Lecture 12: Multiplicative Weight Update Method

Lecturers: Inzaghi Moniaga, Yin Huang Scribe: Tong Ling

Contents

This lecture focuses on the multiplicative weight update method in the context of the experts problem, along with a couple of applications in linear programming and semidefinite programming. At a high level, the MWU method is commonly used in settings where we make use of a fixed set of sources to achieve some goal. Rather than increasing the size of the set to gather more information, the multiplicative weights update method relies solely on iteratively reweighting the sources based on observed performance to move closer to the goal over time.

1 The Experts Problem (actual lecture content)

In this specific setting, we assume that we have access to n Experts where an Expert can be understood as a function $e_i : \mathbb{N} \to \mathbb{R}$ where $e_i(t)$ is the "prediction" that Expert e_i gives on day t. On day t, we have access to the Experts' predictions which we use to make our own prediction for the day. The correct answer is revealed at the end of the day each day. The situation now reduces to an optimization problem: how to make predictions based on the experts' advice such that over the course of T days so that the total number of mistakes you make is minimized? An obvious idea is to trust the better experts more, and vice versa. Problem is, you don't know which Experts are good to start with. But ideally you want to do as well as the best Expert in hindsight.

1.1 Binary outcomes → the majority algorithms

Imagine being someone whose job is to predict whether a certain stock will go up or down, first thing in the morning, every single day, for T days. This is a problem with binary outcomes, and so we may assume that $e_i(t) \in \{-1, 1\}$. We aim to understand the solution to this problem within the MWU framework.

1.1.1 Case 1: There is a perfect Expert → Majority

Assume that one of the n experts never makes mistakes. But again, you do not know which expert that is. The solution is referred to as the *Halving* algorithm. Each expert is weighted either 0 or 1. We start by giving all Experts weight 1. On day t, we make a prediction by taking the majority vote of the experts weighted 1. At the end of day t, we update the weight of Experts who were previously weighted 1 but made the wrong prediction on day t to 0. What it does is simply "eliminate" all Experts who have made a mistake at any point in time. Since there is a perfect Expert, the algorithm always leaves us with at least one Expert.
To summarize the multiplicative weight update rule, where $w_i(t)$ denotes the weight of the expert i on day $t \geq 2$, we have

$$w_i(t) = \begin{cases} 1 \cdot w_i(t-1) & \text{Expert } i \text{ is correct on day } t \\ 0 \cdot w_i(t-1) & \text{otherwise.} \end{cases}$$

Note that each time we make a mistake, at least half of the weight-1 experts have their weights updated to 0. One may then deduce easily that the total number of mistakes the algorithm makes, m, satisfies

$$m \leq \log_2 n.$$

1.1.2 Case 2: No perfect Experts exist → Weighted majority

Assume that all experts can make mistakes. With this assumption, we can't just eliminate an expert once they make a mistake, because then we may be left with no experts at all. Fix some $0 < \eta \leq \frac{1}{2}$. Once again, we start by assigning all experts weight 1. The update rule is given by

$$w_i(t) = \begin{cases} 1 \cdot w_i(t-1) & \text{Expert } i \text{ is correct on day } t \\ (1-\eta) w_i(t-1) & \text{otherwise.} \end{cases}$$

This time, we take the weighted majority vote, i.e., output 1 iff $\sum_{i:e_i(t)=1} w_i(t) \geq \sum_{i:e_i(t)=-1} w_i(t)$. Note similarly that each time we make a mistake, at least half the weights decrease by a factor $1-\eta$.

Theorem 1.1. *Let $m(t)$ be the number of mistakes the algorithm has made and $m_i(t)$ the number of mistakes the Expert i has made up to day t. Then, for any i,*

$$m(t) \leq 2(1+\eta) m_i(t) + \frac{2 \ln n}{\eta}.$$

Proof. Define the potential function to be $\Phi(t) = \sum_i w_i(t)$. Then, $\Phi(t) \geq w_i(t)$ for any i. It follows from the update rule that $w_i(t) = (1-\eta)^{m_i(t)}$. Moreover, for each t where we make a mistake,

$$\Phi(t+1) \leq \Phi(t) \left(\frac{1}{2} + \frac{1}{2}(1-\eta) \right) = \left(1 - \frac{\eta}{2} \right) \Phi(t),$$

because at most a half of the weights stay the same and the others decrease by a factor $1-\eta$, and since $\Phi(1) = n$, we have that

$$\Phi(t) \leq n \left(1 - \frac{\eta}{2} \right)^{m(t)}.$$

Altogether,

$$(1-\eta)^{m_i(t)} = w_i \leq \Phi(t) \leq n \left(1 - \frac{\eta}{2} \right)^{m(t)}.$$

Taking the logarithm of both sides, we have

$$m_i(t) \ln(1-\eta) \leq \ln n + m(t) \ln \left(1 - \frac{\eta}{2} \right).$$

Since $0 < \eta \leq \frac{1}{2}$, we have $\ln(1-\eta) \geq -\eta - \eta^2$, and $\ln(1-\eta/2) \leq -\eta/2$. Hence,

$$(-\eta - \eta^2) m_i(t) \leq m_i(t) \ln(1-\eta) \leq \ln n + m(t) \ln \left(1 - \frac{\eta}{2} \right) \leq \ln n - m(t) \cdot \frac{\eta}{2}$$

$$\implies m(t) \cdot \frac{\eta}{2} \leq \eta(1+\eta) m_i(t) + \ln n$$

$$\implies m(t) \leq 2(1+\eta) m_i(t) + \frac{2 \ln n}{\eta},$$

as required. □

1.2 General setting - weights as probability distributions

In this general setting, we relax the restriction of outcomes being binary. It is natural to now introduce the concept of "cost" which is analogous to making a mistake in the binary case. Suppose that the cost of expert i, $m_i(t)$, is within the range $[-\rho, \rho]$. We normalize the weights to form a probability distribution, $p(t)$, over the n experts. On day t, we simply take the expectation of the experts' prediction w.r.t. $p(t)$, where we have an expected cost of $\langle m(t), p(t)\rangle$. Now instead of minimizing the total number of mistakes, we minimize the total expected cost $\sum_{t=1}^{T}\langle m(t), p(t)\rangle$.
Fix some $0 < \eta \leq \frac{1}{2}$. We start by setting $p_i(1) = \frac{1}{n}$ for all i. The weight update rule is given by

$$w_i(t+1) = \left(1 - \eta \cdot \frac{m_i(t)}{\rho}\right) w_i(t), \quad p_i(t+1) = \frac{w_i(t+1)}{\Phi(t+1)},$$

where $\Phi(t+1) := \sum_i w_i(t+1)$.

Theorem 1.2. *After T days, for any $i \in [n]$,*

$$\sum_{t=1}^{T}\langle m(t), p(t)\rangle \leq \sum_{t=1}^{T} m_i(t) + \frac{\eta}{\rho} \cdot \sum_{t=1}^{T} |m_i(t)| + \frac{\rho \cdot \ln n}{\eta}.$$

Proof. The idea is similar to the proof of Theorem 1.1. Using the inequality $1 - x \leq e^{-x}$ and the definitions, we have

$$\begin{aligned}
\Phi(t+1) = \sum_{i=1}^{n} w_i(t+1) &= \sum_{i=1}^{n} w_i(t)(1 - \frac{\eta}{\rho} m_i(t)) \\
&= \Phi(t) - \frac{\eta}{\rho}\sum_{i=1}^{n} w_i(t) m_i(t) \\
&= \Phi(t) - \frac{\eta}{\rho}\Phi(t)\sum_{i=1}^{n} p_i(t) m_i(t) \\
&= \Phi(t)\left(1 - \frac{\eta}{\rho}\langle p(t), m(t)\rangle\right) \\
&\leq \Phi(t) e^{-\frac{\eta}{\rho}\langle p(t), m(t)\rangle}.
\end{aligned}$$

Since $\Phi(1) = n$, by induction, we have

$$\Phi(T+1) \leq n \exp\left(-\frac{\eta}{\rho}\sum_{t=1}^{T}\langle p(t), m(t)\rangle\right).$$

Using the inequalities $1 - \eta x \geq \begin{cases} (1-\eta)^x & x \in [0,1] \\ (1+\eta)^{-x} & x \in [-1,0] \end{cases}$ where $0 < \eta \leq \frac{1}{2}$, we have

$$\Phi(T+1) \geq w_i(T+1) = \prod_{t=1}^{T}\left(1 - \frac{\eta}{\rho} m_i(t)\right) \geq \left(1 - \frac{\eta}{\rho}\right)^{\sum_{\geq 0} m_i(t)} \left(1 + \frac{\eta}{\rho}\right)^{\sum_{<0} -m_i(t)}.$$

Altogether,

$$\left(1 - \frac{\eta}{\rho}\right)^{\sum_{\geq 0} m_i(t)} \left(1 + \frac{\eta}{\rho}\right)^{\sum_{<0} -m_i(t)} \leq \Phi(T+1) \leq n \exp\left(-\frac{\eta}{\rho}\sum_{t=1}^{T}\langle p(t), m(t)\rangle\right).$$

Taking logarithms, we have

$$\ln\left(1 - \frac{\eta}{\rho}\right)\sum_{\geq 0} m_i(t) - \ln\left(1 + \frac{\eta}{\rho}\right)\sum_{<0} m_i(t) \leq \ln n - \frac{\eta}{\rho}\sum_{t=1}^{T}\langle p(t), m(t)\rangle.$$

Finally, using the inequalities $\ln(1-\eta) \geq -\eta - \eta^2$ and $\ln(1+\eta) \geq \eta - \eta^2$ where $0 < \eta \leq \frac{1}{2}$, we have

$$\left(-\frac{\eta}{\rho} - \left(\frac{\eta}{\rho}\right)^2\right)\sum_{\geq 0} m_i(t) - \left(\frac{\eta}{\rho} - \left(\frac{\eta}{\rho}\right)^2\right)\sum_{<0} m_i(t) = -\frac{\eta}{\rho}\sum_{t=1}^{T} m_i(t) - \frac{\eta^2}{\rho^2}\sum_{t=1}^{T}|m_i(t)| \leq \ln n - \frac{\eta}{\rho}\sum_{t=1}^{T}\langle p(t), m(t)\rangle.$$

Rearranging we have

$$\frac{\eta}{\rho}\sum_{t=1}^{T}\langle p(t), m(t)\rangle \leq \frac{\eta}{\rho}\sum_{t=1}^{T} m_i(t) + \frac{\eta^2}{\rho^2}\sum_{t=1}^{T}|m_i(t)| + \ln n.$$

Hence,

$$\sum_{t=1}^{T}\langle p(t), m(t)\rangle \leq \sum_{t=1}^{T} m_i(t) + \frac{\eta}{\rho}\sum_{t=1}^{T}|m_i(t)| + \frac{\rho \cdot \ln n}{\eta},$$

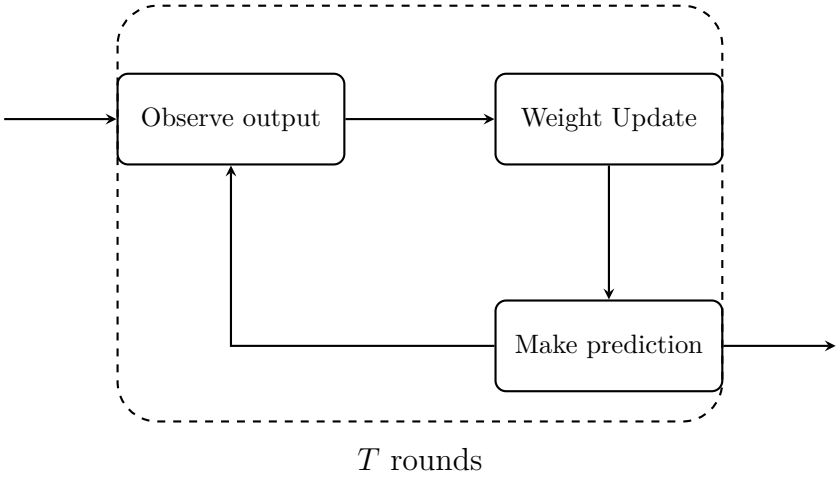
as required. □

2 Application: Linear Programming **(mentioned briefly in class)**

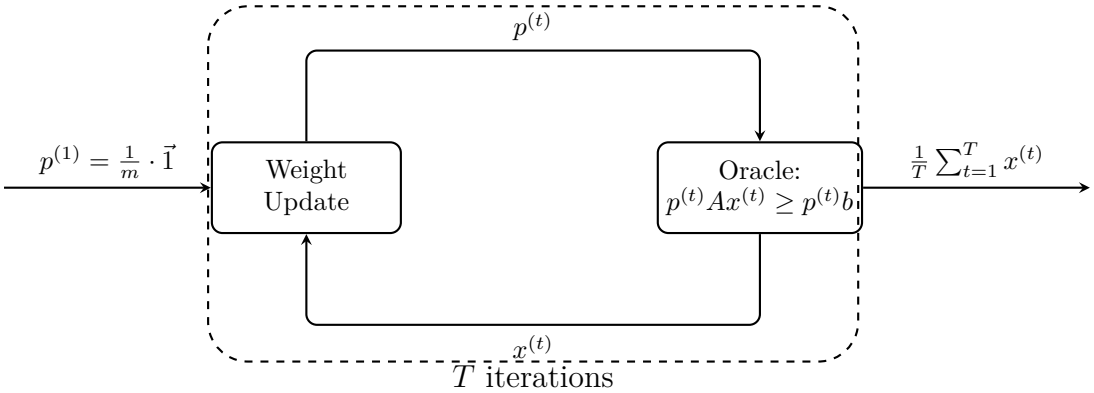
Consider the feasibility version of the linear program

$$\begin{aligned} Ax &\geq b \\ x &\geq 0. \end{aligned} \tag{1}$$

Recall that in the Experts problem, the MWU framework looks roughly like:



It turns out that we can solve the linear program using the MWU method as well. This is what the algorithm roughly looks like:



2.1 The MWU solution

For the linear program, we can think of the m constraints in the following form

$$A_i x - b_i \geq 0,$$

where A_i is the i-th row of A. We may therefore think of the m constraints as m Experts. The smaller $A_i x - b_i$ is, the larger the violation of this constraint is. And we want to put higher weights on the constraints with larger violations (i.e. those constraints will receive "more attention" in the next round), just as how we put a high weight on experts with better performance, with the hope that the violation of that specific constraint will decrease in subsequent rounds. By putting a probability distribution over the constraints, we effectively reduce the linear program m constraints to a much simpler one with only one single constraint plus the easy constraint $x \geq 0$. We then employ an oracle of width w to find a solution for this simple linear program.

2.1.1 Oracle of width w.

Recall that in the Experts problem, the cost of each expert at any time t is bounded by $[-\rho, \rho]$. Here, the oracle is responsible for returning a solution to

$$\begin{aligned} \langle p^{(t)}, Ax^{(t)} \rangle &\geq \langle p^{(t)}, b \rangle \\ x^{(t)} &\geq 0 \end{aligned} \tag{2}$$

satisfying

$$A_i x^{(t)} - b_i \leq w,$$

for all i. We refer to w as the width of the oracle. Note that if (2) is infeasible, then (1) is infeasible. Indeed, suppose for the sake of contradiction that there is a solution x to (1), then for this x, since $p_i^{(t)} \geq 0$, we have

$$\begin{gathered} A_i x \geq b_i, \forall i \implies p_i^{(t)} A_i x \geq p_i^{(t)} b_i, \forall i \implies \langle p^{(t)}, Ax \rangle \geq \langle p^{(t)}, b \rangle \\ x \geq 0 \end{gathered}$$

which means that (2) is feasible, a contradiction.

2.1.2 Weight update

As we noted above, we want to give the higher weight to a constraint with a larger violation. Let $m_i^{(t)} := \frac{A_i x^{(t)} - b_i}{w}$. Update the weight as usual

$$w_i^{(t+1)} = w_i^{(t)}(1 - \eta m_i^{(t)}).$$

Note that $m_i^{(t)} \in [-1, 1]$, which is equivalent to assuming $\rho \in [-1, 1]$. Set $p_i^{(t+1)} = \frac{w_i^{(t+1)}}{\sum_i w_i^{(t+1)}}$.

2.1.3 Violation bound

We take the average solution over T iteration as our final solution. Let $\varepsilon > 0$ be the error parameter. We want to lower bound the violation of each constraint by $-\varepsilon$ to indicate that our solution is fairly good. As we have established the MWU structure for LP, it follows directly from Theorem 1.2 that for all i,

$$\begin{aligned} \underbrace{0 \leq \sum_{t=1}^{T} \langle Ax^{(t)} - b, p^{(t)} \rangle}_{\langle p^{(t)}, Ax^{(t)} \rangle \geq \langle p^{(t)}, b \rangle} &\leq \sum_{i=1}^{T} m_i(t) + \eta \cdot \sum_{t=1}^{T} |m_i(t)| + \frac{\ln m}{\eta} \\ &\leq \frac{1}{w} \sum_{i=1}^{T} (A_i x^{(t)} - b_i) + \eta T + \frac{\ln m}{\eta} \\ \implies \frac{1}{T} \sum_{t=1}^{T} (A_i x^{(t)} - b_i) &\geq -\eta w - \frac{\ln m}{\eta T}. \end{aligned}$$

Letting $\eta = \frac{\varepsilon}{2w}$ and $T = \frac{4w^2 \ln m}{\varepsilon^2}$ we have

$$\frac{1}{T} \sum_{t=1}^{T} (A_i x^{(t)} - b_i) \geq -\varepsilon,$$

as required, i.e., the average is approximately feasible with error at most ε.

2.1.4 Runtime

It follows from Section 2.1.3 that solving the LP with the error parameter ε requires calling the oracle $O(w^2 \ln m/\varepsilon^2)$ times. Additionally, it takes $O(m+n)$ time each time we are calling the oracle given the size of the matrix is $m \times n$. Therefore, the total runtime is $O((m+n)w^2 \ln m/\varepsilon^2)$.

3 Application: Semidefinite Programming

LP makes us think of SDP, whose feasibility version is of the form

$$\begin{aligned} A_i \bullet X &\geq b_i \quad i \in [m] \\ X &\succeq 0. \end{aligned} \tag{3}$$

Definition 3.1 (inner product). The inner product of matrices A and B is defined by $A \bullet B := \text{Tr}(A^T B) = \sum_{i,j} A_{ij} B_{ij}$.

SDP is semantically analogous to LP, generalizing nonnegative vectors x to positive semi-definite matrices X. One may then wonder if we can use multiplicative weight to (approximately) solve SDP. The answer is yes, and we now look at a generalization of MWU to matrices.

3.1 Matrix Multiplicative Weight Update

Recall in MWU, we have probability distribution and cost, $p^{(t)}$ and $m^{(t)}$, as vectors. In Matrix MWU, we work with the density matrix $P^{(t)}$ which by definition is positive semi-definite and has a trace of 1, and loss matrix $M^{(t)}$, which is symmetric and has all eigenvalues in $[-1, 1]$. The quantity we want to minimize is $\sum_{t=1}^{T} P^{(t)} \bullet M^{(t)}$. Fix some $\eta \leq 1$. The update rule is given by

$$W^{(t+1)} = \exp\left(-\eta \sum_{i=1}^{t} M^{(i)}\right).$$

Set $P^{(t+1)} = \frac{W^{(t+1)}}{\text{Tr}(W^{(t+1)})}$ so that $\text{Tr}(P^{(t+1)}) = 1$. Note that since $P^{(t)}$ is PSD and $\text{Tr}(P^{(t)}) = 1$, we have

$$P^{(t)} = Q \Lambda Q^T = \sum_{i=1}^{n} \lambda_i^{(t)} v_i^{(t)} (v_i^{(t)})^T,$$

where the v_i^t are orthonormal eigenvectors corresponding to $\lambda_i^{(t)}$ with $\sum_{i=1}^{n} \lambda_i^{(t)} = 1$ and $\lambda_i^{(t)} \geq 0$. We can then think of the eigenvalues as a probability distribution and the density matrix $P^{(t)}$ as the sum of the rank-1 matrices $v_i v_i^T$ over a probability distribution.

In the matrix version, the update rule is different. We want to preserve the positive semidefiniteness of $P^{(t)}$ and work with positive potential functions ($\text{Tr}(W^{(t)})$). The old update rule ($I - \eta M^{(i)}$) doesn't give us a PSD matrix in general for a symmetric matrix $M^{(t)}$, but $\exp(-\eta M^{(t)})$ does. In fact, since $-\eta M^{(t)}$ is symmetric, $\exp(-\eta M^{(t)})$ is positive definite. This way, we have $W^{(t)}$ that is positive definite, and therefore $\text{Tr}(W^{(t)}) > 0$ and by the update rule $P^{(t)}$ is PSD.

Theorem 3.2. *For any sequence of loss matrices $M^{(1)}, \ldots, M^{(T)}$, the matrix multiplicative weight algorithm generates density matrices $P^{(1)}, \ldots, P^{(T)}$ such that*

$$\sum_{t=1}^{T} M^{(t)} \bullet P^{(t)} \leq \lambda_{\min}\left(\sum_{t=1}^{T} M^{(t)}\right) + \eta \sum_{t=1}^{T} (M^{(t)})^2 \bullet P^{(t)} + \frac{\ln n}{\eta}.$$

Moreover,

$$\sum_{t=1}^{T} P^{(t)} \bullet M^{(t)} \leq \lambda_{\min}\left(\sum_{t=1}^{T} M^{(t)}\right) + \eta T + \frac{\ln n}{\eta}.$$

Proof. Define the potential function by $\Phi(t) = \mathrm{Tr}(W^{(t)})$. The idea is still similar to the proofs of the previous theorems, except that for matrices, we don't usually have $\exp(A+B) = \exp(A)\exp(B)$, so $\mathrm{Tr}(\exp(A+B)) \neq \mathrm{Tr}(\exp(A)\exp(B))$ in general.

Fact 1 (Golden-Thompson inequality). For matrices A, B,

$$\mathrm{Tr}(\exp(A+B)) \leq \mathrm{Tr}(\exp(A)\exp(B)).$$

Note that by the defining properties of $M^{(t)}$, the spectral norm of $M^{(t)}$ satisfies $\|M\| \leq 1$.

Fact 2 (Taylor approximation). For $\|M\| \leq 1$,

$$\exp(-M) \preceq I - M + M^2.$$

Applying Fact 2 to $\eta M^{(T)}$ ($\eta \leq 1, \|M^{(T)}\| \leq 1 \implies \|\eta M^{(T)}\| \leq 1$), we have

$$\exp(-\eta M^{(T)}) \preceq I - \eta M^{(T)} + \eta^2 (M^{(T)})^2.$$

Fact 3 (Linearity of Trace). The Trace function is linear, i.e.,

$$\mathrm{Tr}(A+B) = \mathrm{Tr}(A) + \mathrm{Tr}(B), \text{ and } \mathrm{Tr}(cA) = c\mathrm{Tr}(A).$$

Fact 4 (Trace is cyclic). For matrices A, B, we have $A \bullet B = B \bullet A$.

Fact 5 For PSD matrices A, B, C, if $B \preceq C$ (which implies $\mathrm{Tr}(B) \leq \mathrm{Tr}(C)$), then $A \bullet B \leq A \bullet C$. This can be derived from Fact 4 combined with properties of PSD matrices.

By construction, $W^{(t)}$ is PSD. Since $-\eta M^{(t)}$ is symmetric, $\exp(-\eta M^{(t)})$ is PSD (in fact postive definite). Therefore, $\mathrm{Tr}(\exp(-\eta M^{(t)})) \geq 0$. Then, by linearity and Fact 5, we have

$$(*) \qquad W^{(T)} \bullet \exp(-\eta M^{(T)}) \leq W^{(T)} \bullet (I - \eta M^{(T)} + \eta^2 (M^{(T)})^2).$$

Using Golden-Thompson, $(*)$, the inequality $\exp(-x) \geq 1 - x$, and the definitions, we have

$$\begin{aligned}
\Phi(T+1) = \mathrm{Tr}(W^{(T+1)}) &= \mathrm{Tr}\left(\exp\left(-\eta \sum_{t=1}^{T} M^{(t)}\right)\right) \\
&\leq \mathrm{Tr}\left(\exp\left(-\eta \sum_{t=1}^{T-1} M^{(t)}\right) \exp\left(-\eta M^{(T)}\right)\right) \\
&= W^{(T)} \bullet \exp(-\eta M^{(T)}) \\
&\leq \underbrace{W^{(T)}}_{=\Phi(T)P^{(T)}} \bullet \left(I - \eta M^{(T)} + \eta^2 (M^{(T)})^2\right) \\
&= \Phi(T)\left(1 - \eta M^{(T)} \bullet P^{(T)} + \eta^2 (M^{(T)})^2 \bullet P^{(T)}\right) \\
&\leq \Phi(T) \exp\left(-\eta M^{(T)} \bullet P^{(T)} + \eta^2 (M^{(T)})^2 \bullet P^{(T)}\right).
\end{aligned}$$

Since $\Phi(1) = \mathrm{Tr}(I) = n$, by induction,

$$\Phi(T+1) \leq n \exp\left(-\eta \sum_{t=1}^{T} M^{(t)} \bullet P^{(t)} + \eta^2 \sum_{t=1}^{T} (M^{(t)})^2 \bullet P^{(t)}\right).$$

Denote by $\lambda_k(A)$ the k-th eigenvalue of the matrix A. Suppose A is symmetric, which means all of its eigenvalues are real. Since $\mathrm{Tr}(A) = \sum_{k=1}^{n} \lambda_k(A)$, we have

$$\mathrm{Tr}(\exp(A)) = \sum_{k=1}^{n} \exp(\lambda_k(A)) \geq \exp(\lambda_{\min}(A)).$$

We can lower bound $\Phi(T+1)$ as follows

$$\begin{aligned}\Phi(T+1) = \mathrm{Tr}(W^{(t+1)}) &= \mathrm{Tr}\left(\exp\left(-\eta\sum_{t=1}^{T} M^{(t)}\right)\right)\\ &\geq \lambda_{\max}\left(\exp\left(-\eta\sum_{t=1}^{T} M^{(t)}\right)\right)\\ &= \exp\left(\lambda_{\max}\left(-\eta\sum_{t=1}^{T} M^{(t)}\right)\right)\\ &= \exp\left(-\eta\lambda_{\min}\left(\sum_{t=1}^{T} M^{(t)}\right)\right).\end{aligned}$$

Altogether, we have that

$$\exp\left(-\eta\lambda_{\min}\left(\sum_{t=1}^{T} M^{(t)}\right)\right) \leq n\exp\left(-\eta\sum_{t=1}^{T} M^{(t)}\bullet P^{(t)} + \eta^2\sum_{t=1}^{T}(M^{(t)})^2\bullet P^{(t)}\right).$$

Taking logarithms and rearranging gives

$$\sum_{t=1}^{T} M^{(t)}\bullet P^{(t)} \leq \lambda_{\min}\left(\sum_{t=1}^{T} M^{(t)}\right) + \eta\sum_{t=1}^{T}(M^{(t)})^2\bullet P^{(t)} + \frac{\ln n}{\eta}.$$

Now since $\|M^{(t)}\| \leq 1$, we have $(M^{(t)})^2 \bullet P^{(t)} \leq I \bullet P^{(t)} = 1$. Therefore,

$$\sum_{t=1}^{T} M^{(t)}\bullet P^{(t)} \leq \lambda_{\min}\left(\sum_{t=1}^{T} M^{(t)}\right) + \eta T + \frac{\ln n}{\eta},$$

as required. □

3.2 Solution to SDP

We want to add another constraint $\mathrm{Tr}(X) \leq R$ to the SDP to bound the feasibility region. R can usually be guessed by binary search. So the SDP we are solving is

$$\begin{aligned}\mathrm{Tr}(X) &\leq R\\ A_i \bullet X &\geq b_i \quad i \in [m]\\ X &\succeq 0.\end{aligned} \tag{4}$$

Consider the following system

$$\begin{aligned}b\cdot y &> R\\ \sum_{j=1}^{m} A_j y_j &\preceq I\\ y &\geq 0.\end{aligned} \tag{5}$$

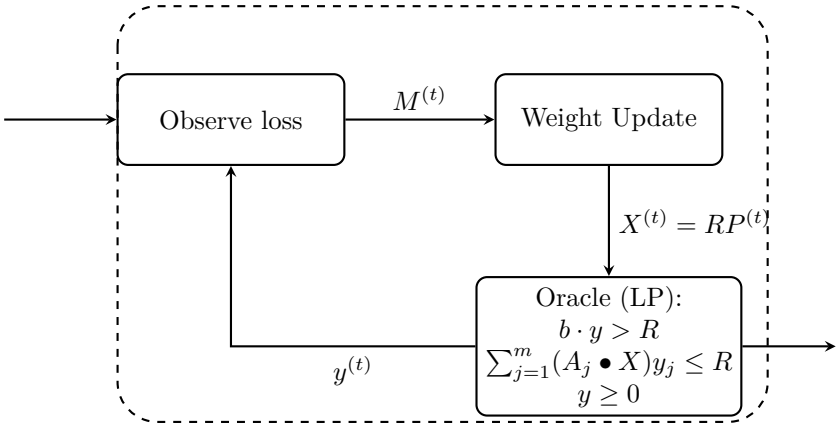
In the standard optimization form, for the primal SDP

$$\begin{aligned}\min \mathrm{Tr}(X)&\\ A_i \bullet X &\geq b_i, \quad i \in [m]\\ X &\succeq 0,\end{aligned}$$

its dual (see also [1]) is given by

$$\begin{aligned}\max b\cdot y&\\ \sum_{j=1}^{m} A_j y_j &\preceq I\\ y &\geq 0.\end{aligned}$$

By weak duality which does hold for SDP, we know that if y is a feasible solution to the dual and X is a feasible solution to the primal, then $b \cdot y \leq \text{Tr}(X) \leq R$. Now turning this into the feasibility form, by enforcing $b \cdot y > R$ in (5), we are ensuring that if (5) is feasible, then (4) is infeasible, because otherwise the weak duality is violated.



To solve the SDP using Matrix MWU, we employ an oracle that solves a linear program derived from (5). If the oracle LP returns a solution y, then the current X is not a solution to (4), because otherwise we have

$$R \geq \sum_{j=1}^{m}(A_j \bullet X)y_j \geq \sum_{j=1}^{m} b_j y_j = b \cdot y > R,$$

a contradiction. Then, we use this dual certificate y to update the weight matrix.

If we wanted to mirror what we did in the LP section, the oracle would be on the primal and it would be looking for X that satisfies

$$\begin{gathered} \text{Tr}(X) \leq R \\ \left(\sum_{j=1}^{m} p_j A_j\right) \bullet X \geq \langle p, b \rangle \\ X \succeq 0. \end{gathered}$$

This Oracle is a simplified SDP whose solution reduces to finding $\lambda_{\max}(\sum_{j=1}^{m} p_j A_j)$ and the eigenvector corresponding to it. See [3] for full treatment. In our primal-dual approach ([1]), the oracle is a linear program with two nontrivial constraints. Both are significantly easier than solving a full SDP.

3.2.1 Oracle of width ρ

As usual, the oracle comes with a width, ρ. For a primal candidate X which we obtain from Weight Update, the oracle returns a solution y to the system

$$\begin{gathered} b \cdot y > R \\ \sum_{j=1}^{m}(A_j \bullet X)y_j \leq R \\ y \geq 0, \end{gathered} \tag{6}$$

or it returns INFEASIBLE if no solutions exist. The width ρ ensures the y returned by the oracle is relatively close to a solution to (5). The solution $y^{(t)}$ returned by the oracle satisfies

$$\| \sum_{j=1}^{m} y_j^{(t)} A_j - I \|_{op} \leq \rho.$$

3.2.2 Weight Update

We use the oracle output to observe the loss and update the weight. The loss matrix is defined by

$$M^{(t)} = -\frac{1}{\rho}\left(\sum_{j=1}^{m} y_j^{(t)} A_j - I\right).$$

Note that we indeed have $\|M^{(t)}\| \leq 1$. Then update the weight as described in Section 3.1, i.e.

$$W^{(t+1)} = \exp\left(-\eta \sum_{i=1}^{t} M^{(i)}\right), \quad P^{(t+1)} = \frac{W^{(t+1)}}{\mathrm{Tr}(W^{(t+1)})}.$$

The next candidate primal solution is given by $X^{(t+1)} = RP^{(t+1)}$ to satisfy the trace bound.

3.2.3 Runtime

Theorem 3.3. *Let $\eta = \frac{\varepsilon}{2\rho R}$ and $T = \lceil \frac{4\rho^2 R^2 \ln n}{\varepsilon^2} \rceil$. Suppose that the oracle never fails in any of the T iterations, which means the primal program is never satisfied. Then, the dual solution output $y = \frac{1}{T(1+\varepsilon/R)} \sum_{t=1}^{T} y^{(t)}$ satisfies the constraints with $b \cdot y \geq R - \varepsilon$.*

Proof. By Theorem 3.2 and the oracle constraints, we have

$$M^{(t)} \bullet P^{(t)} = -\frac{1}{\rho}\left(\sum_{j=1}^{m} y_j^{(t)} A_j - I\right) \bullet \frac{1}{R} X^{(t)} \geq 0.$$

Then, the inequality simplifies to

$$\lambda_{\min}\left(\sum_{t=1}^{T} M^{(t)}\right) + \eta T + \frac{\ln n}{\eta} \geq 0 \implies \lambda_{\min}\left(\frac{1}{T}\sum_{t=1}^{T} M^{(t)}\right) + \eta + \frac{\ln n}{\eta T} \geq 0.$$

Plugging in the values of η and T we have

$$\lambda_{\min}\left(\frac{1}{T}\sum_{t=1}^{T} M^{(t)}\right) = \lambda_{\min}\left(\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{m} -\frac{1}{\rho}(y_j^{(t)} A_j - I)\right) \geq -\frac{\varepsilon}{\rho R}.$$

Hence,

$$\lambda_{\max}\left(\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{m} y_j^{(t)} A_j - I\right) \leq \frac{\varepsilon}{R}.$$

Let $\bar{y} = \frac{1}{T}\sum_{t=1}^{T} y^{(t)}$. Then,

$$\lambda_{\max}\left(\sum_{j=1}^{m} \bar{y}_j A_j - I\right) \leq \frac{\varepsilon}{R}.$$

Therefore,

$$\sum_{j=1}^{m} \bar{y}_j A_j \preceq \left(1 + \frac{\varepsilon}{R}\right) I.$$

Letting $y = \frac{\bar{y}}{1+\varepsilon/R}$, we have $\sum_{j=1}^{m} y_j A_j \preceq I$ meaning that y satisfies the rest of the constraints of the Oracle LP. Moreover,

$$b \cdot y = b \cdot \left(\frac{1}{T(1+\varepsilon/R)}\sum_{t=1}^{T} y^{(t)}\right) > \frac{TR}{T(1+\varepsilon/R)} = \frac{R}{1 + \underbrace{\varepsilon/R}_{>0}} \geq R(1-\varepsilon/R) = R - \varepsilon,$$

as required. □

The error ε measures how close the oracle is to failing, which in turn represents how close the primal SDP is to being satisfied. Therefore, this is indeed a lower bound on how close we are to a feasible primal solution. And it is achieved with $O(\rho^2 R^2 \ln n/\varepsilon^2)$ oracle calls and each oracle call takes at most $O(mn^2)$ time.

References

[1] Sanjeev Arora, and Satyen Kale. 'A Combinatorial, Primal-Dual Approach to Semidefinite Programs'. Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing [San Diego California USA], 2007, pp. 227–36. DOI.org (Crossref), https://doi.org/10.1145/1250790.1250823.
[2] Sanjeev Arora, Elad Hazan, Satyen Kale: The Multiplicative Weights Update Method: a Meta-Algorithm and Applications. Theory Comput. 8(1): 121-164 (2012)
[3] Arora, S., et al. 'Fast Algorithms for Approximate Semidefinite Programming Using the Multiplicative Weights Update Method'. 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05) [Pittsburgh, PA], 2005, pp. 339–48. DOI.org (Crossref), https://doi.org/10.1109/SFCS.2005.35.