1 Matrix Multiplicative Update Method [Steurer 09]

Steurer has been assistant professor at all the important places: Max Plank Institute, Princeton, Microsoft Research. He's an assistant professor at Cornell. The matrix multiplicative update method is a technique for approximately solving semidefinite programs. To review, an SDP takes the following form:

$$\begin{aligned} \min C \bullet X \\ \text{st.} A_i \cdot X = b_i \forall i \\ X \succeq 0 \end{aligned}$$

The method works on a restricted form of the problem. Let $D = \frac{1}{n} I$. Let $\chi = \{X | X \succeq 0, D \bullet X = 1\}$. Let $\chi_{\geq \alpha} \subseteq \chi$ be the set of X for which

$$\begin{aligned} C \bullet X \geq \alpha \\ A_i \cdot X = b_i \forall i \end{aligned}$$

We want to find an $X \in \chi$ that is "close" to $\chi_{\geq \alpha}$. To generalize this problem to the form above, we can just conduct a binary search.

We also assume the existence of a ρ-bounded δ-separation oracle that can quickly determine, for a given $X \in \chi$, whether $X \in \chi_{\geq \alpha}$. Formally, a δ-separation oracle for $\chi_{\geq \alpha}$ is an algorithm that, given a matrix $X \in \chi$, says one of the following things:

- YES. The algorithm determines that X is close to the set $\chi_{\geq \alpha}$.
- NO. The algorithm finds a hyperplane (A, b) that separates X from $\chi_{\geq \alpha}$ by a δ margin such that $A \bullet X \leq b - \delta$ but $\forall X' \in X . X \bullet X' \geq b$.

The separation oracle is ρ-bounded if every A and b in the NO case satisfy

$$\rho D \preceq A - bD \preceq \rho D$$

.

Now for the algorithm. Start with an arbitrary $X \in \chi$. Call the oracle. If $X \in \chi_{\geq \alpha}$ then we're good. Otherwise, we know that $\chi_{\geq \alpha}$ is in the direction of A, so we can move a little bit in that direction and try again.

More formally, the algorithm produces a series $X_1, Y_1, \ldots, X_T, Y_t$ of matrices. X_t is our current guess for X at round t, and Y_t is the error at round t, shifted to be PSD. We define $Y_{<t} := Y_1 + Y_2 + \cdots + Y_{t-1}$. $Y_{<0} = 0$. Let the magical update function $E_\epsilon(Y) := \exp(\epsilon Y)/\text{tr}(\exp(\epsilon Y))$. For t from 1 to T:

1. Call the oracle on input $X_t := D^{-1/2} E_e(Y_{<t}) D^{-1/2}$.
2. If the oracle says YES, stop.
3. Otherwise, the oracle provides a δ-separating hyperplane $A_t \bullet X_t \leq b_t - \delta$.
   Set $Y_t = \frac{1}{2\rho} D^{-1/2}(A_t - b_t D + \rho D) D^{-1/2}$.

Note that X_t and Y_t are symmetric, $X_t \in \chi$, and $0 \preceq Y_t \preceq I$ for all t.

2 Proving the Runtime

We'll start with a theorem that will become useful to us later.

Theorem 1. *Let $\epsilon > 0$ be small enough and let $Y_1, \ldots, Y_T$ be a sequence in $[0, I] \subseteq M_n$, where M_n is the set of symmetric n-by-n matrices over the reals. Then*

$$\lambda_{max}(Y_{<T+1}) < (1+\epsilon)\sum_{t=1}^{T} X'_t \bullet Y_t + \frac{1}{\epsilon}\log n$$

where $X'_t = E_\epsilon(Y_{<T})$.

Proof. We know

$$e^{\epsilon\lambda_{\max}(Y_1+\cdots+Y_T)} = \lambda_{\max} e^{\epsilon(Y_1+\cdots+Y_T)}$$

by looking at the spectral decomposition of $Y_1 + \cdots Y_T$. As the trace is the sum of the eigenvalues,

$$\lambda_{\max} e^{\epsilon(Y_1+\cdots+Y_T)} \leq \mathrm{tr}(e^{\epsilon(Y_1+\cdots+Y_T)})$$

The Golden-Thompson inequality says that $\mathrm{tr}(e^{A+B}) \leq \mathrm{tr}(e^A e^B)$. Therefore:

$$\mathrm{tr}(e^{\epsilon(Y_1+\cdots+Y_T)}) \leq \mathrm{tr}(e^{\epsilon(Y_1+\cdots+Y_{t-1})} e^{\epsilon Y_t})$$

Now, $\forall x \in [0, \epsilon], x \leq 1 + (e^\epsilon - 1)x$. By diagonalization, for PSD M such that $0 \preceq M \preceq \epsilon I$, $\exp(M) \leq I + (e^\epsilon - 1)M$.

$$\mathrm{tr}(e^{\epsilon(Y_1+\cdots+Y_{t-1})} e^{\epsilon Y_t}) \leq \mathrm{tr}(e^{\epsilon(Y_1+\cdots+Y_{T-1})}(I + (e^\epsilon - 1)Y_T))$$

This can be rewritten as

$$= (1 + (e^\epsilon - 1)X_T \bullet Y_T)\mathrm{tr}(e^{\epsilon(Y_1+\cdots+Y_{T-1})})$$

We know $1 + x \leq e^x$, so this is less than

$$e^{(e^\epsilon-1)X_T\bullet Y_T}\mathrm{tr}(e^{\epsilon(Y_1+\cdots+Y_{T-1})})$$

Now, we can keep on doing this trick for each of the Y_i. In the base case, $\mathrm{tr}(e^0) = \mathrm{tr}(I) = n$. Therefore, the expression is less than

$$ne^{(e^\epsilon-1)\sum_{t=1}^T X_T \bullet Y_T}$$

Using the series approximation, $e^\epsilon - 1 = \epsilon + \epsilon^2/2 + O(\epsilon^3) < \epsilon + \epsilon^2$ for small enough ϵ. We get

$$e^{\epsilon\lambda_{\max}(Y_1+\cdots+Y_n)} < e^{\log n + (\epsilon+\epsilon^2)\sum_{t=1}^T X_T\bullet Y_T}$$

$$\lambda_{\max}(Y_1 + \cdots + Y_T) < (1+\epsilon)\sum_{t=1}^{T} X_t \bullet Y_t + \frac{1}{\epsilon}\log n$$

□

Theorem 2. *Let $\epsilon \leq \delta/2\rho$. If a ρ-bounded δ-separation oracle finds a separating hyperplane for $T \geq 2\epsilon^{-2}\log n$ iterations, then $\chi_{\geq\alpha}$ is empty.*

Proof. Let $X'_t := E_\epsilon(Y_{<t})$. We know $A_t \bullet X \leq b_t - \delta$ because the oracle found a separating hyperplane.

$$X'_t \bullet Y_t = D^{1/2}X_t D^{1/2} \bullet \frac{1}{2\rho}D^{-1/2}(A_t - b_t D + \rho D)D^{-1/2} = X_t \bullet \frac{1}{2\rho}(A_t - b_t D + \rho D) \leq \frac{1}{2} - \frac{\delta}{2\rho}$$

Using Theorem 1, this means

$$\lambda_{\max}(Y_1 + \cdots + Y_T) < (1+\epsilon)\left(\frac{1}{2} - \frac{\delta}{2\rho}\right)T + \frac{1}{\epsilon}\log n \leq \frac{1}{2}T - \left(\frac{\delta}{2\rho} - \frac{\epsilon}{2}\right)T + \frac{1}{\epsilon}\log n \leq \frac{1}{2}T$$

On the other hand, for every $X \in \chi_{\geq\alpha}$

$$Y_t \bullet D^{1/2}XD^{1/2} = \frac{1}{2\rho}(A_t - b_t D + \rho D) \bullet X \geq \frac{1}{2}$$

For symmetric A, we know $\max_X A \bullet X$ is just the maximum eigenvector of A, as long as X is PSD and has trace 1. Therefore

$$\lambda_{\max}(Y_1 + \cdots + Y_T) \geq (Y_1 + \cdots + Y_T) \bullet D^{1/2}XD^{1/2} \geq T/2$$

This contradicts the upper bound on $\lambda_{\max}$. There can be no $X \in \chi_{\geq\alpha}$; $\chi_{\geq\alpha}$ is empty. □

3 Oracle for Max Cut

To show the algorithm in a more concrete setting, let's use it to solve the classic Max Cut problem on $G = (V, E)$. We will only consider d-regular graphs. We will normalize the graph so the sum of all edge weights adds to 1. This means each of the edges has weight $2/nd$. We've seen the associated SDP. Let $X_{ij} = v_i^T v_j$.

$$\begin{aligned} \max \sum_{(i,j)\in E} \frac{1}{4}\left(\frac{2}{nd}\right)\|v_i - v_j\|^2 \\ \text{st. } X_{ii} = 1, X \succeq 0 \end{aligned}$$

Therefore, our set $\chi_{\geq\alpha}$ will include all X such that

$$\begin{aligned} \sum_{(i,j)\in E} \frac{1}{4}\left(\frac{2}{nd}\right)\|v_i - v_j\|^2 \geq \alpha \\ X_{ii} = 1, \\ X \geq 0 \end{aligned}$$

To find the max cut, we can do a binary search for the largest value of α such that $\chi_{\geq\alpha}$ is nonempty. This gives the optimal solution to the canonical SDP relaxation, to which we can apply Goemans-Williamson randomized rounding. As we established before, we can check if $\chi_{\geq\alpha}$ is empty by running the algorithm for at most $2e^{-2}\log n$ iterations. But to run the algorithm, we need an oracle.

We can express the objective function as a single Frobenious product.

$$\begin{aligned} &\max \sum_{(i,j)\in E} \frac{1}{2} g_{ij}(1 - X_{ij}) \\ &= \max \sum_{i,j\in V} \frac{1}{4} g_{ij}(1 - X_{ij}) \\ &= \max \frac{1}{4} \sum_{i,j\in V} g_{ij} - \sum_{i,j\in V} g_{ij} X_{ij} \\ &= \max \frac{1}{4}(D - G) \bullet X \\ &= \max \frac{1}{4} L \bullet X \end{aligned}$$

Here, $D = \frac{1}{n}I$, G is the adjacency matrix, and L is the Lapacian matrix, given by

$$L_{ij} = \begin{cases} \sum_k g_{ik} \text{ if } i = j \\ -g_{ij} \text{ if } i \neq j \end{cases}$$

Checking the first constraint is therefore just a matter of taking a product. The second constraint $X_{ii} = 1$ is more difficult, as we have to keep ρ bounded. We will use the following, more complicated condition.

Let $S_1 := \{i | i > 1 + \epsilon\}$, $S_2 := \{i | 1 - \epsilon \leq X_{ii} \leq 1 + \epsilon\}$, $S_3 := \{i | i < 1 - \epsilon\}$. Let D_S be D with all columns outside of S set to 0, let $d(S)$ be the fraction of nodes in S, and let $\delta = \epsilon^2$. Our oracle will return YES if $(D_{S_1} - D_{S_3}) \bullet X < d(S_1) - d(S_3) + \delta$ and $\sum_{(i,j)\in E} \frac{1}{4}\left(\frac{2}{nd}\right)\|v_i - v_j\|^2 \geq \alpha - \delta$. Otherwise, it will output NO, and return the violated constraint. All elements in $\chi_{\geq\alpha}$ satisfy our constraints, and whenever we output NO, we provide a δ-violated constraint.

Now, the oracle could output YES for $X \notin \chi_{>\alpha}$. In this case, the X we provide the oracle will have some vectors that do not have length 1. It's easy to construct an X' where the vectors v_i' do have unit length though:

$$X'_{ij} = \begin{cases} X_{ij}/\sqrt{X_{ii}X_{jj}} \text{ if } i,j \in S_2 \\ 1 \text{ if } i = j \notin S_2 \\ 0 \text{ otherwise} \end{cases}$$

We scale vectors with lengths in S_2 to have unit length, and throw away everything else, replacing vectors that are too long or short with new ones perpendicular to all other vectors. For this X', the value of the objective function is only $O(\epsilon)$ away from the value for the original X. This means we can apply Goemans-Williamson rounding on vectors in X' and still get a good cut.

Theorem 3. *If the oracle outputs YES for input X then*

$$\sum_{(i,j)\in E} \frac{1}{4}\left(\frac{2}{nd}\right)\|v_i - v_j\|^2 \leq \sum_{(i,j)\in E} \frac{1}{4}\left(\frac{2}{nd}\right)\left\|v'_i - v'_j\right\|^2 + O(\epsilon)$$

where the v_i are the vectors of X and the v'_i are the vectors of X', defined as above.

Proof. We know the constraint was satisfied, so

$$(D_{S_1} - D_{S_3}) \bullet X < d(S_1) - d(S_3) + \epsilon^2$$

We can rewrite this as

$$\sum_{i\in S_1}(x_{ii} - 1) + \sum_{j\in S_3}(1 - X_{jj}) < n\epsilon^2$$

But by the definitions of S_1 and S_2, we also know

$$\epsilon(|S_1| + |S_2|) \leq \sum_{i\in S_1}(x_{ii} - 1) + \sum_{j\in S_3}(1 - X_{jj})$$

This means $|S_1| + |S_3| \leq \epsilon * n$. With this in mind, we can divide the sum on the left hand side into three parts.

Case 1: $i, j \in S_1$**:** If nodes i and j are both in S_1, edges between them contribute at most $O(\epsilon)$ to the value of the objective function. We know $\|v_i - v_j\|^2 \leq 2(\|v_i\|^2 + \|v_j\|^2)$, so

$$\frac{2}{dn}\sum_{(i,j)\in E}\|v_i - v_j\|^2 \leq \frac{2}{dn}\cdot 2 \cdot d \cdot \sum_{S_1} X_{ii}$$

But $(1/n)\sum_{S_1} X_{ii} = D_{S_1} \bullet X$. Since the oracle returned YES, we can use the constraint on D_{S_1}.

$$D_{S_1} \bullet X \leq d(S_1) + \delta + D_{S_3} \bullet X$$

Because $D_{S_3} \bullet X$, δ and $d(S_1)$ are $O(\epsilon)$,

$$D_{S_1} \bullet X \leq d(S_1) + \delta + |S_3|/n \leq O(\epsilon)$$

Case 2: $i \in S_1, j \notin S_1$**:** If exactly one of i and j is in S_1, the term that is not in S_1 will be bounded by $1 + \epsilon$.

$$\frac{2}{dn}\sum_{(i,j)\in E}\|v_i - v_j\|^2 \leq \frac{2}{dn}\cdot 2 \cdot d \cdot \sum_{S_1}(1+\epsilon) + X_{ii}$$

But we can use the same bound on $(1/n)\sum_{S_1} X_{ii}$ as before to say this is $O(\epsilon)$.

Case 3: $i \in S_3, j \notin S_1$**:** If neither i nor j is in S_1, $\|v_i - v_j\|^2 \le 4(1+\epsilon)$. If at least one of the two is in S_3, we get

$$\frac{2}{dn} \cdot 2 \cdot d \cdot 4(1+\epsilon) \cdot |S_3| \le O(\epsilon)$$

Case 4: $i \in S_2, j \in S_2$**:** For i and j in S_2, $X_{ij} - X'_{ij} \le O(\epsilon)$ by the definition of S_2. Using the previous three cases, this meas that

$$\sum_{(i,j)\in E \text{ and } i,j \notin S_2} \frac{1}{4}\left(\frac{2}{nd}\right)\|v_i - v_j\|^2 \le O(\epsilon) + \sum_{(i,j)\in E \text{ and } i,j \in S_2} \frac{1}{4}\left(\frac{2}{nd}\right)\left\|v'_i - v'_j\right\|^2$$

$$\sum_{(i,j)\in E} \frac{1}{4}\left(\frac{2}{nd}\right)\|v_i - v_j\|^2 \le \sum_{(i,j)\in E} \frac{1}{4}\left(\frac{2}{nd}\right)\left\|v'_i - v'_j\right\|^2 + O(\epsilon)$$

□

4 Calculating the Exponential

While the update rule

$$X_t := D^{-1/2}\exp(\epsilon Y_{<t})/\mathrm{tr}(\exp \epsilon Y_{<t})D^{-1/2}$$

allows the algorithm to quickly converge on a member of $\chi_{\ge\alpha}$, it requires calculating a matrix exponential. The matrix exponential is defined as $e^X = \sum_{i=0}^{\infty} \frac{X^i}{i!}$. Computing an infinite series is difficult. We approximate the infinite series by only considering the first $r+1$ terms. Let $P(Y) := \sum_{i=0}^{r}\left(\frac{\epsilon Y}{i!}\right)^i$. Then we can use an approximation of the update rule

$$\hat{X}_t := cD^{-1/2}P(\epsilon Y_{<t})D^{-1/2}$$

where the multiplier c is chosen to make $D \bullet \hat{X}_t = 1$.

If we choose high enough r, this approximation will be exponentially close. In the scalar case, consider $x \in [0, \beta]$.

$$\left| e^x - \sum_{i=0}^{r} \frac{x^i}{i!} \right| \le \sum_{r+1}^{\infty} \frac{x^i}{i!} \le \sum_{i=r+1}^{\infty} \frac{\beta^i}{i!}$$

We know that $i! \le ((i+1)/e)^{i+1}$. Therefore, for $r \ge 2e\beta$

$$\sum_{i=r+1}^{\infty} \frac{\beta^i}{i!} \le \sum_{i=r+1}^{\infty} \left(\frac{e\beta}{r}\right)^i \le 2^{-r}$$

5 Reducing Dimensionality

When we construct a new X to test for membership in $\chi_{\ge\alpha}$, we should remember that each X_{ii} represents the dot product of two vectors, so $X = V^T V$ for some vectors V. In particular, $V = X^{1/2}$. As a further optimization, it would be nice if we could reduce the dimensions of the vectors in V. We just need the lower-dimensional approximations to have approximately the same distances between them. But we know a way to reduce dimensionality without changing the distance between vectors very much: Johnson Lindenstraus!

To remind everyone of the main Lemma in Johnson Lindenstraus, let R be a $d \times n$ matrix such that R_{ij} is an independent unit Gaussian $N(0,1)$. Then for a fixed vector $v \in R^n$, for $d = O(1/\epsilon^2 \log(1/\delta)$,

$$\Pr[(1-\epsilon)\|v\| \le \frac{\|Rv\|}{\sqrt{d}} \le (1+\epsilon)\|v\|] \ge 1-\delta$$

To approximate the vectors in V, therefore, we can use a $d \times n$ Gaussian matrix Φ, with each entry chosen independently from $N(0, 1/\delta)$. Then ΦV approximates the vectors in V in d dimensions. Then if $\hat{X}_t = cD^{-1/2}P(\epsilon Y_{<t})D^{-1/2}$, we can construct an approximate update rule for $\hat{X}'_t$:

$$\begin{aligned}
(\hat{X})'_t &:= (V'_t)^T V'_t \\
&= (\Phi V_t)^T (\Phi V_t) \\
&= cD^{-1/2}(\Phi P(\epsilon(1/2)Y_{<t}))^T(\Phi P(\epsilon(1/2)Y_{<t}))D^{-1/2} \\
&= D^{-1/2}P(\epsilon(1/2)Y_{<t})\Phi^T\Phi P(\epsilon(1/2)Y_{<t})D^{-1/2}
\end{aligned}$$