_i\}$ be a finite sequence of independent, random, p.s.d. matrices of dimension m, such that $\lambda = \lambda_{\min}(\sum \mathrm{E}X_i)$, and $\lambda_{\max}(X_i) \leq 1$. Then for every $\mu \in [0,1]$:

$$\Pr[\lambda_{\min} \sum X_i \leq \mu\lambda] \leq m e^{-\frac{(1-\mu)^2 \lambda}{2}}$$

Explanation: w.h.p., the sum's smallest eigenvalue is not much smaller than the smallest eigenvalue of the expected sum.

Tail bound for random matrix sums

We'll show instead that:

$$\Pr[\lambda_{\min}\textstyle\sum X_i \le (1-\delta)\lambda] \le m\left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\lambda}$$

Proof:

$$\begin{aligned}
&\Pr[\lambda_{\min}\textstyle\sum X_i \le (1-\delta)\lambda] \\
&= \Pr[\lambda_{\max}\textstyle\sum -\theta X_i \ge -\theta(1-\delta)\lambda] \\
&= \Pr\left[e^{\lambda_{\max}\sum -\theta X_i} \ge e^{-\theta(1-\delta)\lambda}\right] \\
&\le e^{\theta(1-\delta)\lambda}\mathrm{E}\left[e^{\lambda_{\max}(\sum -\theta X_i)}\right]
\end{aligned}$$

By Markov's inequality

Tail bound for random matrix sums

$$\Pr[\lambda_{\min}\sum X_i \le (1-\delta)\lambda] \le m\left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\lambda}$$

Proof:

$$\begin{aligned}
&\dots \le e^{\theta(1-\delta)\lambda}\mathrm{E}\left[e^{\lambda_{\max}(\sum -\theta X_i)}\right] \\
&= e^{\theta(1-\delta)\lambda}\mathrm{E}\left[\lambda_{\max}\left(e^{(\sum -\theta X_i)}\right)\right] \\
&\le e^{\theta(1-\delta)\lambda}\mathrm{E}\left[\mathrm{tr}\left(e^{(\sum -\theta X_i)}\right)\right] \\
&\le e^{\theta(1-\delta)\lambda}\mathrm{tr}\left(\exp\sum\log\mathrm{E}\left[e^{(-\theta X_i)}\right]\right) \\
&\le e^{\theta(1-\delta)\lambda}\mathrm{tr}\left(\exp\sum\left(e^{-\theta}-1\right)\mathrm{E}[X_i]\right)
\end{aligned}$$

By Lieb's theorem and the monotonicity of matrix expectation

Matrix m.g.f.

$$\mathrm{E}\left[e^{(-\theta X_i)}\right] \preccurlyeq \exp\left(\left(e^{-\theta}-1\right)\mathrm{E}[X_i]\right)$$

Tail bound for random matrix sums

$$\Pr[\lambda_{\min}\textstyle\sum X_i \le (1-\delta)\lambda] \le m\left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\lambda}$$

Proof:

$$\ldots \le e^{\theta(1-\delta)\lambda}\operatorname{tr}\left(\exp\textstyle\sum\left(e^{-\theta}-1\right)\mathrm{E}[X_i]\right)$$

$$= \left(\frac{1}{1-\delta}\right)^{(1-\delta)\lambda} \operatorname{tr}\left(\exp\textstyle\sum -\delta\mathrm{E}[X_i]\right) \qquad \theta = -\log(1-\delta)$$

$$\le \left(\frac{1}{1-\delta}\right)^{(1-\delta)\lambda} m\lambda_{\max}\left(\exp\textstyle\sum -\delta\mathrm{E}[X_i]\right)$$

$$= \left(\frac{1}{1-\delta}\right)^{(1-\delta)\lambda} m\exp\left(\lambda_{\max}\left(\textstyle\sum -\delta\mathrm{E}[X_i]\right)\right)$$

$$= \left(\frac{1}{1-\delta}\right)^{(1-\delta)\lambda} m\exp\left(-\delta\lambda_{\min}\left(\textstyle\sum\mathrm{E}[X_i]\right)\right)$$

$$= \left(\frac{1}{1-\delta}\right)^{(1-\delta)\lambda} m\exp(-\delta\lambda) = m\left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\lambda}$$

Tail bound for random matrix sums

Assume $B = I$
Then $\sum A_i x_i \succcurlyeq I$

How to use it in our case?

$$X_i = A_i \hat{x}_i$$

$$\lambda = \lambda_{\min}\left(\sum \mathrm{E}[A_i \hat{x}_i]\right) = \rho \lambda_{\min}\left(\sum A_i x_i\right) \geq \rho$$

Then for $\mu = \frac{1}{\rho}$:

Probability that rounded solution is infeasible

$$\Pr[\lambda_{\min} \sum A_i \hat{x}_i \leq 1] \leq m e^{-\frac{\left(1-\frac{1}{\rho}\right)^2 \rho}{2}}$$

For $\rho = O(\log m)$, that is smaller than $\frac{1}{2}$.

Online Randomized Rounding

Changing our assumptions

- In order to use the tail bound, we assumed $B = I$. Actually if B is of full rank we can transform it into the identity.
- For B to have full rank throughout the online process, we allow the dimension m to increase over online steps.

Online Randomized Rounding

How to round?

- Our only assumption was: $\mathrm{E}[\hat{x}] = \rho x$.
- To achieve that:
  - Whenever x_i increases by δ, add one to $\hat{x}_i$ w.p. $\delta\rho$.

- Expected cost is simply $\rho = O(\log m)$ times the fractional cost.

Example: Max Cut

- Given a graph $G = (V, E)$, find a cut of maximum weight
- Linear relaxations only achieve factor **0.5** (maximization problem)
- Semidefinite relaxations can be rounded with ratio ≈**0.878** [Goemanns Williamson]

Online Max Cut

The max cut SDP is a packing SDP:

$$\begin{aligned} &\max \mathcal{L} \cdot Y \\ &s.t.\ Y_{ii} \leq 1 \\ &Y \succcurlyeq 0 \end{aligned}$$

$\mathcal{L}$ =Laplacian of the graph

It's dual is a covering SDP:

$$\begin{aligned} &\min \sum x_i \\ &s.t. diag(x) \succcurlyeq \mathcal{L} \\ &x \geq 0 \end{aligned}$$

Online Max Cut

$$\min \sum x_i \quad s.t. diag(x) \succcurlyeq \mathcal{L}_t \quad x \geq 0$$

$$\max \mathcal{L}_t \cdot Y \quad s.t.\ Y_{ii} \leq 1 \quad Y \succcurlyeq 0$$

We can find an online relaxed solution using our general framework.

Rounding offline is based on Choleski decomposition – not clear if it maintains "monotonicity" notion in online version.

Results

- **Online semidefinite programming**
  An $\boldsymbol{O(\log n)}$ competitive algorithm.

- **Online semidefinite programming *with box constraints***
  An $\boldsymbol{O(\log F^*)}$ competitive algorithm.
  F^* is the sparsity of the program.

- **Online *integer* semidefinite programming**
  An online randomized **rounding procedure**, with expected cost at most $\boldsymbol{O(\log m)}$ times the fractional solution.
  m is the dimension of the program.

Thank You!

Questions?