

# Directed Studies 2.

## Matrix Multiplicate Weights Algorithm and its connections to quantum computing

Attila Monos    Advisor: András Gilyén

Eötvös Loránd University

Friday 10<sup>th</sup> January, 2025

## ① Matrix Multiplicative Weights (MMW)

# Multiply Weights I.

- Multiplicative Weights algorithm:

- 1: Fix an  $\eta \leq \frac{1}{2}$ . With each decision  $i$  associate the weight  $w_i = 1$
- 2: **for**  $t = 1, 2, \dots, T$  **do**
- 3: Choose decision  $i$  with probability proportional to its weight  $w_i^{(t)}$ .
- 4: Observe the costs of decisions  $m^{(t)}$ .
- 5: Penalize the costly decisions: for every decision  $i$  set

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

## Theorem

Assume that all costs  $m_i^{(t)} \in [-1; 1]$  and  $\eta \leq \frac{1}{2}$ . Then the Multiply Weights algorithm guarantees that after  $T$  rounds, for any decision  $i$ , we have:

$$\sum_{t=1}^T m_i^{(t)} \cdot p_i^{(t)} \leq \sum_{t=1}^T m_i^{(t)} + \eta \sum_{t=1}^T |m_i^{(t)}| + \frac{\ln n}{\eta},$$

# Solving a LP with MW I.

- Matrix game

# Solving a LP with MW I.

- Matrix game
  - Moves:  $[n]$ ,  $[m]$

# Solving a LP with MW I.

- Matrix game
  - Moves:  $[n], [m]$
  - Payoff:  $A \in [-1; 1]^{n \times m}$

# Solving a LP with MW I.

- Matrix game
  - Moves:  $[n], [m]$
  - Payoff:  $A \in [-1; 1]^{n \times m}$
  - Probabilities:  $x \in \Delta^n, y \in \Delta^m$
  - Goal: optimize  $y$  for the best possible strategy  $x$

$$\min_{y \in \Delta^m} \max_{x \in \Delta^n} x^T A y = \min_{y \in \Delta^m} \max_{i \in [n]} e_i^T A y,$$

- Matrix game

- Moves:  $[n], [m]$
- Payoff:  $A \in [-1; 1]^{n \times m}$
- Probabilities:  $x \in \Delta^n, y \in \Delta^m$
- Goal: optimize  $y$  for the best possible strategy  $x$

$$\min_{y \in \Delta^m} \max_{x \in \Delta^n} x^T A y = \min_{y \in \Delta^m} \max_{i \in [n]} e_i^T A y,$$

Primal:

$$\begin{aligned} \min \quad & \lambda \\ \text{s.t.} \quad & A y \leq \lambda e \\ & y \in \Delta^m \\ & \lambda \in [-1; 1] \end{aligned}$$

Dual:

$$\begin{aligned} \max \quad & \lambda' \\ \text{s.t.} \quad & A^T x \geq \lambda' e \\ & x \in \Delta^n \\ & \lambda' \in [-1; 1] \end{aligned}$$

- Modified Grigoriados-Khachiyan (MGK) algorithm:

1:  $x^{(0)} \leftarrow 0 \in \mathbb{R}^n$  and  $y^{(0)} \leftarrow 0 \in \mathbb{R}^m$

2: **for**  $t = 1, 2, \dots$  **do**

3:  $\eta^{(t)} = \frac{1}{2\sqrt{t}}$

4:  $u^{(t)} \leftarrow -A^T x^{(t)}$  and  $v^{(t)} \leftarrow Ay^{(t)}$

5:  $P^{(t)} \leftarrow \exp(u^{(t)}) = \exp(-A^T x^{(t)})$  and  
 $Q^{(t)} \leftarrow \exp(v^{(t)}) = \exp(Ay^{(t)})$

6:  $p^{(t)} \leftarrow \frac{P^{(t)}}{\|P^{(t)}\|_1}$  and  $q^{(t)} \leftarrow \frac{Q^{(t)}}{\|Q^{(t)}\|_1}$

7: Sample  $a \sim p^{(t)}$  and  $b \sim q^{(t)}$

8:  $y^{(t+1)} = y^{(t)} + \eta^{(t)} e_a$  and  $x^{(t+1)} = x^{(t)} + \eta^{(t)} e_b$ .

# Solving a LP with MW III.

## Theorem

- Let  $\delta \in (0, \frac{1}{3})$ . With probability at least  $\delta$  the MGK algorithm produces a sequence of  $x^{(t)}$  and  $y^{(t)}$  such that for all  $t$  the intermediate solutions  $\frac{x^{(t)}}{\|x^{(t)}\|_1}$  and  $\frac{y^{(t)}}{\|y^{(t)}\|_1}$  are  $\varepsilon$ -optimal solutions with

$$\varepsilon = \frac{2}{\sqrt{t}} \cdot \left( 3 \ln t + \ln(nm) + \ln\left(\frac{1}{\delta}\right) + 2 \right)$$

- Given a pair of strategies  $x$  and  $y$  and  $k = \mathcal{O}(\frac{1}{\varepsilon^2})$  independent samples  $i_1, \dots, i_k$  from  $x$  and  $j_1, \dots, j_k$  from  $y$ ,  $\frac{1}{k} \sum_{l=1}^k A_{i_l j_l}$  is an  $\varepsilon$ -approximate estimate of the value of the game corresponding to strategies  $x, y$ .
- The MGK algorithm can be implemented to find an  $\varepsilon$ -optimal pair of solutions with probability at least  $1 - \delta$  in  $\frac{16}{\varepsilon^2} \ln(\frac{nm}{\delta})$  iterations, using  $\mathcal{O}(n + m)$  time per iteration and  $n + m$  queries to the entries of  $A$ .

- Generalize the matrix game:

- Generalize the matrix game:
  - Strategy: unit vector  $v$  from a distribution  $\mathcal{P}$

- Generalize the matrix game:
  - Strategy: unit vector  $v$  from a distribution  $\mathcal{P}$
  - Loss matrix  $M$  with eigenvalues from  $[-1; 1]$

- Generalize the matrix game:
  - Strategy: unit vector  $v$  from a distribution  $\mathcal{P}$
  - Loss matrix  $M$  with eigenvalues from  $[-1; 1]$
  - Largest eigenvalue:  $\lambda_n(M)$
  - Payout:  $v^T M v = M \bullet v v^T$   
 $A \bullet B = \text{Tr}(A^T B) = \sum_{ij} A_{ij} B_{ij}$

- Generalize the matrix game:
  - Strategy: unit vector  $v$  from a distribution  $\mathcal{P}$
  - Loss matrix  $M$  with eigenvalues from  $[-1; 1]$
  - Largest eigenvalue:  $\lambda_n(M)$
  - Payout:  $v^T M v = M \bullet v v^T$   
 $A \bullet B = \text{Tr}(A^T B) = \sum_{ij} A_{ij} B_{ij}$
  - Expected loss:  $M \bullet \mathbb{E}_{\mathcal{P}}(v v^T)$

- Generalize the matrix game:
  - Strategy: unit vector  $v$  from a distribution  $\mathcal{P}$
  - Loss matrix  $M$  with eigenvalues from  $[-1; 1]$
  - Largest eigenvalue:  $\lambda_n(M)$
  - Payout:  $v^T M v = M \bullet v v^T$   
 $A \bullet B = \text{Tr}(A^T B) = \sum_{ij} A_{ij} B_{ij}$
  - Expected loss:  $M \bullet \mathbb{E}_{\mathcal{P}}(v v^T)$
  - Goal: minimize  $\sum_{t=1}^T v^T M^{(t)} v$

- Matrix Multiply Weights algorithm:
  - 1: Fix  $\eta \leq 1$ , number of rounds  $T$ .
  - 2: Set  $W^{(1)} = I$ .
  - 3: **for**  $t = 1, 2, \dots, T$  **do**
  - 4: Use the density matrix  $P^{(t)} = \frac{W^{(t)}}{\text{Tr}(W^{(t)})}$
  - 5: Obtain loss matrix  $M^{(t)}$
  - 6: Update the weight matrix as follows:

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

## Theorem

*For any sequence of loss matrices  $M^{(1)}, M^{(2)}, \dots, M^{(T)}$ , the MMW algorithm generates density matrices  $P^{(1)}, P^{(2)}, \dots, P^{(t)}$  such that*

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

Primal:

$$\max C \bullet X$$

$$\forall j \in [m]: A_j \bullet X \preceq b_j$$

$$X \succeq 0$$

Dual:

$$\min b \cdot y$$

$$\sum_{j=1}^m A_j y_j \succeq C$$

$$y \geq 0$$

- Oracle is an algorithm that, given a primal candidate solution  $X \succeq 0$  either outputs a vector  $y$  (if there is such a vector) such that

$$\begin{aligned}b \cdot y &\leq \alpha \\ \sum_{j=1}^m (A_j \bullet X) y_j &\geq C \bullet X \\ y &\geq 0\end{aligned}$$

our outputs "fail". In the latter case, we'll say that the (Oracle) failed.

- Oracle is an algorithm that, given a primal candidate solution  $X \succeq 0$  either outputs a vector  $y$  (if there is such a vector) such that

$$\begin{aligned} b \cdot y &\leq \alpha \\ \sum_{j=1}^m (A_j \bullet X) y_j &\geq C \bullet X \\ y &\geq 0 \end{aligned}$$

our outputs "fail". In the latter case, we'll say that the (Oracle) failed.

- Oracle fails:  $X$  can be scaled to become a primal feasible solution

## A Primal-Dual solver with MMW III.

- Width of Oracle: smallest  $\rho \geq 0$ : for primal candidate  $X, y$  satisfies  $\|A_j y_j - C\| \leq \rho$

## A Primal-Dual solver with MMW III.

- Width of Oracle: smallest  $\rho \geq 0$ : for primal candidate  $X$ ,  $y$  satisfies  $\|A_j y_j - C\| \leq \rho$

Algorithm:

- 1: Fix trace bound  $R$ , width bound  $\rho$ , parameters  $\eta \leq 1$  and  $T$  for the MMW algorithm
- 2: **for**  $t = 1, 2, \dots, T$  **do**
- 3: Obtain the density matrix  $P^{(t)}$  from step  $t$  of the MMW algorithm
- 4:  $X^{(t)} = RP^{(t)}$
- 5: Run Oracle with primal candidate  $X^{(t)}$
- 6:     **if** Oracle fails **then**
- 7:         Abort.
- 8: Let  $y^{(t)}$  be the vector generated by Oracle
- 9: Use  $M^{(t)} = \frac{1}{\rho} (\sum_{j=1}^m A_j y_j^{(t)} - C)$  as the loss matrix in the next step of the MMW algorithm
- 10: If Oracle never fails in all  $T$  rounds, then output the dual solution  $\bar{y} = \frac{1}{T} \sum_{t=1}^T y^{(t)} + \frac{\epsilon}{T} e_1$ .

## Theorem

*Run the Primal-Dual SDP solver with MMW with parameter setting  $\eta = \frac{\varepsilon}{2\rho R}$  and  $T = \lceil \frac{4\rho^2 R^2 \ln n}{\varepsilon^2} \rceil$  and assume that Oracle never fails in any of the  $T$  iterations. Then the dual solution output,  $\bar{y}$  is feasible with objective value at most  $\alpha + \varepsilon$ .*

## ① Matrix Multiplicative Weights (MMW)

- 1 Matrix Multiplicative Weights (MMW)
- 2 Quantum computing, some optimization problems

- 1 Matrix Multiplicative Weights (MMW)
- 2 Quantum computing, some optimization problems
- 3 Generalized Markov Chains in quantum computing and MMW

- S. Arora - E. Hazan - S. Kale *The Multiplicative Weights Update Method: A Meta-Algorithm and Applications*  
<https://theoryofcomputing.org/articles/v008a006/>
- J. van Apeldoorn - A. Gilyén *Quantum algorithms for zero-sum games, Main Algorithm, p. 3*  
<https://arxiv.org/pdf/1904.03180>
- S. Arora - S. Kale *A Combinatorial, Primal-Dual Approach to Semidefinite Programs*  
<https://dl.acm.org/doi/pdf/10.1145/2837020>
- J. van Apeldoorn - A. Gilyén *Improvements in Quantum SDP-Solving with Applications, section 2.2*  
<https://arxiv.org/pdf/1804.05058>