A Quadratic Speedup in Finding Nash Equilibria of Quantum Zero-Sum Games

[arXiv:2311.10859]

Francisca Vasconcelos*
UC Berkeley

Quantum Techniques in Machine Learning Conference 2023

November 21st, 2023

Joint work with:

Emmanouil Vlatakis-Gkaragkounis*
UC Berkeley

Panayotis Mertikopoulos
CNRS Grenoble

Georgios Piliouras
SUTD

Michael I. Jordan
UC Berkeley

Overview of Results

Overview of Results

- We propose a new algorithm (**Optimistic Matrix Multiplicative Weight Updates**) for finding approximate Nash equilibria of quantum zero-sum games.

Overview of Results

- We propose a new algorithm (**Optimistic Matrix Multiplicative Weight Updates**) for finding approximate Nash equilibria of quantum zero-sum games.
  - To do so we leverage a **gradient-based** view of QZSG.

Overview of Results

- We propose a new algorithm (**Optimistic Matrix Multiplicative Weight Updates**) for finding approximate Nash equilibria of quantum zero-sum games.
  - To do so we leverage a **gradient-based** view of QZSG.
  - This allows us to easily leverage optimization techniques from the **classical** games literature.

Overview of Results

- We propose a new algorithm (**Optimistic Matrix Multiplicative Weight Updates**) for finding approximate Nash equilibria of quantum zero-sum games.
  - To do so we leverage a **gradient-based** view of QZSG.
  - This allows us to easily leverage optimization techniques from the **classical** games literature.

- We prove that OMMWU achieves an $\mathcal{O}(1/\epsilon)$ iteration complexity.

Overview of Results

- We propose a new algorithm (**Optimistic Matrix Multiplicative Weight Updates**) for finding approximate Nash equilibria of quantum zero-sum games.
  - To do so we leverage a **gradient-based** view of QZSG.
  - This allows us to easily leverage optimization techniques from the **classical** games literature.
- We prove that OMMWU achieves an $\mathcal{O}(1/\epsilon)$ iteration complexity.
  - This is a **quadratic speedup** relative to the best prior algorithm [JW09].

Overview of Results

- We propose a new algorithm (**Optimistic Matrix Multiplicative Weight Updates**) for finding approximate Nash equilibria of quantum zero-sum games.
  - To do so we leverage a **gradient-based** view of QZSG.
  - This allows us to easily leverage optimization techniques from the **classical** games literature.
- We prove that OMMWU achieves an $\mathcal{O}(1/\epsilon)$ iteration complexity.
  - This is a **quadratic speedup** relative to the best prior algorithm [JW09].
  - We leverage the proof technique of [EN20] for **monotone variational inequalities**.

Overview of Results

- We propose a new algorithm (**Optimistic Matrix Multiplicative Weight Updates**) for finding approximate Nash equilibria of quantum zero-sum games.
  - To do so we leverage a **gradient-based** view of QZSG.
  - This allows us to easily leverage optimization techniques from the **classical** games literature.
- We prove that OMMWU achieves an $\mathcal{O}(1/\epsilon)$ iteration complexity.
  - This is a **quadratic speedup** relative to the best prior algorithm [JW09].
  - We leverage the proof technique of [EN20] for **monotone variational inequalities**.
- We further introduce a design framework for QZSG algorithms.

Overview of Results

- We propose a new algorithm (**Optimistic Matrix Multiplicative Weight Updates**) for finding approximate Nash equilibria of quantum zero-sum games.
  - To do so we leverage a **gradient-based** view of QZSG.
  - This allows us to easily leverage optimization techniques from the **classical** games literature.
- We prove that OMMWU achieves an $\mathcal{O}(1/\epsilon)$ iteration complexity.
  - This is a **quadratic speedup** relative to the best prior algorithm [JW09].
  - We leverage the proof technique of [EN20] for **monotone variational inequalities**.
- We further introduce a design framework for QZSG algorithms.
  - We use this to **unify** the QZSG algorithms landscape and **motivate** OMMWU.

A Quadratic Speedup in Finding Nash Equilibria of Quantum Zero-Sum Games

What is a Quantum Zero-Sum Game (QZSG)?

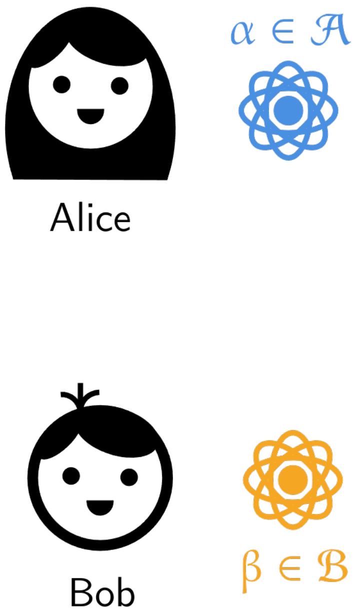
What is a Quantum Zero-Sum Game (QZSG)?

- A **two-player** game

Alice

Bob

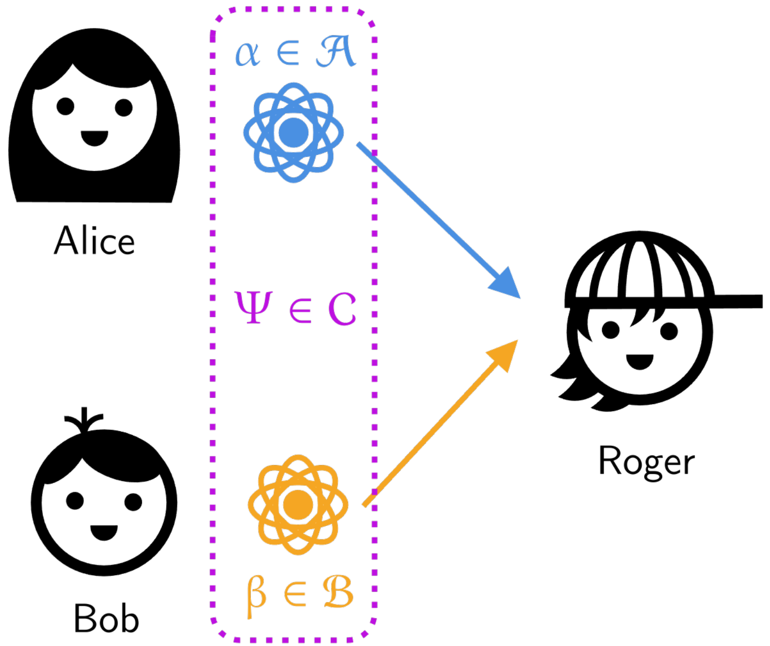
What is a Quantum Zero-Sum Game (QZSG)?



- A **two-player** game
- In each round, players play **unentangled** mixed states (**spectraplex**):

$$\alpha \in \mathcal{A} = \mathbb{C}^{2^n x 2^n}, \qquad \beta \in \mathcal{B} = \mathbb{C}^{2^m x 2^m}$$

What is a Quantum Zero-Sum Game (QZSG)?

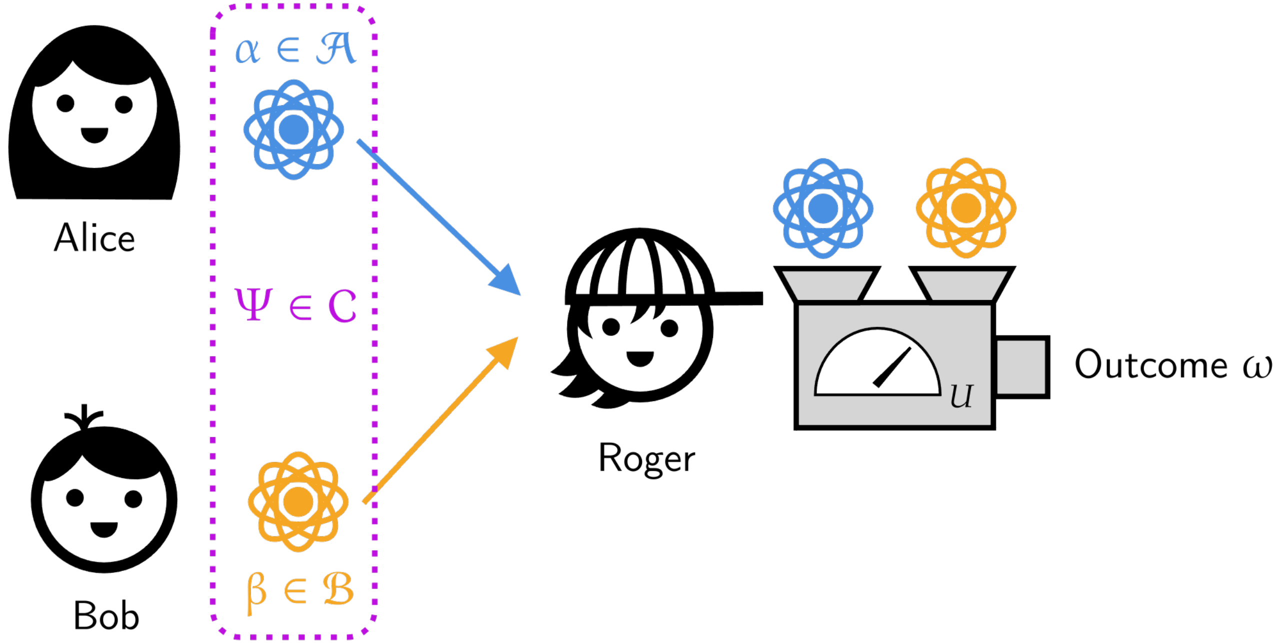


- A **two-player** game
- In each round, players play **unentangled** mixed states (**spectraplex**):

$$\alpha \in \mathcal{A} = \mathbb{C}^{2^n x 2^n}, \qquad \beta \in \mathcal{B} = \mathbb{C}^{2^m x 2^m}$$

- They send **joint state** $\Psi = (\alpha, \beta)$ to a referee

What is a Quantum Zero-Sum Game (QZSG)?



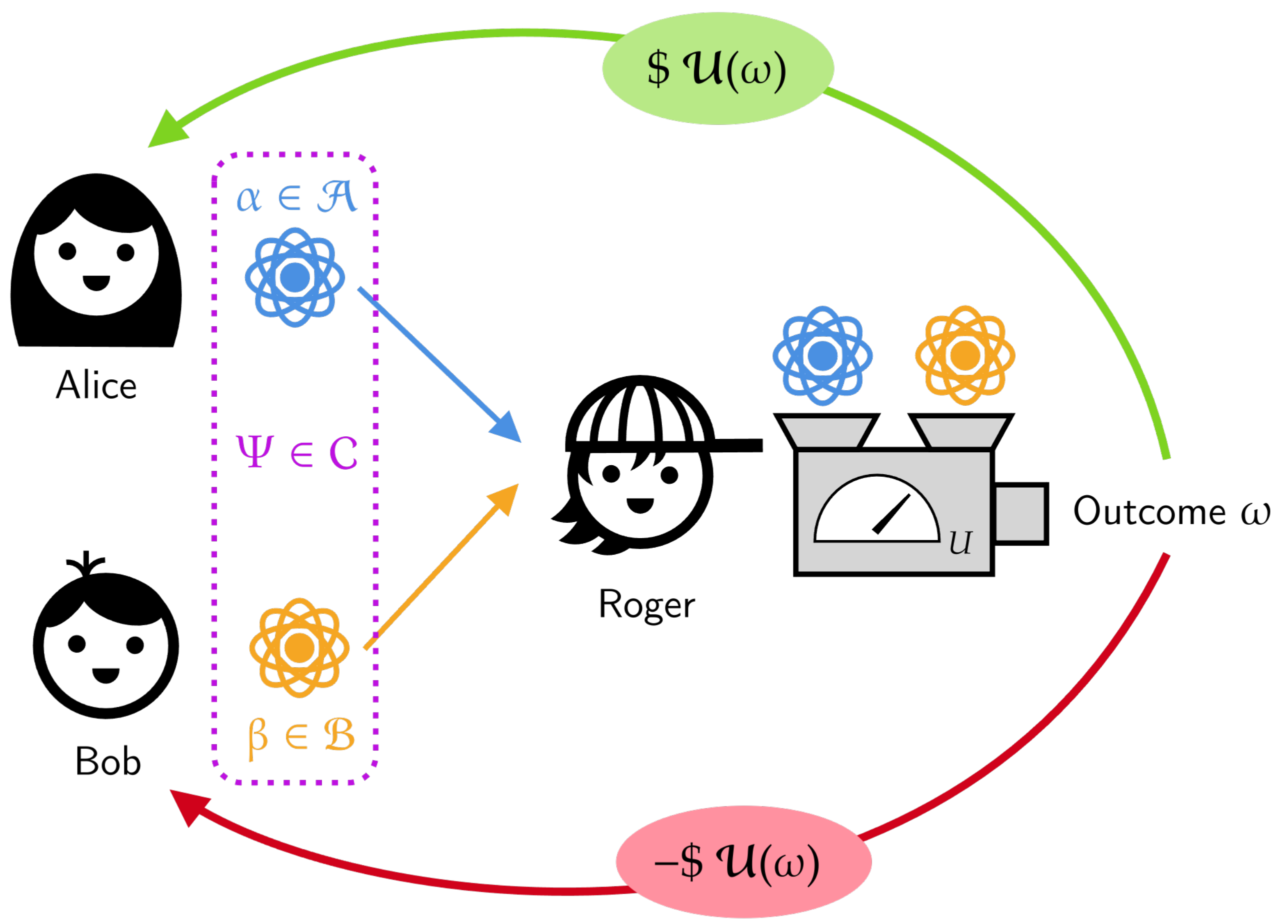
- A **two-player** game
- In each round, players play **unentangled** mixed states (**spectraplex**):

$$\alpha \in \mathcal{A} = \mathbb{C}^{2^n x 2^n}, \qquad \beta \in \mathcal{B} = \mathbb{C}^{2^m x 2^m}$$

- They send **joint state** $\Psi = (\alpha, \beta)$ to a referee
- The referee makes a **joint measurement** w.r.t. payoff observable:

$$U = \sum_{\omega \in \Omega} \mathcal{U}(\omega)\, P_\omega$$

What is a Quantum Zero-Sum Game (QZSG)?



- A **two-player** game
- In each round, players play **unentangled** mixed states (**spectraplex**):

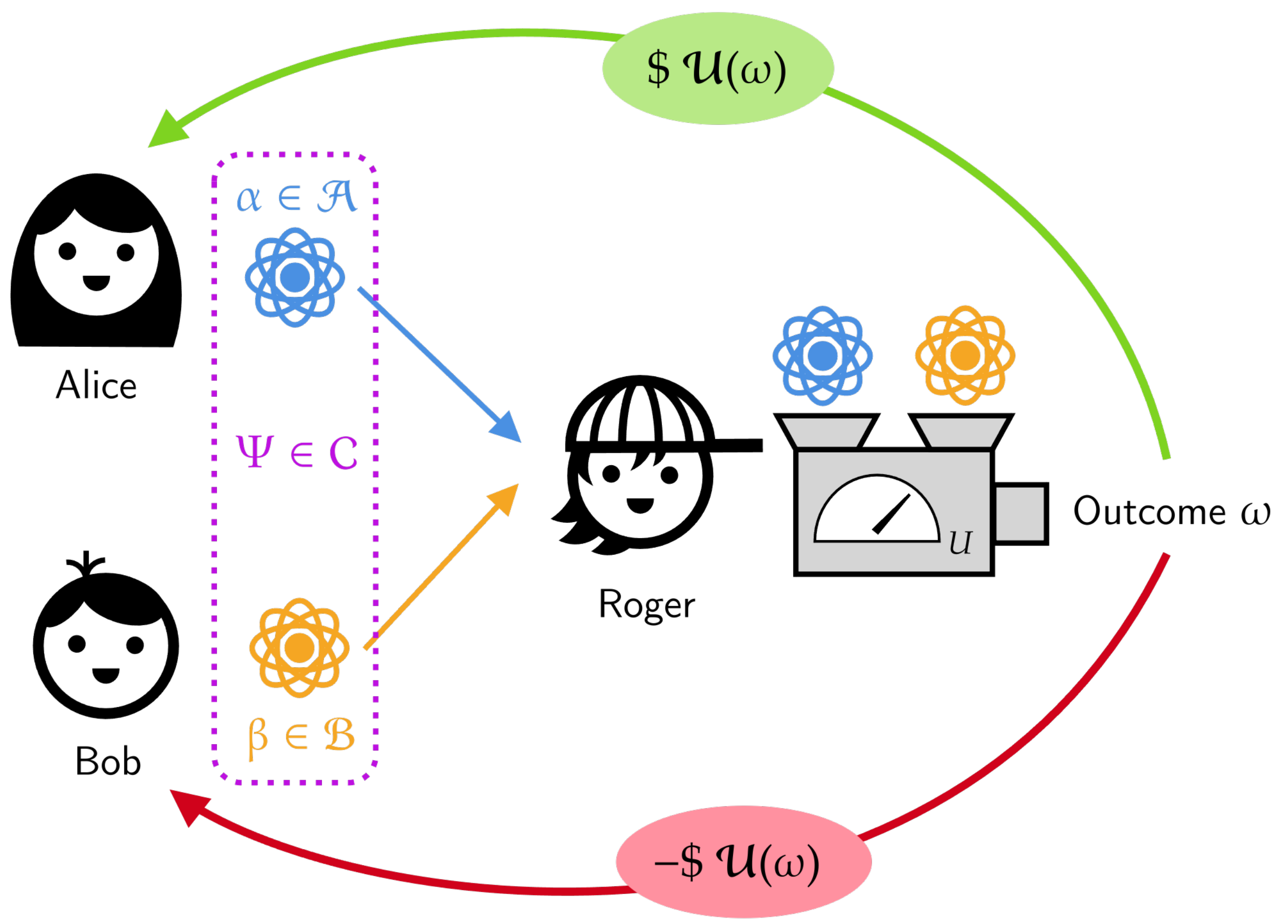
$$\alpha \in \mathcal{A} = \mathbb{C}^{2^n x 2^n}, \qquad \beta \in \mathcal{B} = \mathbb{C}^{2^m x 2^m}$$

- They send **joint state** $\Psi = (\alpha, \beta)$ to a referee
- The referee makes a **joint measurement** w.r.t. payoff observable:

$$U = \sum_{\omega \in \Omega} \mathcal{U}(\omega)\, P_\omega$$

- Based on the measurement outcome, the referee **rewards** the players

What is a Quantum Zero-Sum Game (QZSG)?



- A **two-player** game
- In each round, players play **unentangled** mixed states (**spectraplex**):

$$\alpha \in \mathcal{A} = \mathbb{C}^{2^n x 2^n}, \qquad \beta \in \mathcal{B} = \mathbb{C}^{2^m x 2^m}$$

- They send **joint state** $\Psi = (\alpha, \beta)$ to a referee
- The referee makes a **joint measurement** w.r.t. payoff observable:

$$U = \sum_{\omega \in \Omega} \mathcal{U}(\omega)\, P_\omega$$

- Based on the measurement outcome, the referee **rewards** the players
  - **zero-sum** ⇒ one player's **win** is the other's **loss**

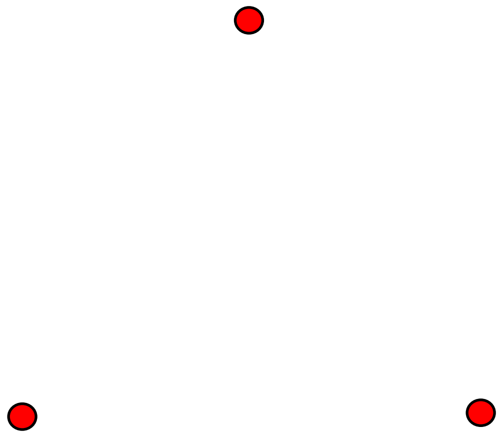
Game Strategies & Expected Payoff

Game Strategies & Expected Payoff

- CZSG are played in the **simplex**

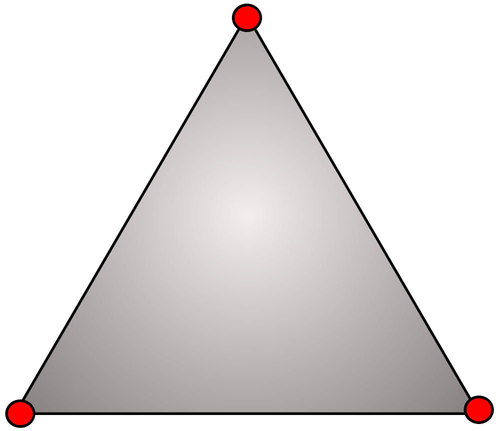
Game Strategies & Expected Payoff

- CZSG are played in the **simplex**:



Game Strategies & Expected Payoff

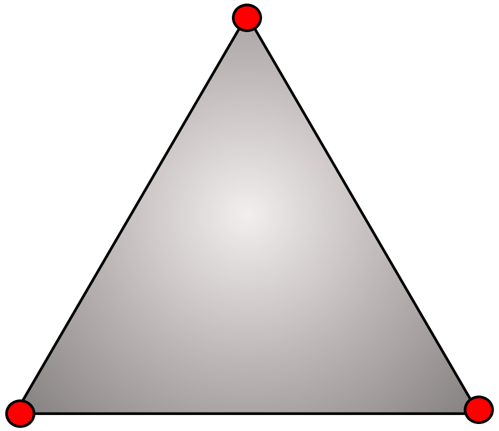
- CZSG are played in the **simplex**:



- Players play **indeterministic strategies**:
  - Probability vectors: $|\alpha\rangle, |\beta\rangle \in [0,1]^n$

Game Strategies & Expected Payoff

- CZSG are played in the **simplex**:

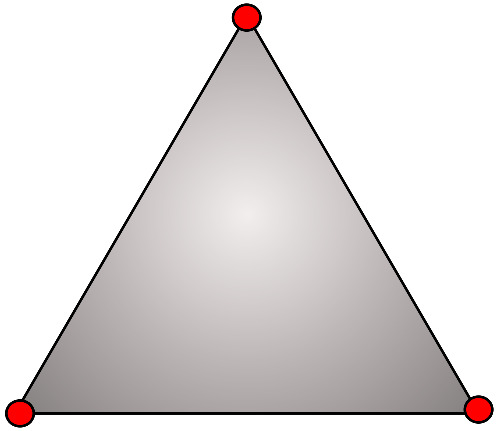


- Players play **indeterministic strategies**:
  - Probability vectors: $|\alpha\rangle, |\beta\rangle \in [0,1]^n$
- **Expected utility** for specific strategies:

$$u(\alpha, \beta) = \langle \beta | U | \alpha \rangle$$

Game Strategies & Expected Payoff

- CZSG are played in the **simplex**:



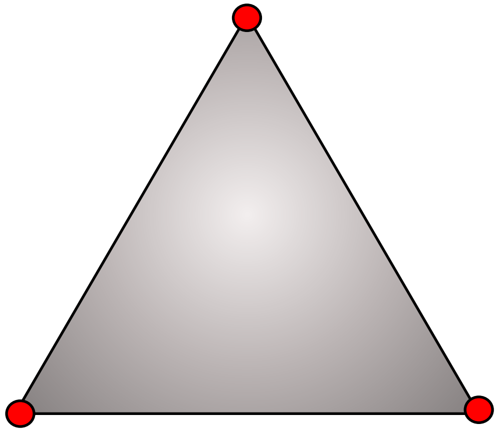
- Players play **indeterministic strategies**:
  - Probability vectors: $|\alpha\rangle, |\beta\rangle \in [0,1]^n$
- **Expected utility** for specific strategies:

$$u(\alpha, \beta) = \langle\beta|U|\alpha\rangle$$

- QZSG are played in the **spectraplex**

Game Strategies & Expected Payoff

- CZSG are played in the **simplex**:



- QZSG are played in the **spectraplex**:

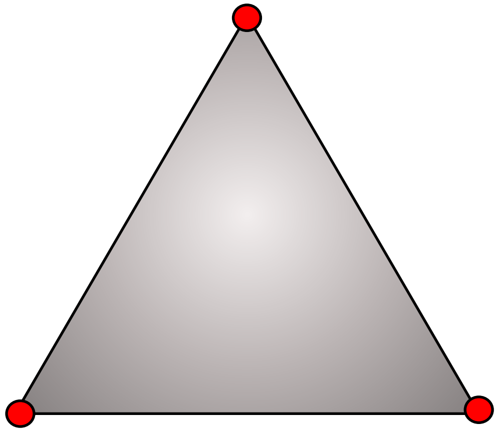
- Players play **indeterministic strategies**:
  - Probability vectors: $|\alpha\rangle, |\beta\rangle \in [0,1]^n$

- **Expected utility** for specific strategies:

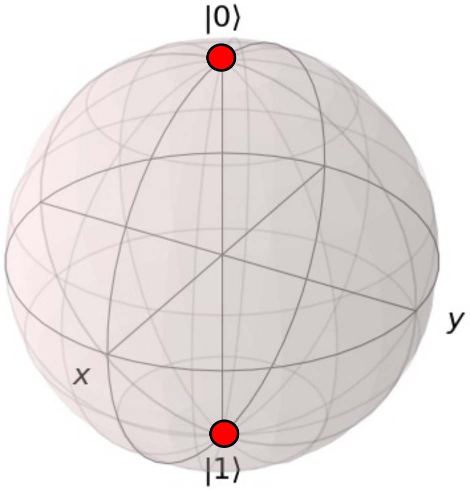
$$u(\alpha, \beta) = \langle\beta|U|\alpha\rangle$$

Game Strategies & Expected Payoff

- CZSG are played in the **simplex**:



- QZSG are played in the **spectraplex**:



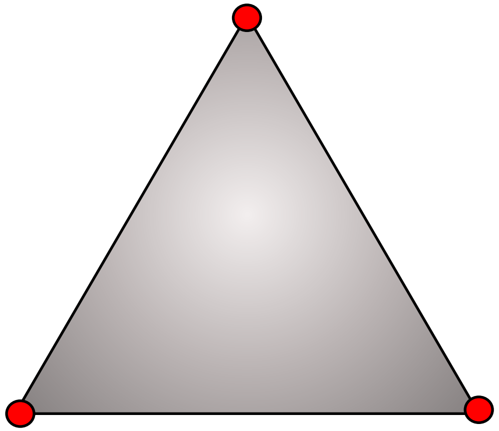
- Players play **indeterministic strategies**:
  - Probability vectors: $|\alpha\rangle, |\beta\rangle \in [0,1]^n$

- **Expected utility** for specific strategies:

$$u(\alpha, \beta) = \langle\beta|U|\alpha\rangle$$

Game Strategies & Expected Payoff

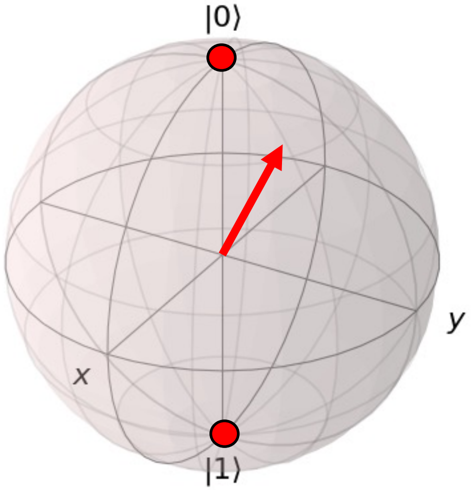
- CZSG are played in the **simplex**:



- Players play **indeterministic strategies**:
  - Probability vectors: $|\alpha\rangle, |\beta\rangle \in [0,1]^n$
- **Expected utility** for specific strategies:

$$u(\alpha, \beta) = \langle\beta|U|\alpha\rangle$$

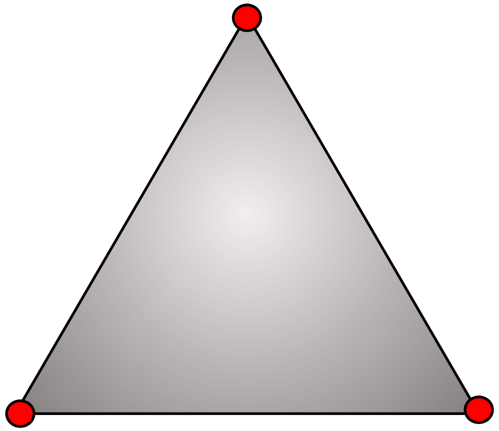
- QZSG are played in the **spectraplex**:



- Players play **mixtures** of indeterministic strategies (**meta-strategies**):
  - Density matrices: $\alpha \in \mathbb{C}^{2^n x 2^n}$, $\beta \in \mathbb{C}^{2^m x 2^m}$

Game Strategies & Expected Payoff

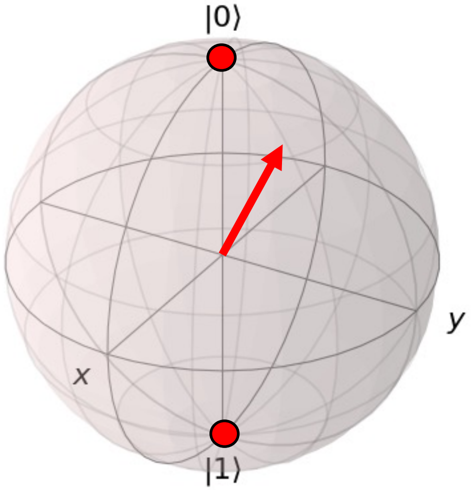
- CZSG are played in the **simplex**:



- Players play **indeterministic strategies**:
  - Probability vectors: $|\alpha\rangle, |\beta\rangle \in [0,1]^n$
- **Expected utility** for specific strategies:

$$u(\alpha, \beta) = \langle \beta | U | \alpha \rangle$$

- QZSG are played in the **spectraplex**:



- Players play **mixtures** of indeterministic strategies (**meta-strategies**):
  - Density matrices: $\alpha \in \mathbb{C}^{2^n x 2^n}$, $\beta \in \mathbb{C}^{2^m x 2^m}$
- **Expected utility** for specific strategies:

$$u(\alpha, \beta) = \mathrm{Tr}[U(\alpha \otimes \beta)]$$

The QZSG Objective

The QZSG Objective

- If Alice's expected payoff is $u(\alpha, \beta)$,

  Bob's expected payoff is $-u(\alpha, \beta)$

The QZSG Objective

- If Alice's expected payoff is $u(\alpha, \beta)$,

  Bob's expected payoff is $-u(\alpha, \beta)$

- In playing the game, each player wants to **maximize** their expected payoff:

The QZSG Objective

- If Alice's expected payoff is $u(\alpha, \beta)$,

  Bob's expected payoff is $-u(\alpha, \beta)$

- In playing the game, each player wants to **maximize** their expected payoff:

  - Alice wants: $\max_{\alpha} u(\alpha, \beta)$

The QZSG Objective

- If Alice's expected payoff is $u(\alpha, \beta)$,

  Bob's expected payoff is $-u(\alpha, \beta)$

- In playing the game, each player wants to **maximize** their expected payoff:

  – Alice wants: $\max_{\alpha} u(\alpha, \beta)$

  – Bob wants: $\max_{\beta} - u(\alpha, \beta) = \min_{\beta} u(\alpha, \beta)$

The QZSG Objective

- If Alice's expected payoff is $u(\alpha, \beta)$,

  Bob's expected payoff is $-u(\alpha, \beta)$

- In playing the game, each player wants to **maximize** their expected payoff:

  – Alice wants: $\max_{\alpha} u(\alpha, \beta)$

  – Bob wants: $\max_{\beta} -u(\alpha, \beta) = \min_{\beta} u(\alpha, \beta)$

- These are **competing** interests, defining a **minimax** optimization problem:

$$\min_{\beta \in \mathcal{B}} \max_{\alpha \in \mathcal{A}} u(\alpha, \beta)$$

The QZSG Objective

- If Alice's expected payoff is $u(\alpha, \beta)$,

  Bob's expected payoff is $-u(\alpha, \beta)$

- In playing the game, each player wants to **maximize** their expected payoff:
  - Alice wants: $\max_{\alpha} u(\alpha, \beta)$
  - Bob wants: $\max_{\beta} -u(\alpha, \beta) = \min_{\beta} u(\alpha, \beta)$

- These are **competing** interests, defining a **minimax** optimization problem:

$$\min_{\beta \in \mathcal{B}} \max_{\alpha \in \mathcal{A}} u(\alpha, \beta)$$

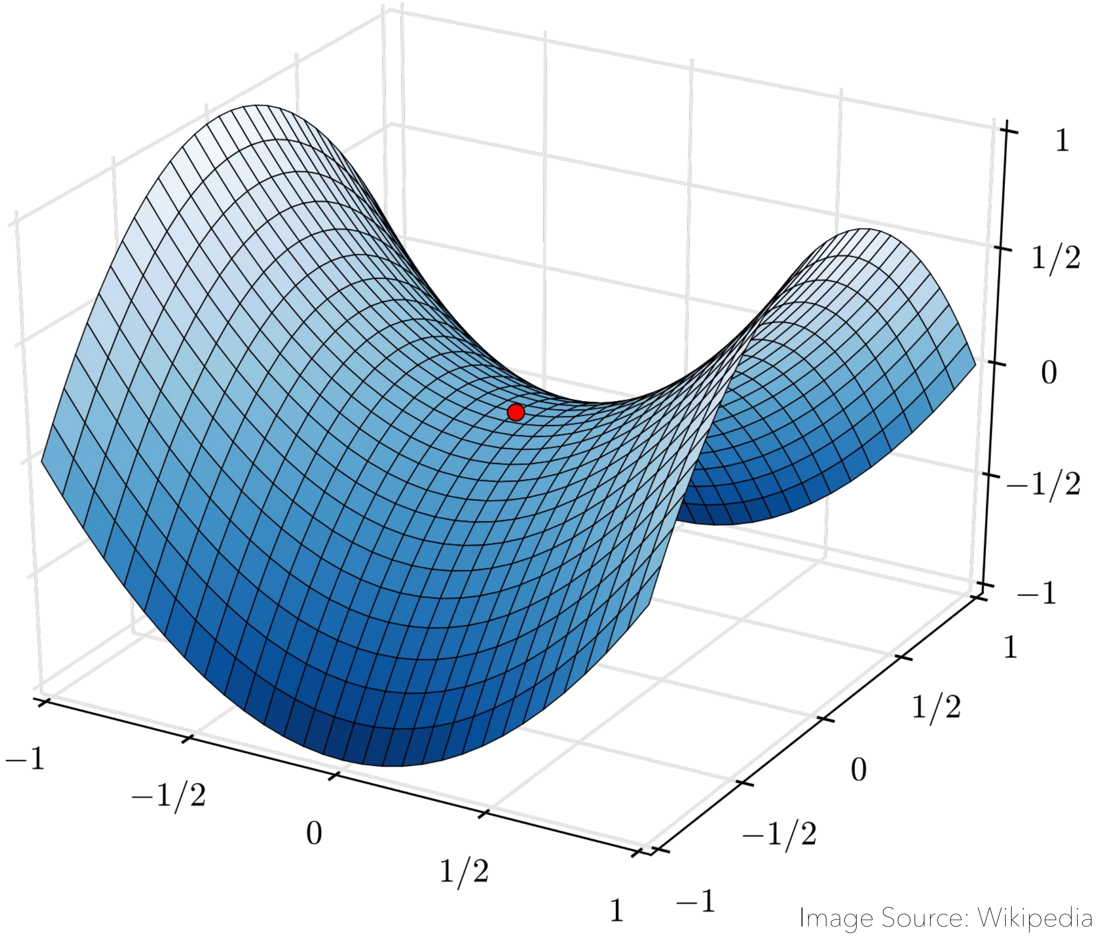


Image Source: Wikipedia

Why Study Quantum Games?

Why Study Quantum Games?

- General quantum games have emerged in many areas of quantum information:

Why Study Quantum Games?

- General quantum games have emerged in many areas of quantum information:
  - Non-local games (Bell, CSHS, …, MIP*=RE)

Why Study Quantum Games?

- General quantum games have emerged in many areas of quantum information:
  - Non-local games (Bell, CSHS, …, MIP*=RE)
  - Quantum interactive proofs (competitive refereed games)

Why Study Quantum Games?

- General quantum games have emerged in many areas of quantum information:
  - Non-local games (Bell, CSHS, …, MIP*=RE)
  - Quantum interactive proofs (competitive refereed games)
  - Multi-prover quantum interactive proofs (cooperative games)

Why Study Quantum Games?

- General quantum games have emerged in many areas of quantum information:
  - Non-local games (Bell, CSHS, …, MIP*=RE)
  - Quantum interactive proofs (competitive refereed games)
  - Multi-prover quantum interactive proofs (cooperative games)
  - Quantum coin-flipping (two player game)

Why Study Quantum Games?

- General quantum games have emerged in many areas of quantum information:
  - Non-local games (Bell, CSHS, …, MIP*=RE)
  - Quantum interactive proofs (competitive refereed games)
  - Multi-prover quantum interactive proofs (cooperative games)
  - Quantum coin-flipping (two player game)

- However, optimization of general quantum games is **PPAD-complete** [BW22]

Why Study QZSG, Specifically?

Why Study QZSG, Specifically?

- Meanwhile, as classically, QZSG optimization is **computationally tractable**

Why Study QZSG, Specifically?

- Meanwhile, as classically, QZSG optimization is **computationally tractable**
  - [JW09] proposed an explicit QZSG algo that converges to an ϵ-approx soln in $\mathcal{O}(1/\epsilon^2)$ iterations

Why Study QZSG, Specifically?

- Meanwhile, as classically, QZSG optimization is **computationally tractable**
  - [JW09] proposed an explicit QZSG algo that converges to an ϵ-approx soln in $\mathcal{O}(1/\epsilon^2)$ iterations
- Uses of QZSG:

Why Study QZSG, Specifically?

- Meanwhile, as classically, QZSG optimization is **computationally tractable**
  - [JW09] proposed an explicit QZSG algo that converges to an ϵ-approx soln in $\mathcal{O}(1/\epsilon^2)$ iterations
- Uses of QZSG:
  - Game theory: proof that quantum strategies ≥ classical strategies [M99]

Why Study QZSG, Specifically?

- Meanwhile, as classically, QZSG optimization is **computationally tractable**
  - [JW09] proposed an explicit QZSG algo that converges to an ϵ-approx soln in $\mathcal{O}(1/\epsilon^2)$ iterations
- Uses of QZSG:
  - Game theory: proof that quantum strategies $\geq$ classical strategies [M99]
  - Complexity theory: proof that QRG(1) $\subseteq$ PSPACE [JW09]

Why Study QZSG, Specifically?

- Meanwhile, as classically, QZSG optimization is **computationally tractable**
  - [JW09] proposed an explicit QZSG algo that converges to an ϵ-approx soln in $\mathcal{O}(1/\epsilon^2)$ iterations
- Uses of QZSG:
  - Game theory: proof that quantum strategies ≥ classical strategies [M99]
  - Complexity theory: proof that QRG(1) ⊆ PSPACE [JW09]
  - Machine learning: **Quantum Generative Adversarial Networks** [DK18]

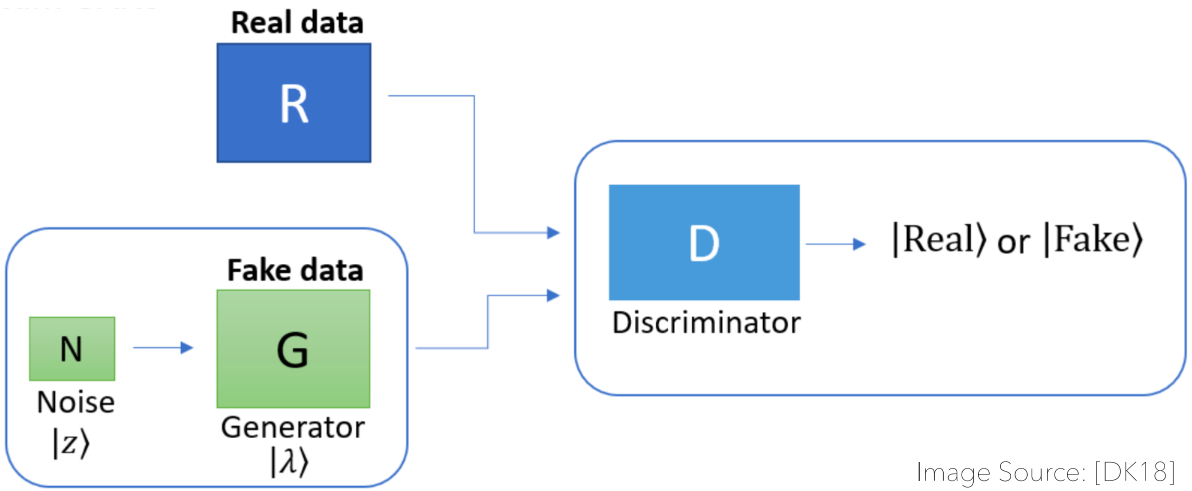


Image Source: [DK18]

A Quadratic Speedup in Finding Nash Equilibria of Quantum Zero-Sum Games

Nash Equilibria of QZSG

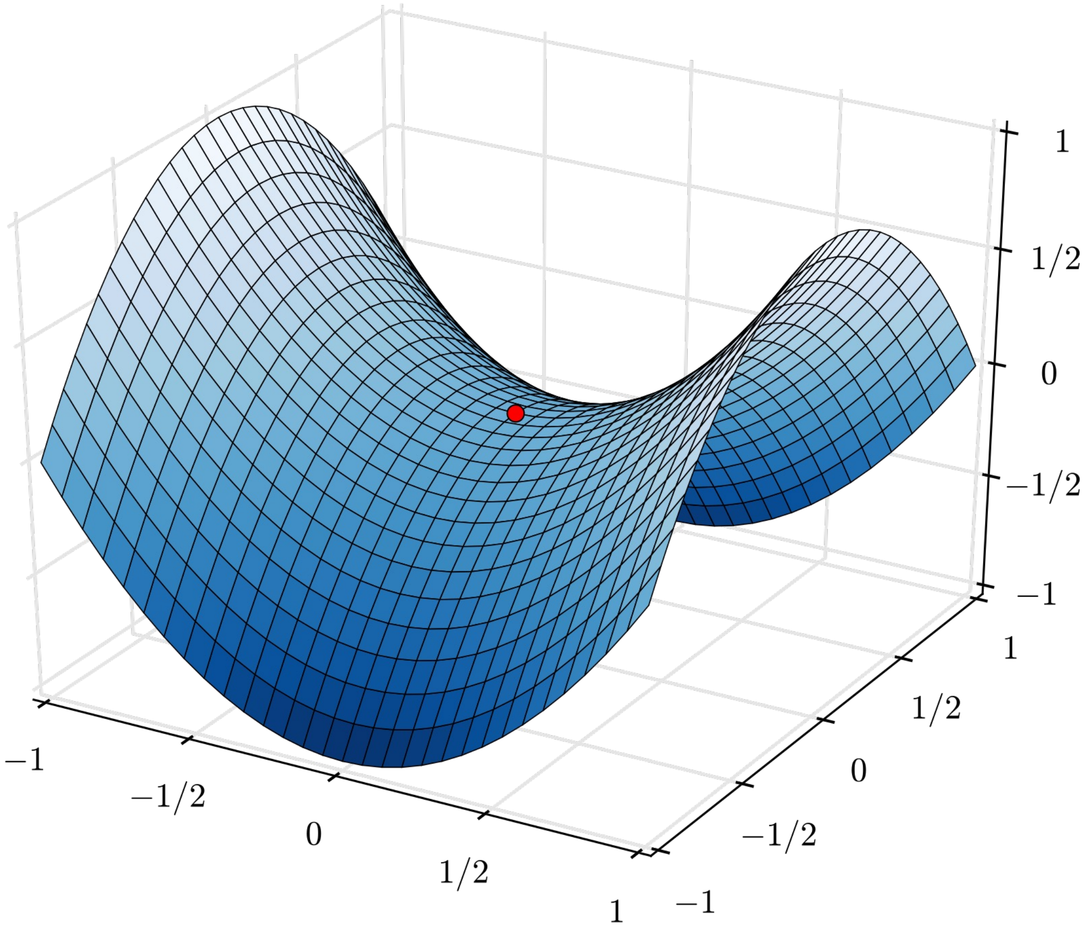


Image Source: Wikipedia

Nash Equilibria of QZSG

- The solutions (**fixed points**) of this minimax define the game's value:

$$u(\alpha^*, \beta^*) = \min_{\beta} \max_{\alpha} u(\alpha, \beta) = \max_{\alpha} \min_{\beta} u(\alpha, \beta)$$

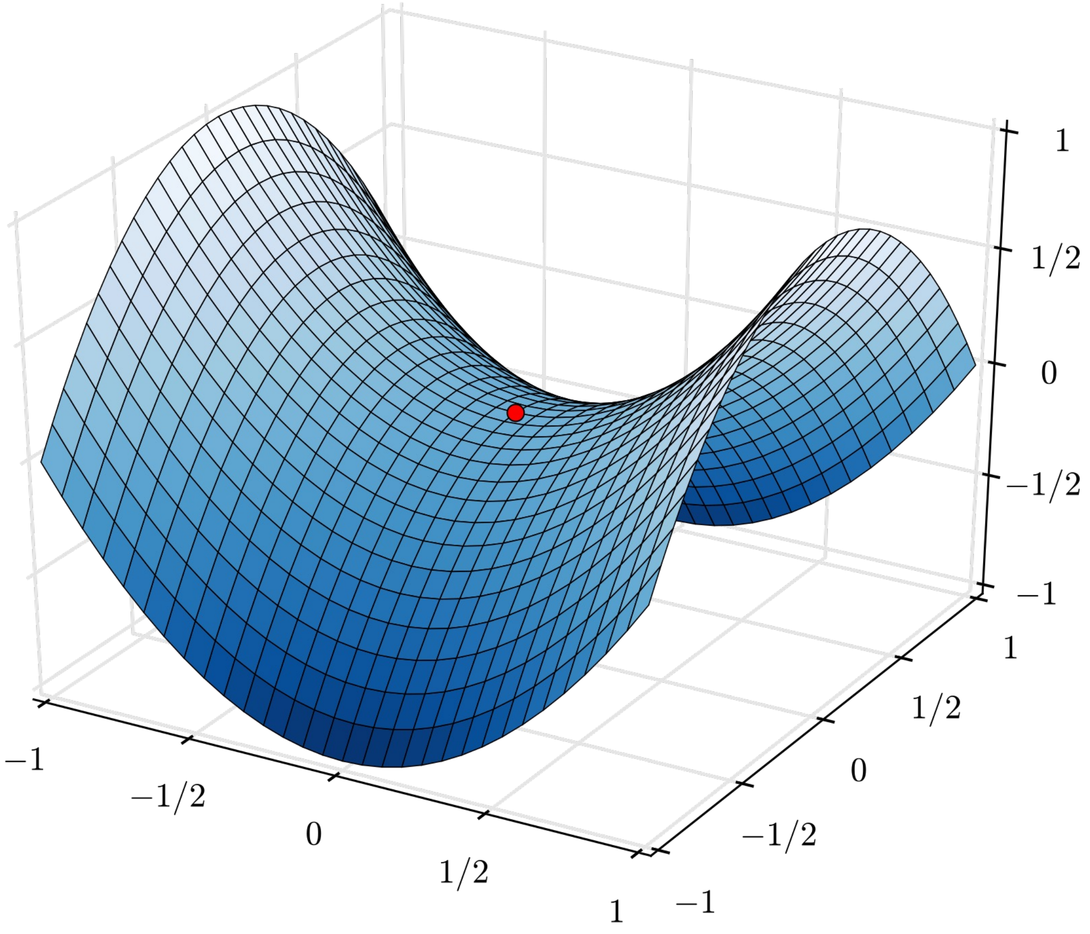


Image Source: Wikipedia

Nash Equilibria of QZSG

- The solutions (**fixed points**) of this minimax define the game's value:

$$u(\alpha^*, \beta^*) = \min_{\beta} \max_{\alpha} u(\alpha, \beta) = \max_{\alpha} \min_{\beta} u(\alpha, \beta)$$

von Neumann's Minimax Thm

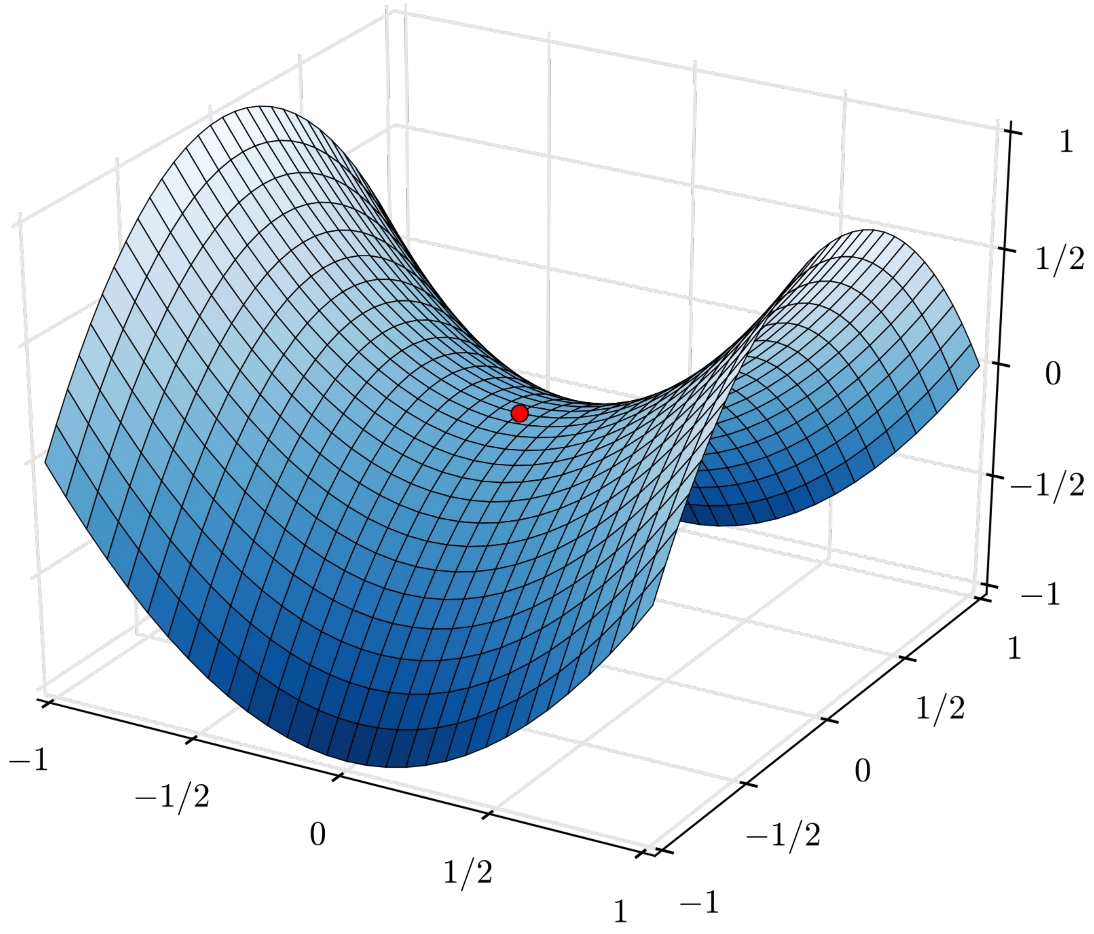


Image Source: Wikipedia

Nash Equilibria of QZSG

- The solutions (**fixed points**) of this minimax define the game's value:

$$u(\alpha^*,\beta^*) = \min_{\beta}\max_{\alpha} u(\alpha,\beta) = \max_{\alpha}\min_{\beta} u(\alpha,\beta)$$

von Neumann's Minimax Thm

- **Nash equilibria** are game states (α^*,β^*) such that neither player has an incentive to change to another state unilaterally:

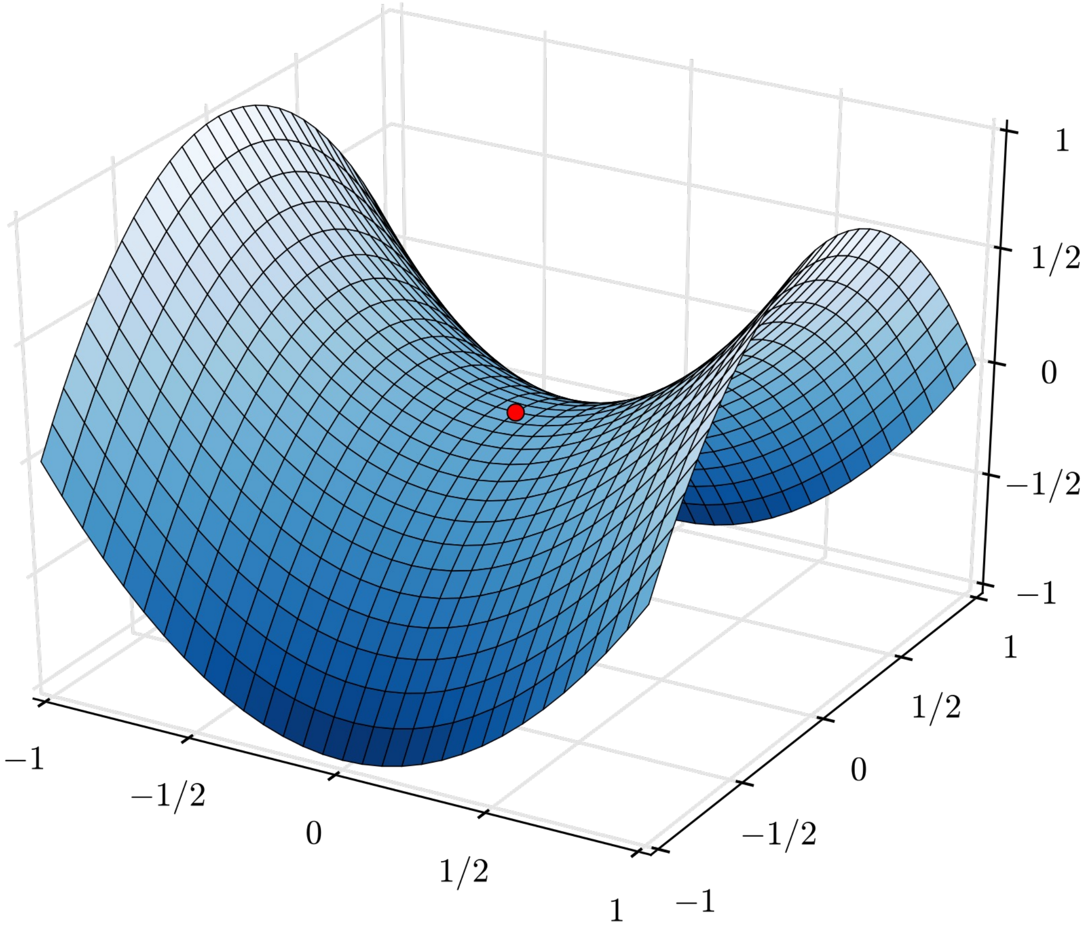


Image Source: Wikipedia

Nash Equilibria of QZSG

- The solutions (**fixed points**) of this minimax define the game's value:

$$u(\alpha^*, \beta^*) = \min_{\beta} \max_{\alpha} u(\alpha, \beta) = \max_{\alpha} \min_{\beta} u(\alpha, \beta)$$

von Neumann's Minimax Thm

- **Nash equilibria** are game states (α^*, β^*) such that neither player has an incentive to change to another state unilaterally:

$$u(\alpha^*, \beta^*) \geq u(\alpha, \beta^*), \qquad \forall \alpha \in \mathcal{A}$$

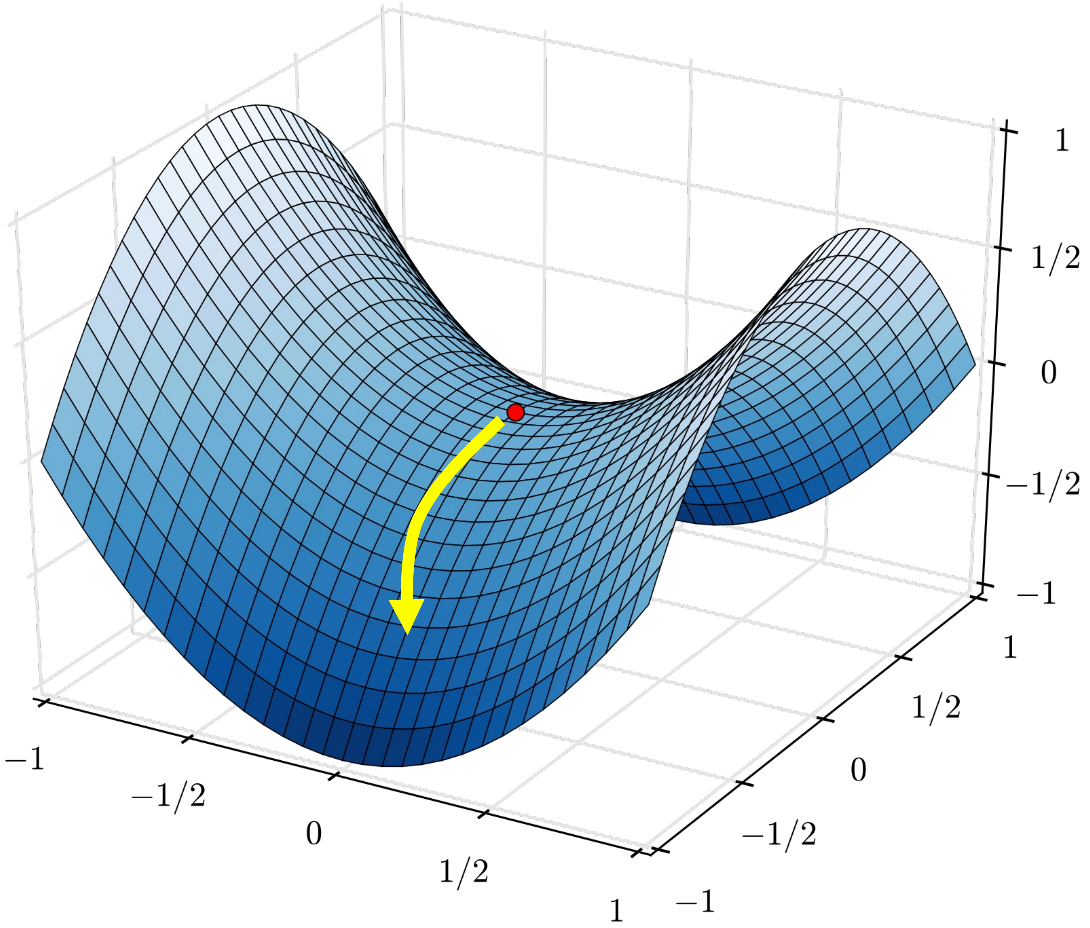


Image Source: Wikipedia

Nash Equilibria of QZSG

- The solutions (**fixed points**) of this minimax define the game's value:

$$u(\alpha^*, \beta^*) = \min_{\beta} \max_{\alpha} u(\alpha, \beta) = \max_{\alpha} \min_{\beta} u(\alpha, \beta)$$

↑ von Neumann's Minimax Thm

- **Nash equilibria** are game states (α^*, β^*) such that neither player has an incentive to change to another state unilaterally:

$$u(\alpha^*, \beta^*) \geq u(\alpha, \beta^*), \qquad \forall \alpha \in \mathcal{A}$$
$$u(\alpha^*, \beta^*) \leq u(\alpha^*, \beta\ ), \qquad \forall \beta \in \mathcal{B}$$

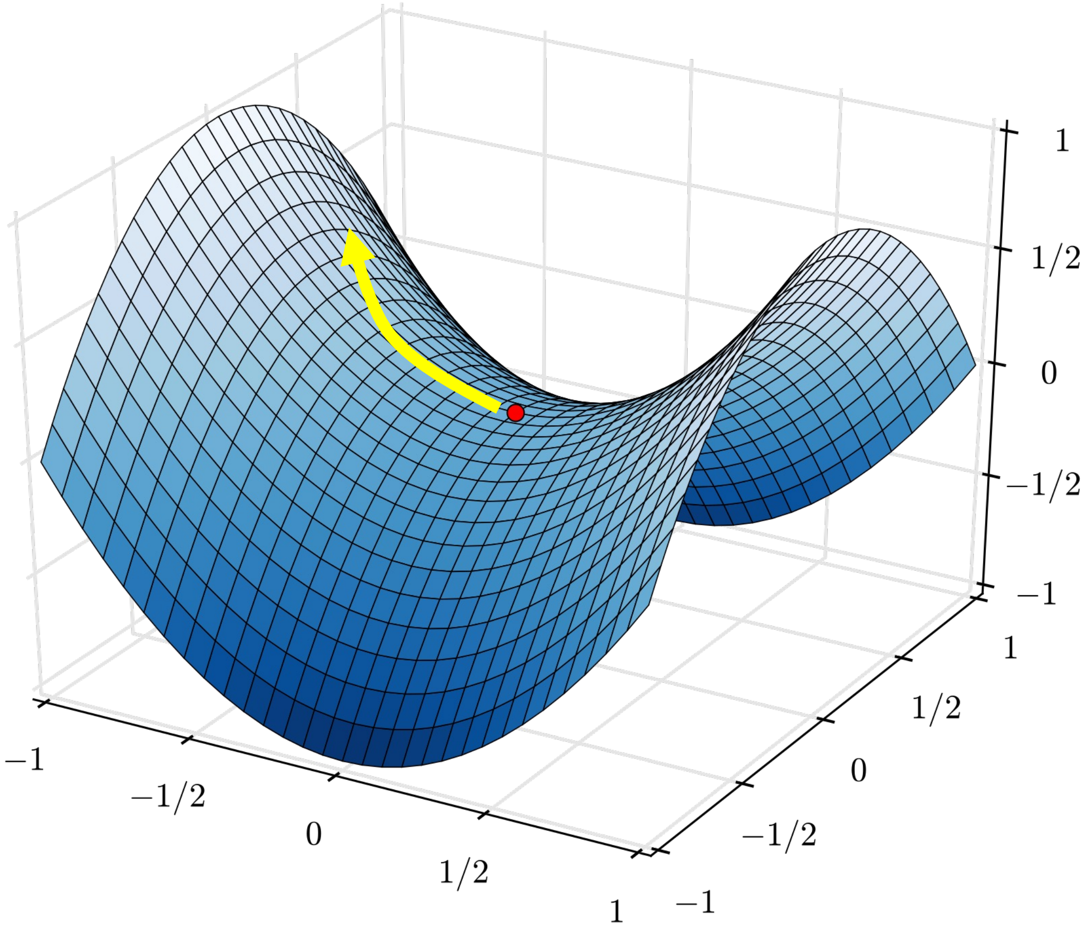


Image Source: Wikipedia

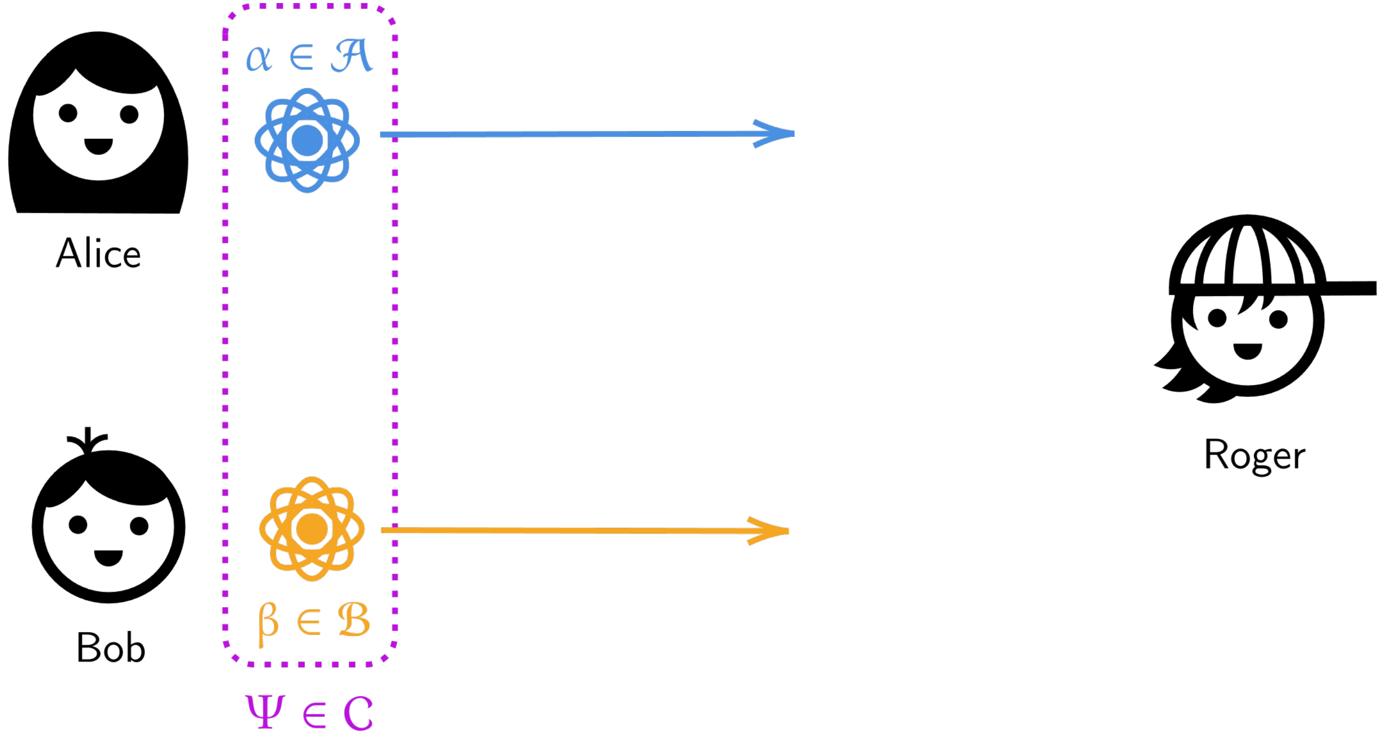
Algorithmic QZSG

Algorithmic QZSG

- Algorithmically, we will view the game as an **online** learning problem

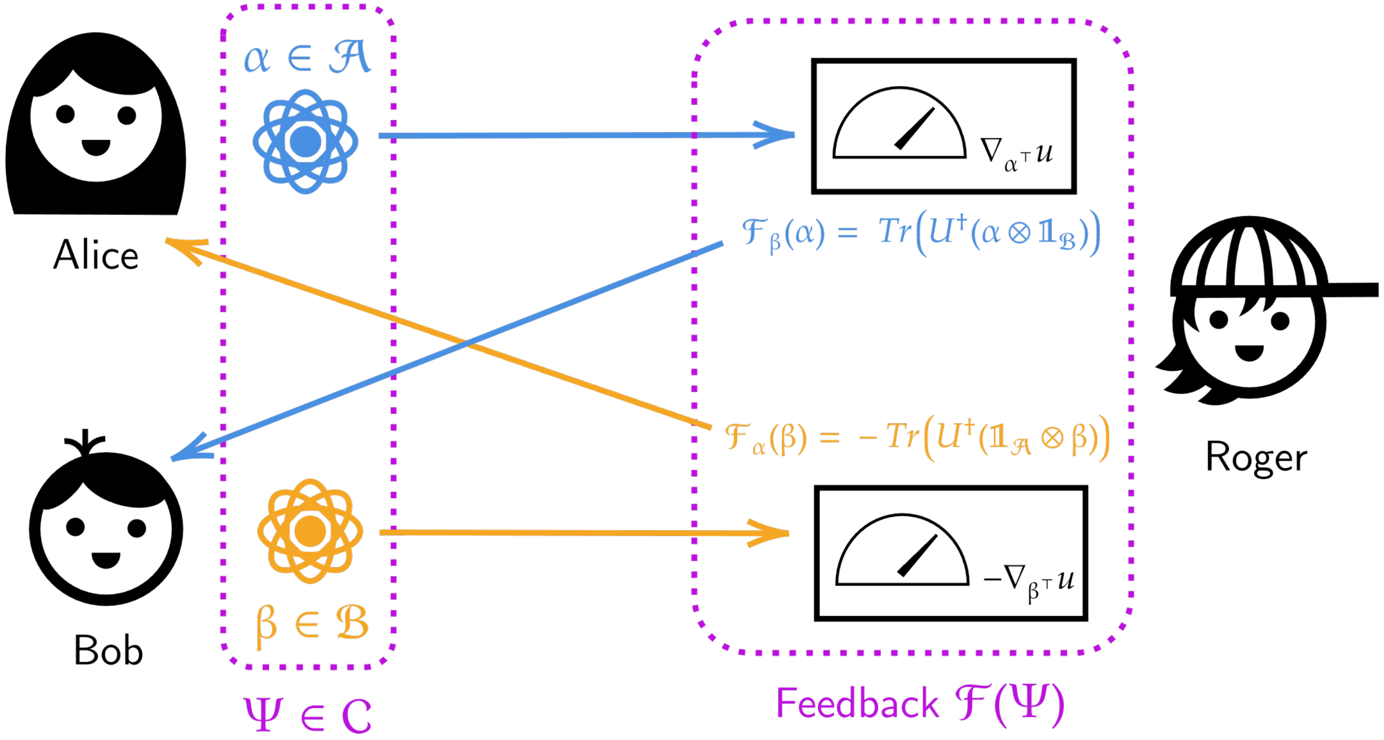
Algorithmic QZSG

- Algorithmically, we will view the game as an **online** learning problem
- In each round, each player **queries** the referee ("oracle") with a state



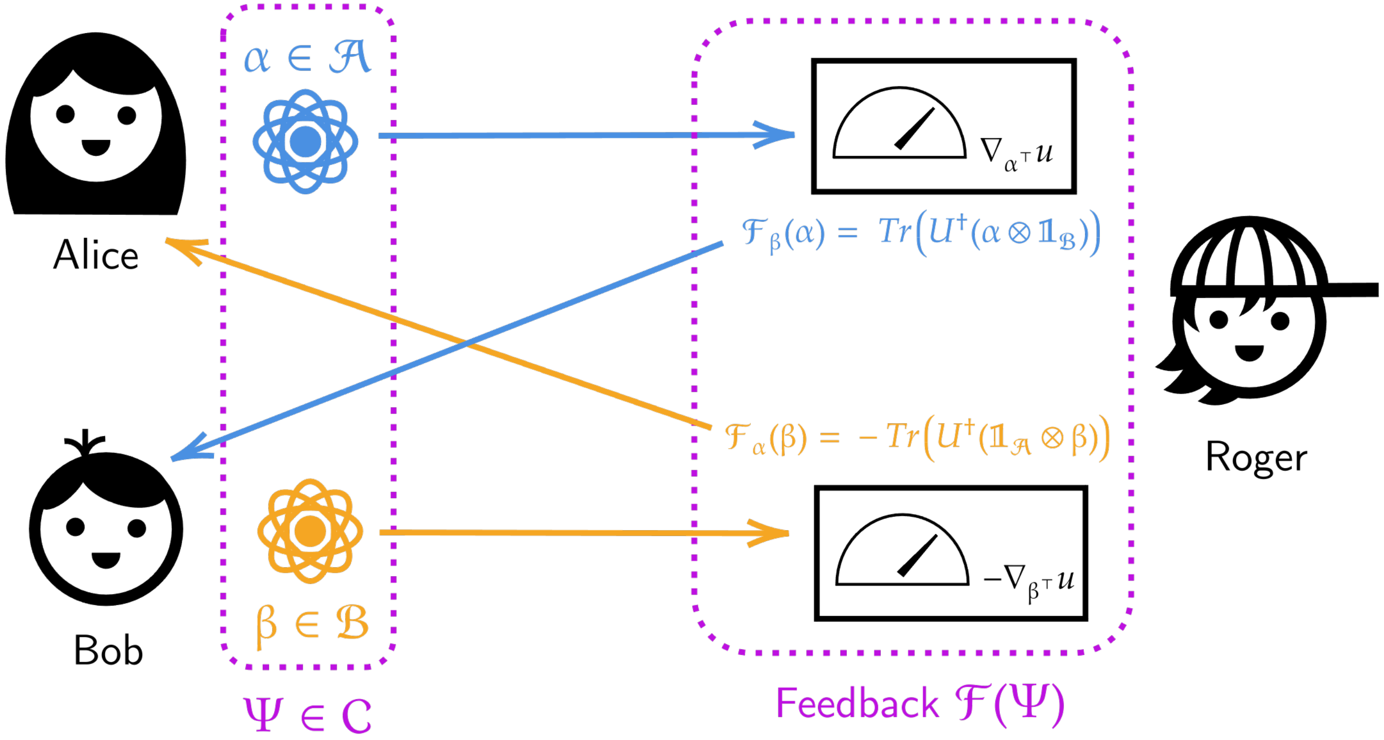
Algorithmic QZSG

- Algorithmically, we will view the game as an **online** learning problem
- In each round, each player **queries** the referee ("oracle") with a state
- The ref returns **feedback** to each player



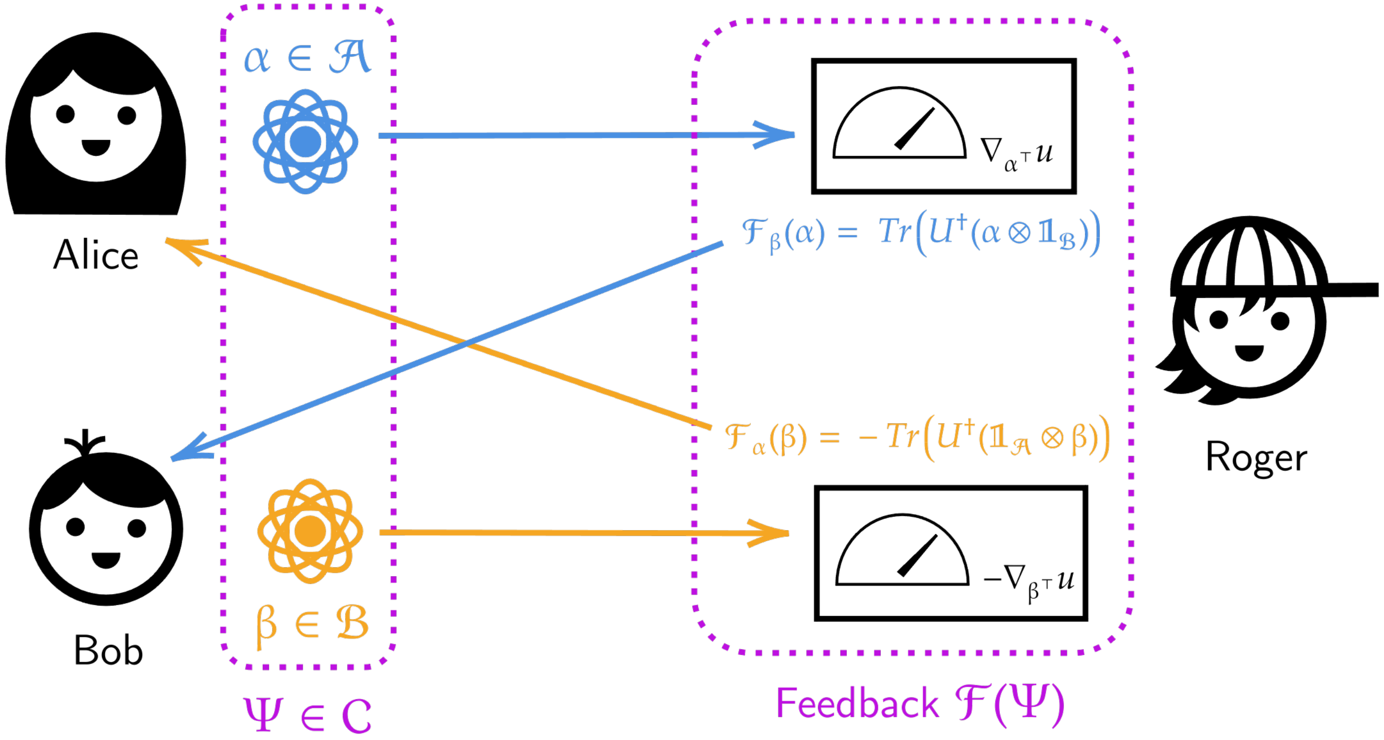
Algorithmic QZSG

- Algorithmically, we will view the game as an **online** learning problem
- In each round, each player **queries** the referee ("oracle") with a state
- The ref returns **feedback** to each player
- The players use this feedback to update and **improve** their states



Algorithmic QZSG

- Algorithmically, we will view the game as an **online** learning problem
- In each round, each player **queries** the referee ("oracle") with a state
- The ref returns **feedback** to each player
- The players use this feedback to update and **improve** their states
- **Goal**: minimize the number of rounds until the players reach an ϵ-approx Nash equilibrium



Superoperator vs Gradient-Based Feedback

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) \;\; = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$
  - Bob's Feedback: $\Xi^{*}(\alpha) = \mathrm{Tr}_{\mathcal{A}}[U(\alpha^{\mathrm{T}} \otimes \mathbb{I}_{\mathcal{B}})]$

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$
  - Bob's Feedback: $\Xi^*(\alpha) = \mathrm{Tr}_{\mathcal{A}}[U(\alpha^{\mathrm{T}} \otimes \mathbb{I}_{\mathcal{B}})]$
- We instead characterize the game's feedback via **gradient-based** operators:

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) \ = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$
  - Bob's Feedback: $\Xi^*(\alpha) = \mathrm{Tr}_{\mathcal{A}}[U(\alpha^{\mathrm{T}} \otimes \mathbb{I}_{\mathcal{B}})]$
- We instead characterize the game's feedback via **gradient-based** operators:
  - Alice's Feedback: $\mathcal{F}_{\alpha}(\beta) = \ \nabla_{\alpha^{\mathrm{T}}} u(\alpha, \beta) = \ \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) \quad = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$
  - Bob's Feedback: $\Xi^*(\alpha) = \mathrm{Tr}_{\mathcal{A}}[U(\alpha^{\mathrm{T}} \otimes \mathbb{I}_{\mathcal{B}})]$
- We instead characterize the game's feedback via **gradient-based** operators:
  - Alice's Feedback: $\mathcal{F}_{\alpha}(\beta) = \quad \nabla_{\alpha^{\mathrm{T}}}\, u(\alpha, \beta) = \quad \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
  - Bob's Feedback: $\mathcal{F}_{\beta}(\alpha) = -\nabla_{\beta^{\mathrm{T}}}\, u(\alpha, \beta) = -\mathrm{Tr}_{\mathcal{A}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) \ = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$
  - Bob's Feedback: $\Xi^*(\alpha) = \mathrm{Tr}_{\mathcal{A}}[U(\alpha^{\mathrm{T}} \otimes \mathbb{I}_{\mathcal{B}})]$
- We instead characterize the game's feedback via **gradient-based** operators:
  - Alice's Feedback: $\mathcal{F}_\alpha(\beta) = \ \nabla_{\alpha^{\mathrm{T}}} u(\alpha, \beta) = \ \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
  - Bob's Feedback: $\mathcal{F}_\beta(\alpha) = -\nabla_{\beta^{\mathrm{T}}} u(\alpha, \beta) = -\mathrm{Tr}_{\mathcal{A}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
- The two characterizations are **equivalent**:

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$
  - Bob's Feedback: $\Xi^*(\alpha) = \mathrm{Tr}_{\mathcal{A}}[U(\alpha^{\mathrm{T}} \otimes \mathbb{I}_{\mathcal{B}})]$
- We instead characterize the game's feedback via **gradient-based** operators:
  - Alice's Feedback: $\mathcal{F}_\alpha(\beta) = \nabla_{\alpha^{\mathrm{T}}} u(\alpha, \beta) = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
  - Bob's Feedback: $\mathcal{F}_\beta(\alpha) = -\nabla_{\beta^{\mathrm{T}}} u(\alpha, \beta) = -\mathrm{Tr}_{\mathcal{A}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
- The two characterizations are **equivalent**:
  - Feedback: $\mathcal{F}_\alpha(\beta) = \Xi(\beta^{\mathrm{T}})$, $\mathcal{F}_\beta(\alpha) = -\Xi^*(\alpha^{\mathrm{T}})$

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$
  - Bob's Feedback: $\Xi^*(\alpha) = \mathrm{Tr}_{\mathcal{A}}[U(\alpha^{\mathrm{T}} \otimes \mathbb{I}_{\mathcal{B}})]$
- We instead characterize the game's feedback via **gradient-based** operators:
  - Alice's Feedback: $\mathcal{F}_\alpha(\beta) = \nabla_{\alpha^{\mathrm{T}}} u(\alpha, \beta) = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
  - Bob's Feedback: $\mathcal{F}_\beta(\alpha) = -\nabla_{\beta^{\mathrm{T}}} u(\alpha, \beta) = -\mathrm{Tr}_{\mathcal{A}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
- The two characterizations are **equivalent**:
  - Feedback: $\mathcal{F}_\alpha(\beta) = \Xi(\beta^{\mathrm{T}}), \ \mathcal{F}_\beta(\alpha) = -\Xi^*(\alpha^{\mathrm{T}})$
  - Alice's expected payoff : $u(\alpha, \beta) = \mathrm{Tr}[\alpha\, \mathcal{F}_\alpha(\beta)] = \mathrm{Tr}[\alpha\, \Xi(\beta^{\mathrm{T}})]$

Superoperator vs Gradient-Based Feedback

- Previous work on QZSG characterized the game's feedback via **superoperators**:
  - Alice's Feedback: $\Xi(\beta) = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta^{\mathrm{T}})]$
  - Bob's Feedback: $\Xi^*(\alpha) = \mathrm{Tr}_{\mathcal{A}}[U(\alpha^{\mathrm{T}} \otimes \mathbb{I}_{\mathcal{B}})]$
- We instead characterize the game's feedback via **gradient-based** operators:
  - Alice's Feedback: $\mathcal{F}_\alpha(\beta) = \nabla_{\alpha^{\mathrm{T}}} u(\alpha, \beta) = \mathrm{Tr}_{\mathcal{B}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
  - Bob's Feedback: $\mathcal{F}_\beta(\alpha) = -\nabla_{\beta^{\mathrm{T}}} u(\alpha, \beta) = -\mathrm{Tr}_{\mathcal{A}}[U(\mathbb{I}_{\mathcal{A}} \otimes \beta)]$
- The two characterizations are **equivalent**:
  - Feedback: $\mathcal{F}_\alpha(\beta) = \Xi(\beta^{\mathrm{T}})$, $\mathcal{F}_\beta(\alpha) = -\Xi^*(\alpha^{\mathrm{T}})$
  - Alice's expected payoff : $u(\alpha, \beta) = \mathrm{Tr}[\alpha\, \mathcal{F}_\alpha(\beta)] = \mathrm{Tr}[\alpha\, \Xi(\beta^{\mathrm{T}})]$
  - Bob's expected payoff: $-u(\alpha, \beta) = \mathrm{Tr}[\beta\, \mathcal{F}_\beta(\alpha)] = -\mathrm{Tr}[\beta\, \Xi^*(\alpha^{\mathrm{T}})]$

Formulating QZSG as a Variational Inequality

Formulating QZSG as a Variational Inequality

- With gradient-based feedback, the expected payoff is a **directional derivative**:

Formulating QZSG as a Variational Inequality

- With gradient-based feedback, the expected payoff is a **directional derivative**:
  - In Alice's direction: $u(\alpha,\beta) = \mathrm{Tr}[\alpha\, \mathcal{F}_\alpha(\beta)] = \mathrm{Tr}\left[\alpha\, \nabla_{\alpha^\mathrm{T}}\, u(\alpha,\beta)\right]$

Formulating QZSG as a Variational Inequality

- With gradient-based feedback, the expected payoff is a **directional derivative**:
  - In Alice's direction: $u(\alpha,\beta) = \quad \mathrm{Tr}[\alpha\, \mathcal{F}_\alpha(\beta)] = \mathrm{Tr}\left[\alpha\, \nabla_{\alpha^{\mathrm{T}}}\, u(\alpha,\beta)\right]$
  - In Bob's direction: $u(\alpha,\beta) = -\mathrm{Tr}\left[\beta\, \mathcal{F}_\beta(\alpha)\right] = \mathrm{Tr}\left[\beta\, \nabla_{\beta^{\mathrm{T}}}\, u(\alpha,\beta)\ \right]$

Formulating QZSG as a Variational Inequality

- With gradient-based feedback, the expected payoff is a **directional derivative**:
  - In Alice's direction: $u(\alpha,\beta) = \mathrm{Tr}[\alpha\,\mathcal{F}_\alpha(\beta)] = \mathrm{Tr}\left[\alpha\,\nabla_{\alpha^\mathrm{T}}\,u(\alpha,\beta)\right]$
  - In Bob's direction: $u(\alpha,\beta) = -\mathrm{Tr}\left[\beta\,\mathcal{F}_\beta(\alpha)\right] = \mathrm{Tr}\left[\beta\,\nabla_{\beta^\mathrm{T}}\,u(\alpha,\beta)\right]$

$$u(\Psi) = \mathrm{Tr}[\Psi\,\mathcal{F}(\Psi)] = \mathrm{Tr}\left[\Psi\,\nabla_{\Psi^\mathrm{T}} u(\Psi)\right]$$

Formulating QZSG as a Variational Inequality

- With gradient-based feedback, the expected payoff is a **directional derivative**:
  - In Alice's direction: $u(\alpha,\beta) = \mathrm{Tr}[\alpha\,\mathcal{F}_\alpha(\beta)] = \mathrm{Tr}[\alpha\,\nabla_{\alpha^\mathrm{T}}\,u(\alpha,\beta)]$
  - In Bob's direction: $u(\alpha,\beta) = -\mathrm{Tr}[\beta\,\mathcal{F}_\beta(\alpha)] = \mathrm{Tr}[\beta\,\nabla_{\beta^\mathrm{T}}\,u(\alpha,\beta)]$

  $$u(\Psi) = \mathrm{Tr}[\Psi\,\mathcal{F}(\Psi)] = \mathrm{Tr}[\Psi\,\nabla_{\Psi^\mathrm{T}} u(\Psi)]$$

- With a directional derivative, we can characterize the game's equilibria as solutions of the **variational inequality (VI)**:

$$\mathrm{Tr}[(\Psi - \Psi^*)\,\mathcal{F}(\Psi^*)] \leq 0, \quad \forall\,\Psi \in \mathcal{A} \oplus \mathcal{B}$$

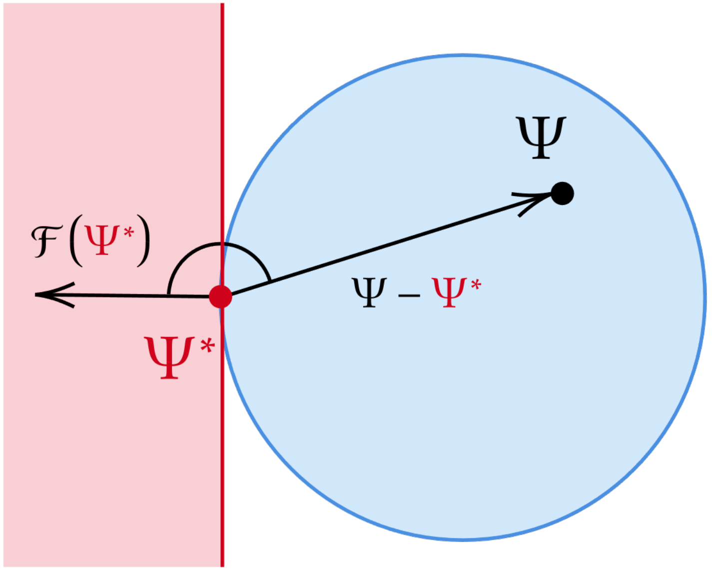
Formulating QZSG as a Variational Inequality

- With gradient-based feedback, the expected payoff is a **directional derivative**:
  - In Alice's direction: $u(\alpha,\beta) = \mathrm{Tr}[\alpha\,\mathcal{F}_\alpha(\beta)] = \mathrm{Tr}[\alpha\,\nabla_{\alpha^\mathrm{T}}\,u(\alpha,\beta)]$
  - In Bob's direction: $u(\alpha,\beta) = -\mathrm{Tr}[\beta\,\mathcal{F}_\beta(\alpha)] = \mathrm{Tr}[\beta\,\nabla_{\beta^\mathrm{T}}\,u(\alpha,\beta)]$

  $$u(\Psi) = \mathrm{Tr}[\Psi\,\mathcal{F}(\Psi)] = \mathrm{Tr}[\Psi\,\nabla_{\Psi^\mathrm{T}} u(\Psi)]$$

- With a directional derivative, we can characterize the game's equilibria as solutions of the **variational inequality (VI)**:

$$\mathrm{Tr}[(\Psi - \Psi^*)\,\mathcal{F}(\Psi^*)] \leq 0, \quad \forall\, \Psi \in \mathcal{A} \oplus \mathcal{B}$$



Formulating QZSG as a Variational Inequality

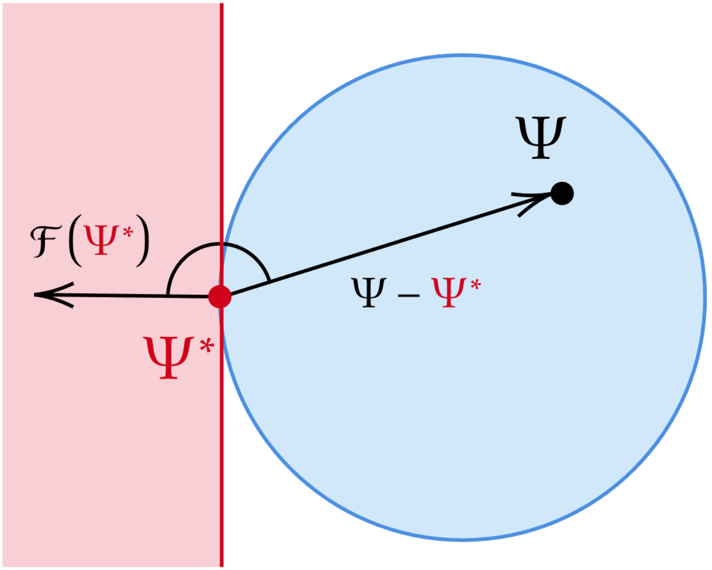
- With gradient-based feedback, the expected payoff is a **directional derivative**:
  - In Alice's direction: $u(\alpha,\beta) = \mathrm{Tr}[\alpha\, \mathcal{F}_\alpha(\beta)] = \mathrm{Tr}\big[\alpha\, \nabla_{\alpha^\mathsf{T}}\, u(\alpha,\beta)\big]$
  - In Bob's direction: $u(\alpha,\beta) = -\mathrm{Tr}\big[\beta\, \mathcal{F}_\beta(\alpha)\big] = \mathrm{Tr}\big[\beta\, \nabla_{\beta^\mathsf{T}}\, u(\alpha,\beta)\big]$

  $\Big\}\; u(\Psi) = \mathrm{Tr}[\Psi\, \mathcal{F}(\Psi)] = \mathrm{Tr}\big[\Psi\, \nabla_{\Psi^\mathsf{T}} u(\Psi)\big]$

- With a directional derivative, we can characterize the game's equilibria as solutions of the **variational inequality (VI)**:

$$\mathrm{Tr}[(\Psi - \Psi^*)\, \mathcal{F}(\Psi^*)] \leq 0, \quad \forall\, \Psi \in \mathcal{A} \oplus \mathcal{B}$$

- We further prove that $\mathcal{F}(\Psi)$ is **monotone** and **Lipschitz**, which offers additional structure about the game that we can use to **leverage efficient classical algorithms** for solving such VIs.



A Quadratic Speedup in Finding Nash Equilibria of Quantum Zero-Sum Games

Gradient to Mirror Descent

Gradient to Mirror Descent

- Classical **gradient descent (GD)**: $x_{t+1} = x_t - \eta \, \nabla F(x_t)$

Gradient to Mirror Descent

- Classical **gradient descent (GD)**: $x_{t+1} = x_t - \eta \, \nabla F(x_t)$

- Equivalently, GD minimizes the 1st-order approx of F with **Euclidean regularizer** :

$$x_{t+1} = \underset{x}{\operatorname{argmin}} \left(F(x_t) + \nabla F(x_t)^T (x - x_t) + \frac{1}{2\eta} \, \|x - x_t\|^2 \right)$$

Gradient to Mirror Descent

- Classical **gradient descent (GD)**: $x_{t+1} = x_t - \eta \, \nabla F(x_t)$

- Equivalently, GD minimizes the 1st-order approx of F with **Euclidean regularizer** :

$$x_{t+1} = \underset{x}{\operatorname{argmin}} \left(F(x_t) + \nabla F(x_t)^T (x - x_t) + \frac{1}{2\eta} \, \|x - x_t\|^2 \right)$$

- To generalize GD to other regularizers h, perform **mirror descent (MD)**:

$$x_{t+1} = (\nabla h)^{-1} [\nabla h(x_t) - \eta \, \nabla F(x_t) \,]$$

Gradient to Mirror Descent

- Classical **gradient descent (GD)**: $x_{t+1} = x_t - \eta \, \nabla F(x_t)$

- Equivalently, GD minimizes the 1st-order approx of F with **Euclidean regularizer** :

$$x_{t+1} = \underset{x}{\operatorname{argmin}} \left(F(x_t) + \nabla F(x_t)^T (x - x_t) + \frac{1}{2\eta} \, \|x - x_t\|^2 \right)$$

- To generalize GD to other regularizers h, perform **mirror descent (MD)**:

$$x_{t+1} = (\nabla h)^{-1} [\nabla h(x_t) - \eta \, \nabla F(x_t) \,]$$

For $h(y) = \frac{1}{2} \|y\|^2$, $\nabla \mathrm{h}(\mathrm{y}) = \mathrm{y}$

Gradient to Mirror Descent

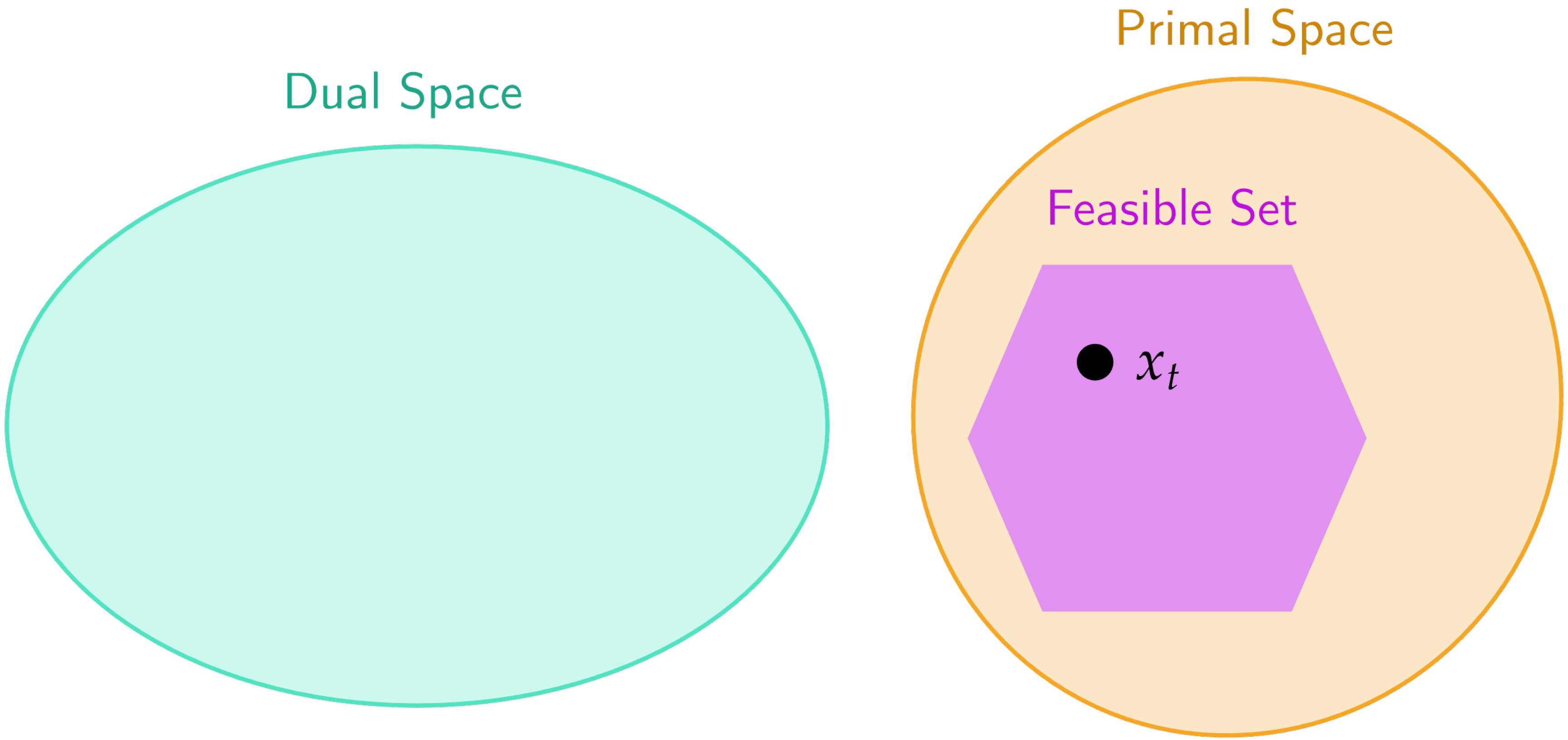
- To generalize GD to other regularizers h, perform **mirror descent (MD)**:

$$x_{t+1} = (\nabla h)^{-1}[\nabla h(x_t) - \eta \, \nabla F(x_t)\,]$$

Gradient to Mirror Descent

- To generalize GD to other regularizers h, perform **mirror descent (MD)**:

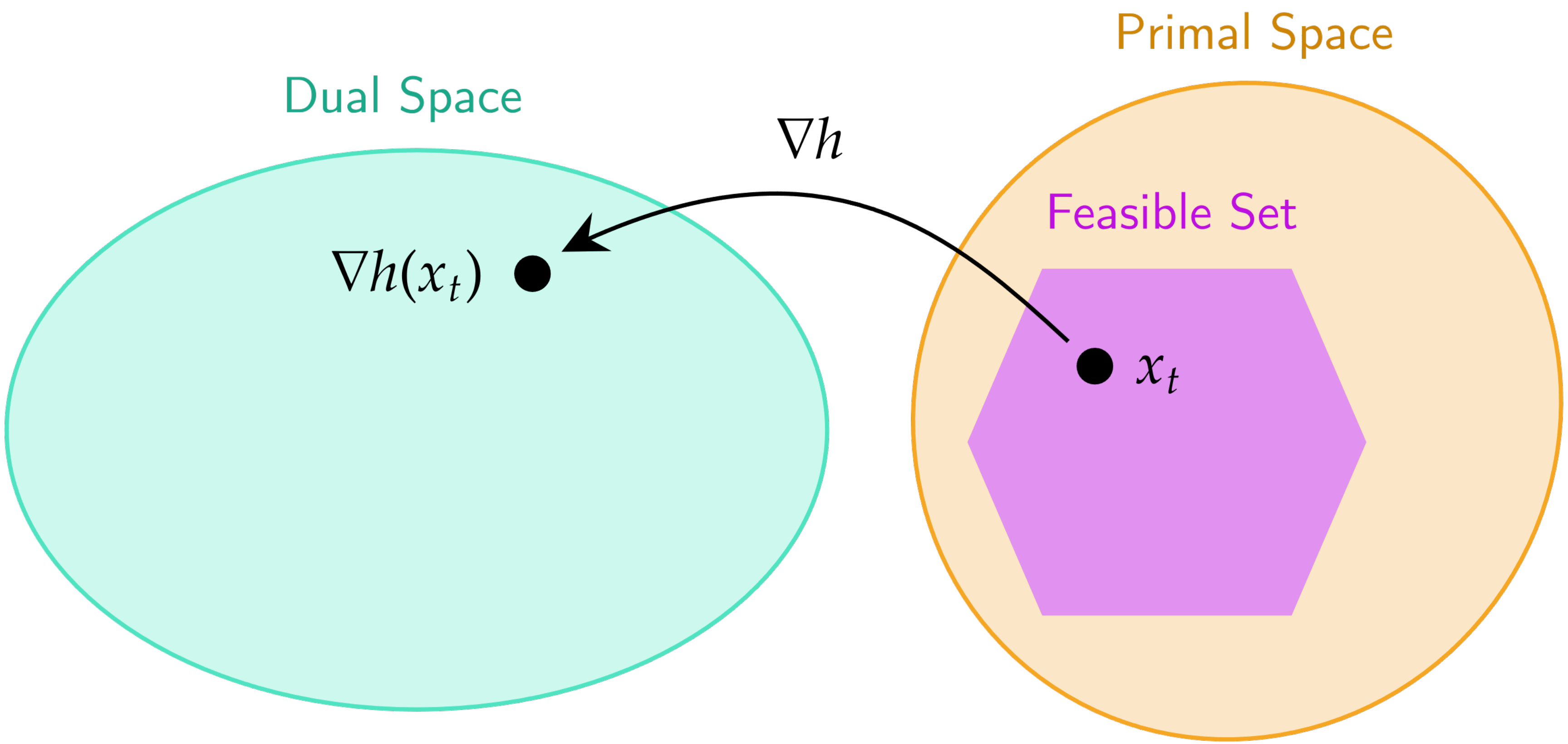
$$x_{t+1} = (\nabla h)^{-1}[\nabla h(x_t) - \eta\, \nabla F(x_t)\,]$$



Gradient to Mirror Descent

- To generalize GD to other regularizers h, perform **mirror descent (MD)**:

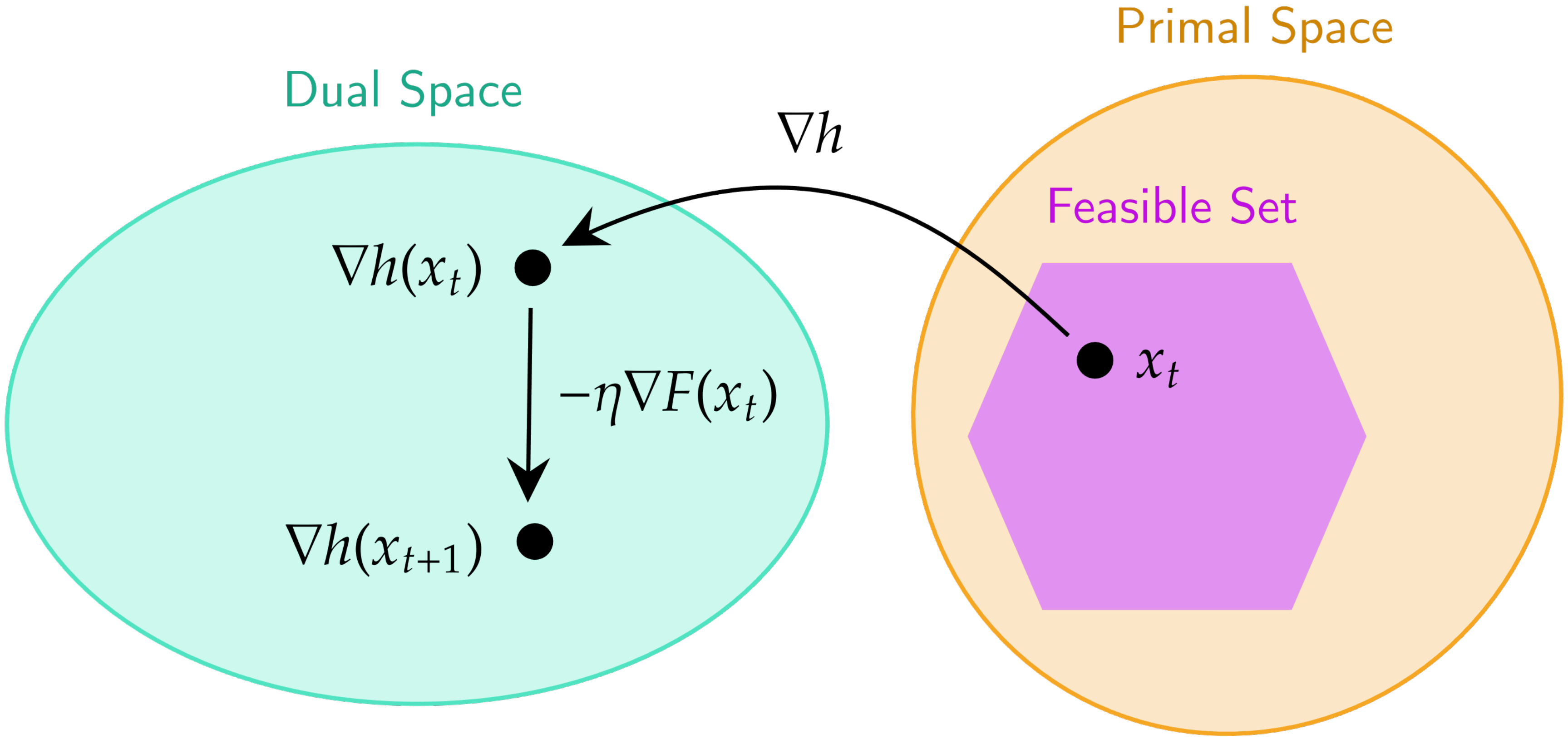
$$x_{t+1} = (\nabla h)^{-1}[\nabla h(x_t) - \eta \, \nabla F(x_t)\,]$$



Gradient to Mirror Descent

- To generalize GD to other regularizers h, perform **mirror descent (MD)**:

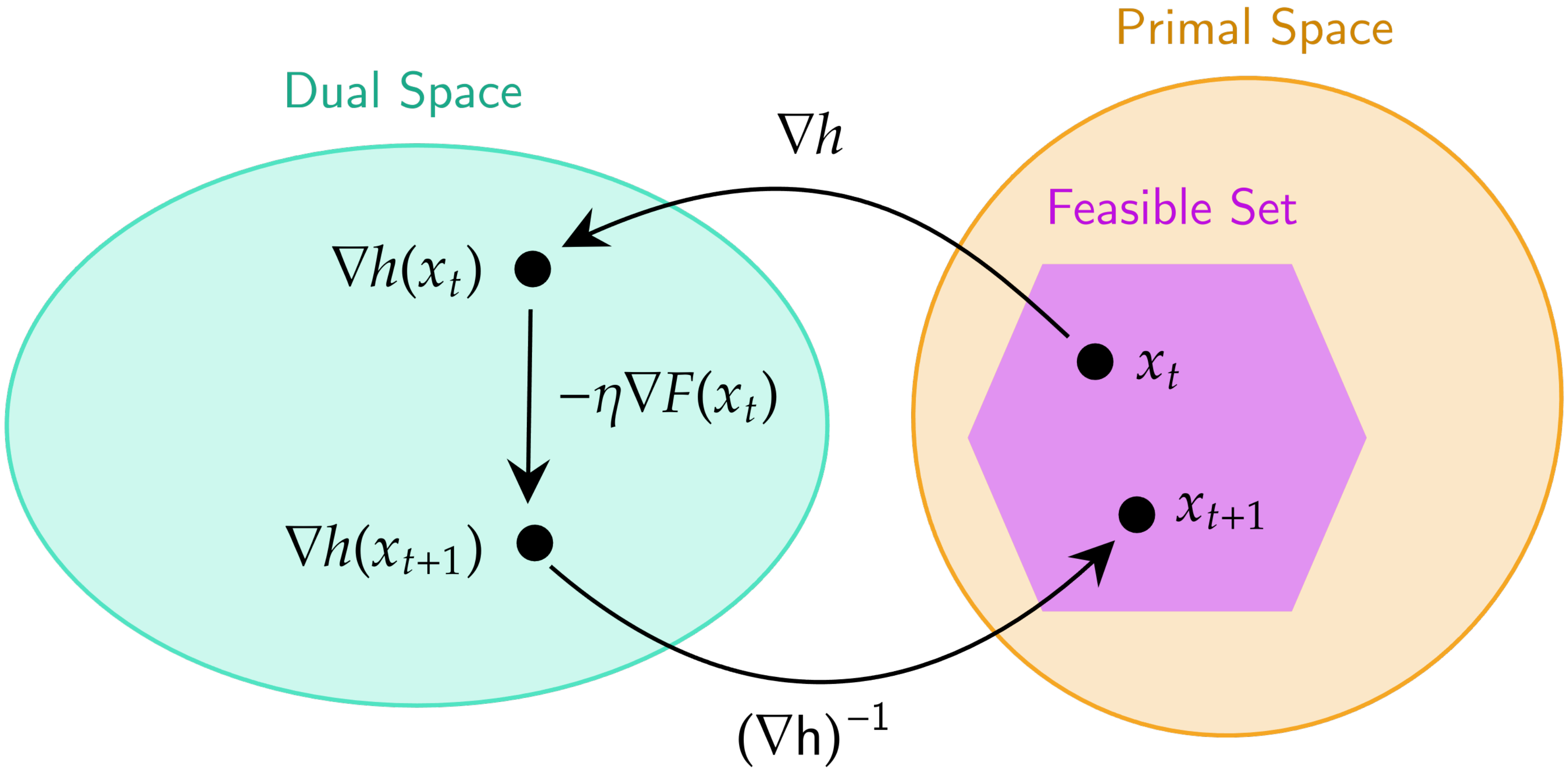
$$x_{t+1} = (\nabla h)^{-1}[\nabla h(x_t) - \eta\, \nabla F(x_t)\,]$$



Gradient to Mirror Descent

- To generalize GD to other regularizers h, perform **mirror descent (MD)**:

$$x_{t+1} = (\nabla h)^{-1}[\nabla h(x_t) - \eta\, \nabla F(x_t)\,]$$



Prior Work: The Jain-Watrous Algorithm [JW09]

Prior Work: The Jain-Watrous Algorithm [JW09]

- In 2009, Jain and Watrous proposed the **Matrix Multiplicative Weight Updates (MMWU)** algorithm, with the following update in each round t :

$$\alpha_t = \Lambda\left(\eta \sum_{i=0}^{t-1} \Xi(\beta_i^{\mathrm{T}})\right), \qquad \beta_t = \Lambda\left(-\eta \sum_{i=0}^{t-1} \Xi^*(\alpha_i^{\mathrm{T}})\right), \qquad \text{where } \Lambda(x) = \frac{\exp(x)}{\mathrm{Tr}(\exp(x))}$$

Prior Work: The Jain-Watrous Algorithm [JW09]

- In 2009, Jain and Watrous proposed the **Matrix Multiplicative Weight Updates (MMWU)** algorithm, with the following update in each round t :

$$\alpha_t = \Lambda\left(\eta \sum_{i=0}^{t-1} \Xi(\beta_i^{\mathrm{T}})\right), \qquad \beta_t = \Lambda\left(-\eta \sum_{i=0}^{t-1} \Xi^*(\alpha_i^{\mathrm{T}})\right), \qquad \text{where } \Lambda(x) = \frac{\exp(x)}{\mathrm{Tr}(\exp(x))}$$

- We show MMWU is **"Lazy" Mirror Descent**, with a von Neumann **entropy** regularizer:

$$h(\psi) = Tr[\psi \log \psi]$$

Prior Work: The Jain-Watrous Algorithm [JW09]

- In 2009, Jain and Watrous proposed the **Matrix Multiplicative Weight Updates (MMWU)** algorithm, with the following update in each round t :

$$\alpha_t = \Lambda\left(\eta \sum_{i=0}^{t-1} \Xi(\beta_i^{\mathrm{T}})\right), \qquad \beta_t = \Lambda\left(-\eta \sum_{i=0}^{t-1} \Xi^*(\alpha_i^{\mathrm{T}})\right), \qquad \text{where } \Lambda(x) = \frac{\exp(x)}{\mathrm{Tr}(\exp(x))}$$

- We show MMWU is **"Lazy" Mirror Descent**, with a von Neumann **entropy** regularizer:

$$h(\psi) = Tr[\psi \log \psi]$$

- Like classical MWU, they prove an $\mathcal{O}(1/\epsilon^2)$ convergence.

Prior Work: The Jain-Watrous Algorithm [JW09]

- In 2009, Jain and Watrous proposed the **Matrix Multiplicative Weight Updates (MMWU)** algorithm, with the following update in each round t :

$$\alpha_t = \Lambda\left(\eta \sum_{i=0}^{t-1} \Xi(\beta_i^{\mathrm{T}})\right), \qquad \beta_t = \Lambda\left(-\eta \sum_{i=0}^{t-1} \Xi^*(\alpha_i^{\mathrm{T}})\right), \qquad \text{where } \Lambda(x) = \frac{\exp(x)}{\mathrm{Tr}(\exp(x))}$$

- We show MMWU is **"Lazy" Mirror Descent**, with a von Neumann **entropy** regularizer:

$$h(\psi) = Tr[\psi \log \psi]$$

- Like classical MWU, they prove an $\mathcal{O}(1/\epsilon^2)$ convergence.
  - However, in classical games, while this is optimal for classical **black-box** optimization, Nemirovski [N04] showed that $\mathcal{O}(1/\epsilon)$ can be achieved for monotone VIs.

Achieving Speedup: Proximal Steps and Optimism

Achieving Speedup: Proximal Steps and Optimism

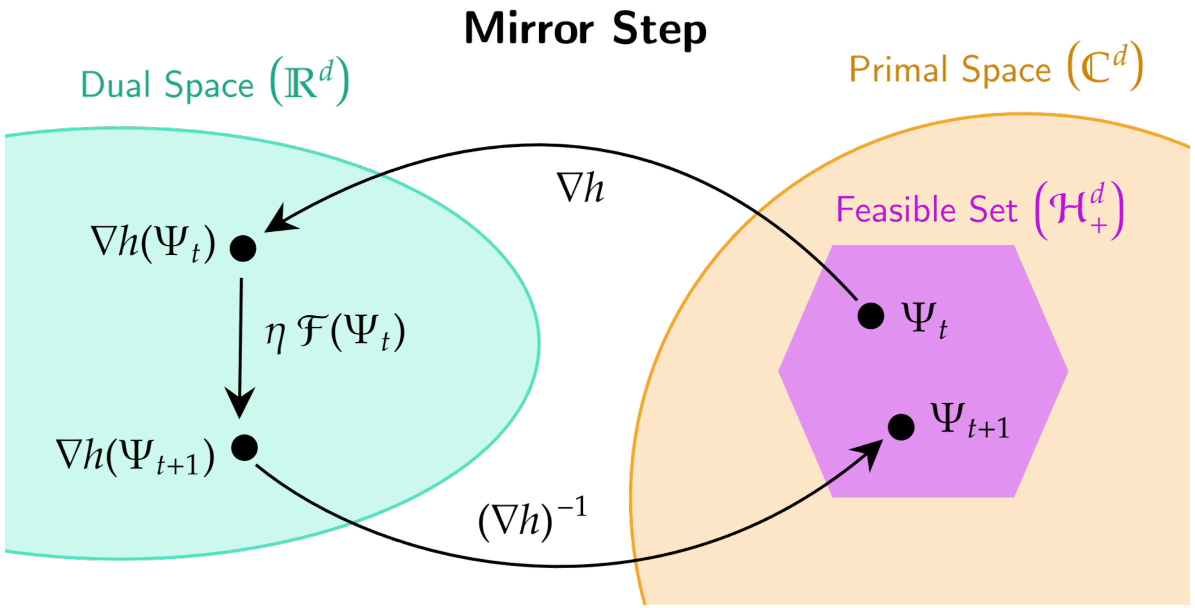
- To improve upon the MMWU, we leverage **proximal** instead of mirror steps

Achieving Speedup: Proximal Steps and Optimism

- To improve upon the MMWU, we leverage **proximal** instead of mirror steps
  - Proximal steps introduce "**momentum**"

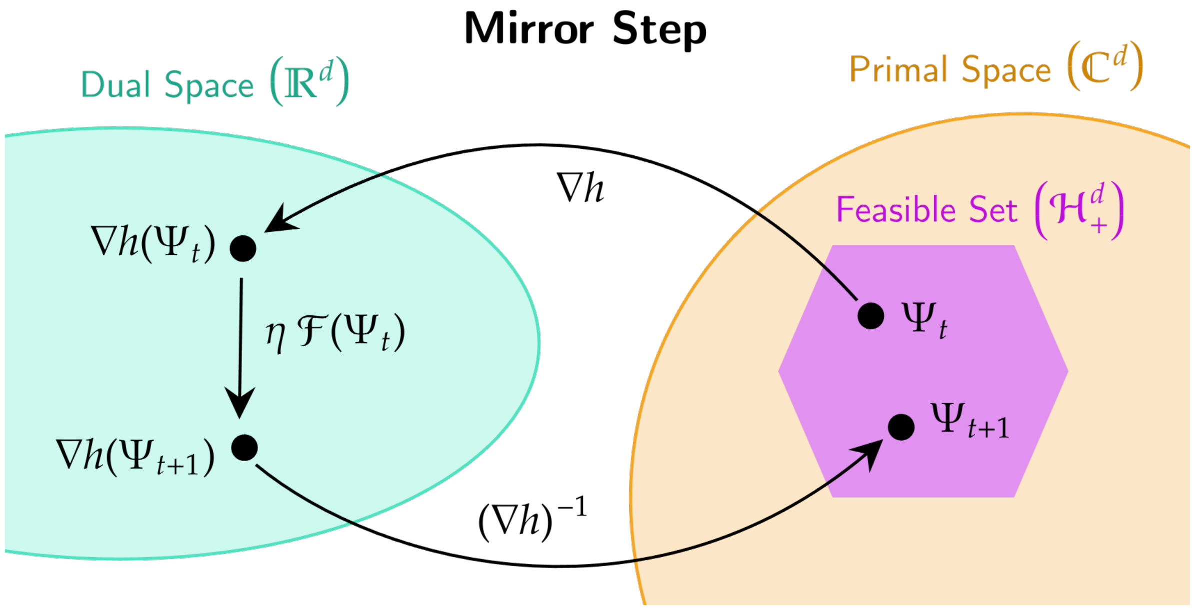
Achieving Speedup: Proximal Steps and Optimism

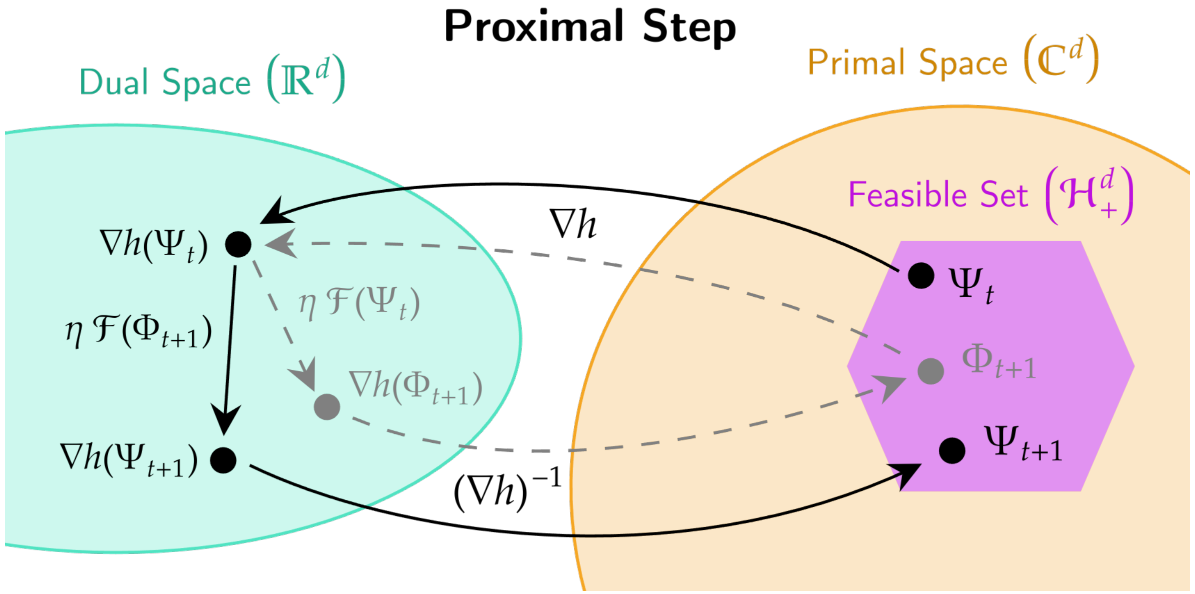
- To improve upon the MMWU, we leverage **proximal** instead of mirror steps
  - Proximal steps introduce "**momentum**"



Achieving Speedup: Proximal Steps and Optimism

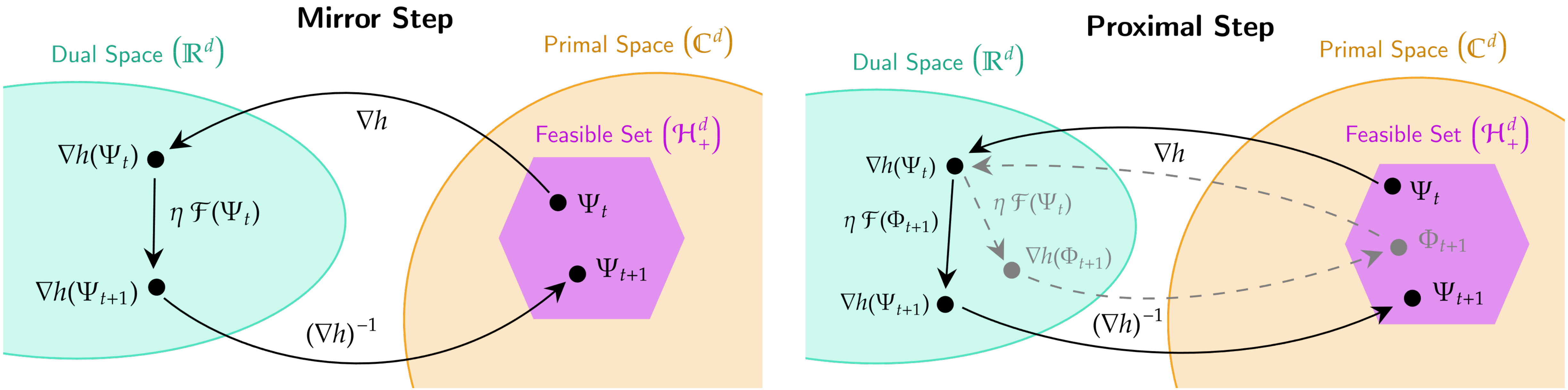
- To improve upon the MMWU, we leverage **proximal** instead of mirror steps
  - Proximal steps introduce "**momentum**"





Achieving Speedup: Proximal Steps and Optimism

- To improve upon the MMWU, we leverage **proximal** instead of mirror steps
  - Proximal steps introduce "**momentum**"



- We further leverage **"optimism"** to reduce the total number of oracle calls from 2 to 1

Our Proposal: Optimistic Matrix Multiplicative Weight Updates

Our Proposal: Optimistic Matrix Multiplicative Weight Updates

- We leverage proximal maps and optimism in our proposal of **Optimistic Matrix Multiplicative Weight Updates (OMMWU)** :

Our Proposal: Optimistic Matrix Multiplicative Weight Updates

- We leverage proximal maps and optimism in our proposal of **Optimistic Matrix Multiplicative Weight Updates (OMMWU)** :

OMMWU

$$\Phi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_t))$$

$$\Psi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_{t+1}))$$

Our Proposal: Optimistic Matrix Multiplicative Weight Updates

- We leverage proximal maps and optimism in our proposal of **Optimistic Matrix Multiplicative Weight Updates (OMMWU)** :

OMMWU

$$\Phi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_t))$$

$$\Psi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_{t+1}))$$

MMWU

$$\Psi_{t+1} = \Lambda\Big(\eta\, \Sigma_{i=0}^{t}\, \mathcal{F}(\Psi_i)\Big)$$

Our Proposal: Optimistic Matrix Multiplicative Weight Updates

- We leverage proximal maps and optimism in our proposal of **Optimistic Matrix Multiplicative Weight Updates (OMMWU)** :

OMMWU

$$\Phi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_t))$$
$$\Psi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_{t+1}))$$

MMWU

$$\Psi_{t+1} = \Lambda\left(\eta\, \Sigma_{i=0}^{t}\, \mathcal{F}(\Psi_i)\right)$$

- Like classical OMWU, we prove an $\mathcal{O}(1/\epsilon)$ convergence.

Our Proposal: Optimistic Matrix Multiplicative Weight Updates

- We leverage proximal maps and optimism in our proposal of **Optimistic Matrix Multiplicative Weight Updates (OMMWU)** :

OMMWU

$$\Phi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_t))$$
$$\Psi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_{t+1}))$$

MMWU

$$\Psi_{t+1} = \Lambda\Big(\eta \, \Sigma_{i=0}^{t} \, \mathcal{F}(\Psi_i) \Big)$$

- Like classical OMWU, we prove an $\mathcal{O}(1/\epsilon)$ convergence.
  - The proof follows the proof structure of [EN20] for monotone VIs.

Our Proposal: Optimistic Matrix Multiplicative Weight Updates

- We leverage proximal maps and optimism in our proposal of **Optimistic Matrix Multiplicative Weight Updates (OMMWU)** :

OMMWU

$$\Phi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_t))$$
$$\Psi_{t+1} = \Lambda(\log \Phi_t + \eta\, \mathcal{F}(\Psi_{t+1}))$$

MMWU

$$\Psi_{\mathrm{t+1}} = \Lambda\left(\eta\, \Sigma_{i=0}^{t}\, \mathcal{F}(\Psi_i)\right)$$

- Like classical OMWU, we prove an $\mathcal{O}(1/\epsilon)$ convergence.
  - The proof follows the proof structure of [EN20] for monotone VIs.
  - We leverage notions of strong convexity, smoothness, and Fenchel conjugacy.

QZSG Algorithm Design

QZSG Algorithm Design

Matrix Multiplicative Weights Update (MMWU)

$$\Psi_{t+1} = \Lambda\Big(\eta \, \Sigma_{i=0}^{t} \, \mathcal{F}(\Psi_i) \Big)$$

☆ Jain & Watrous 2009 Proposal

QZSG Algorithm Design

Matrix Dual Averaging (MDA)

$$\Phi_{t+1} = \Phi_t + \mathcal{F}(\Psi_t)$$

$$\Psi_{t+1} = \mathrm{Mir}\Pi_{\mathrm{C}}^{h}(\eta\,\Phi_{t+1})$$

✔ 1 Gradient Call Per Iteration

h = entropy →

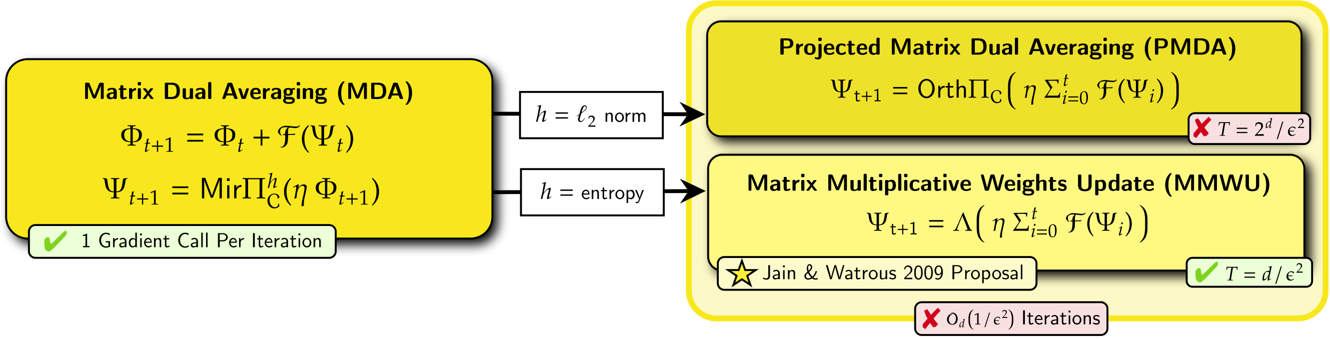
Matrix Multiplicative Weights Update (MMWU)

$$\Psi_{\mathrm{t+1}} = \Lambda\Big(\,\eta\,\Sigma_{i=0}^{t}\,\mathcal{F}(\Psi_i)\Big)$$

☆ Jain & Watrous 2009 Proposal

✔ $T = d\,/\,\epsilon^2$

QZSG Algorithm Design



QZSG Algorithm Design

Matrix Dual Averaging (MDA)

$$\Phi_{t+1} = \Phi_t + \mathcal{F}(\Psi_t)$$

$$\Psi_{t+1} = \mathrm{Mir}\Pi_{\mathrm{C}}^{h}(\eta\, \Phi_{t+1})$$

✔ 1 Gradient Call Per Iteration

$h = \ell_2$ norm →

Projected Matrix Dual Averaging (PMDA)

$$\Psi_{t+1} = \mathrm{Orth}\Pi_{\mathrm{C}}\Big(\, \eta\, \Sigma_{i=0}^{t}\, \mathcal{F}(\Psi_i)\,\Big)$$

✘ $T = 2^d / \epsilon^2$

$h =$ entropy →

Matrix Multiplicative Weights Update (MMWU)

$$\Psi_{t+1} = \Lambda\Big(\, \eta\, \Sigma_{i=0}^{t}\, \mathcal{F}(\Psi_i)\,\Big)$$

☆ Jain & Watrous 2009 Proposal

✔ $T = d / \epsilon^2$

✘ $O_d(1/\epsilon^2)$ Iterations

Mirror ⟶ Proximal Maps

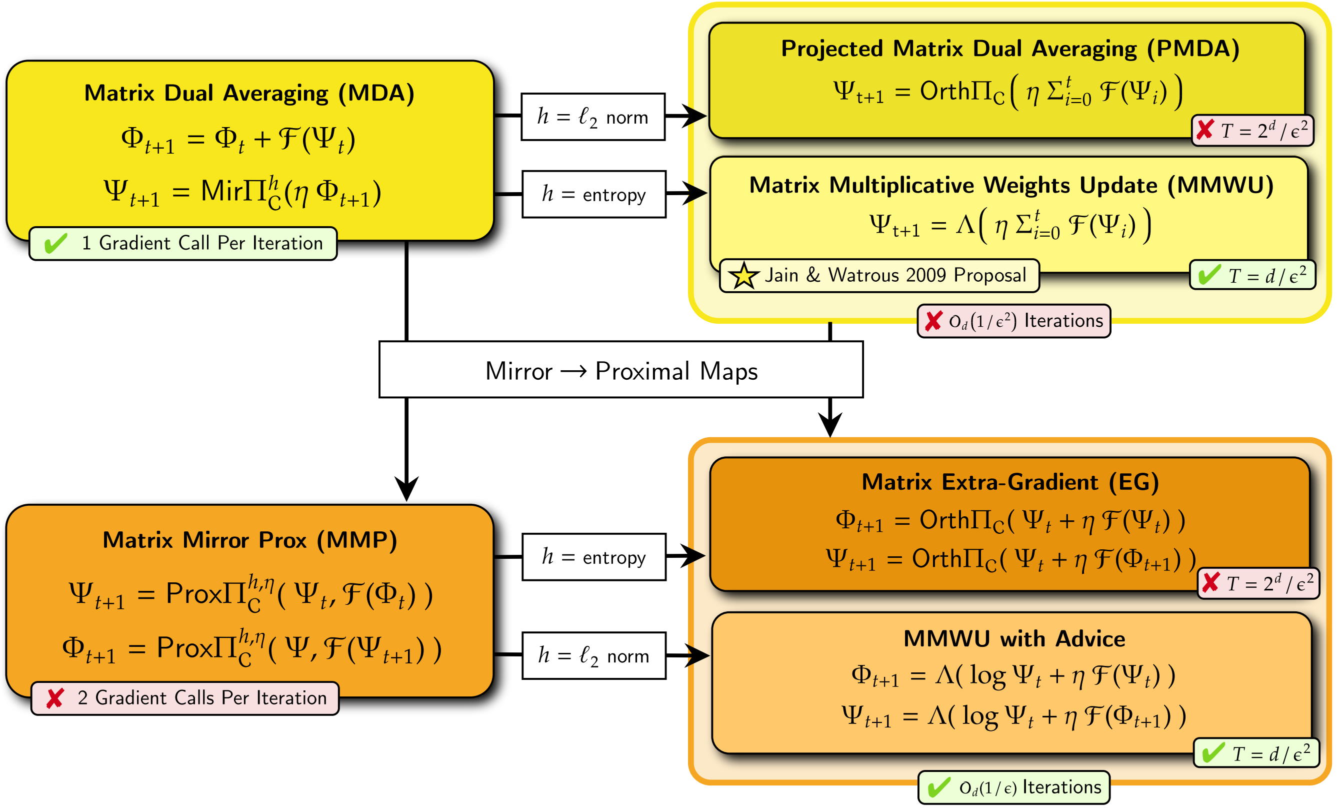
↓

Matrix Mirror Prox (MMP)

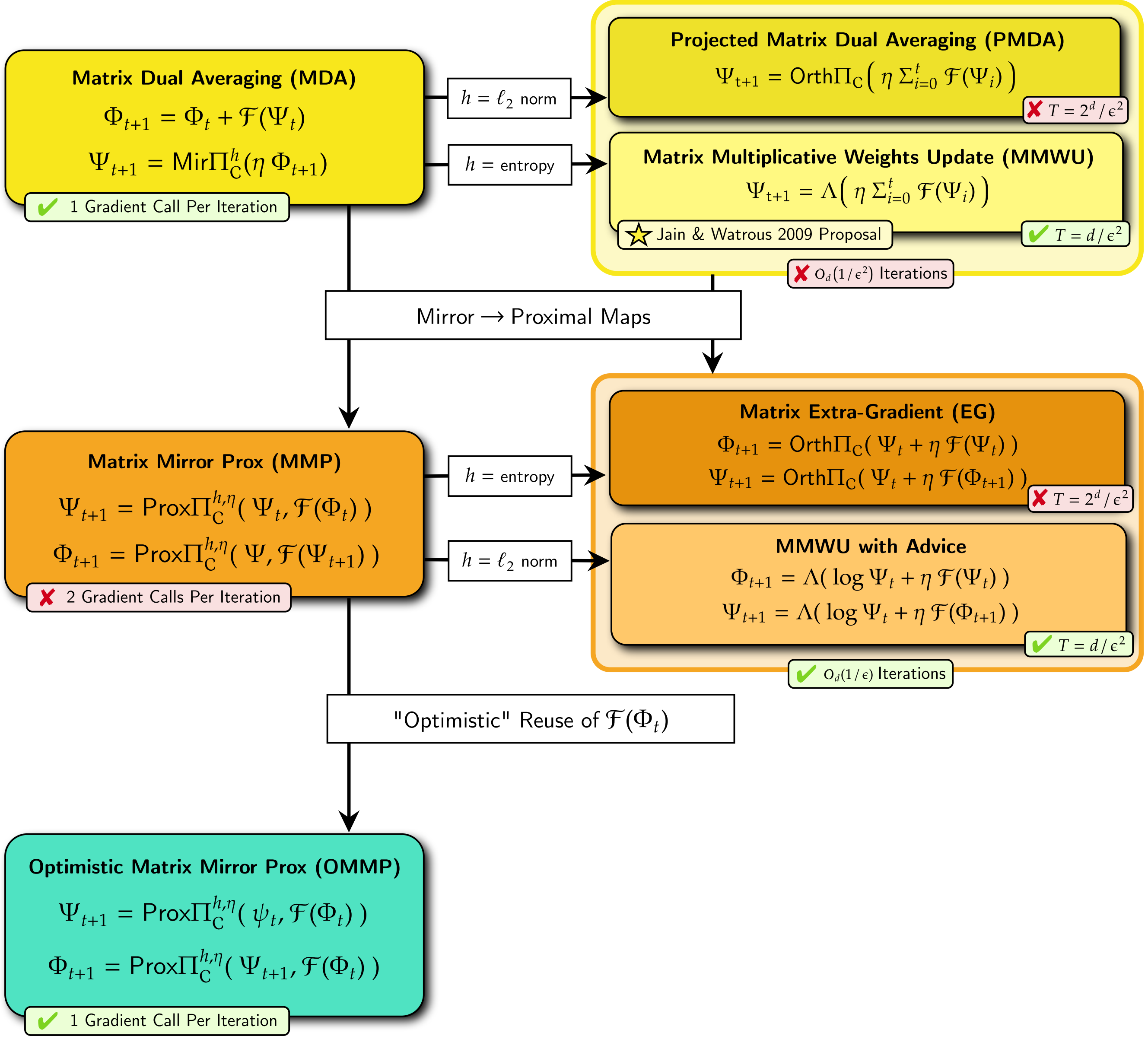
$$\Psi_{t+1} = \mathrm{Prox}\Pi_{\mathrm{C}}^{h,\eta}(\, \Psi_t, \mathcal{F}(\Phi_t)\,)$$

$$\Phi_{t+1} = \mathrm{Prox}\Pi_{\mathrm{C}}^{h,\eta}(\, \Psi, \mathcal{F}(\Psi_{t+1})\,)$$

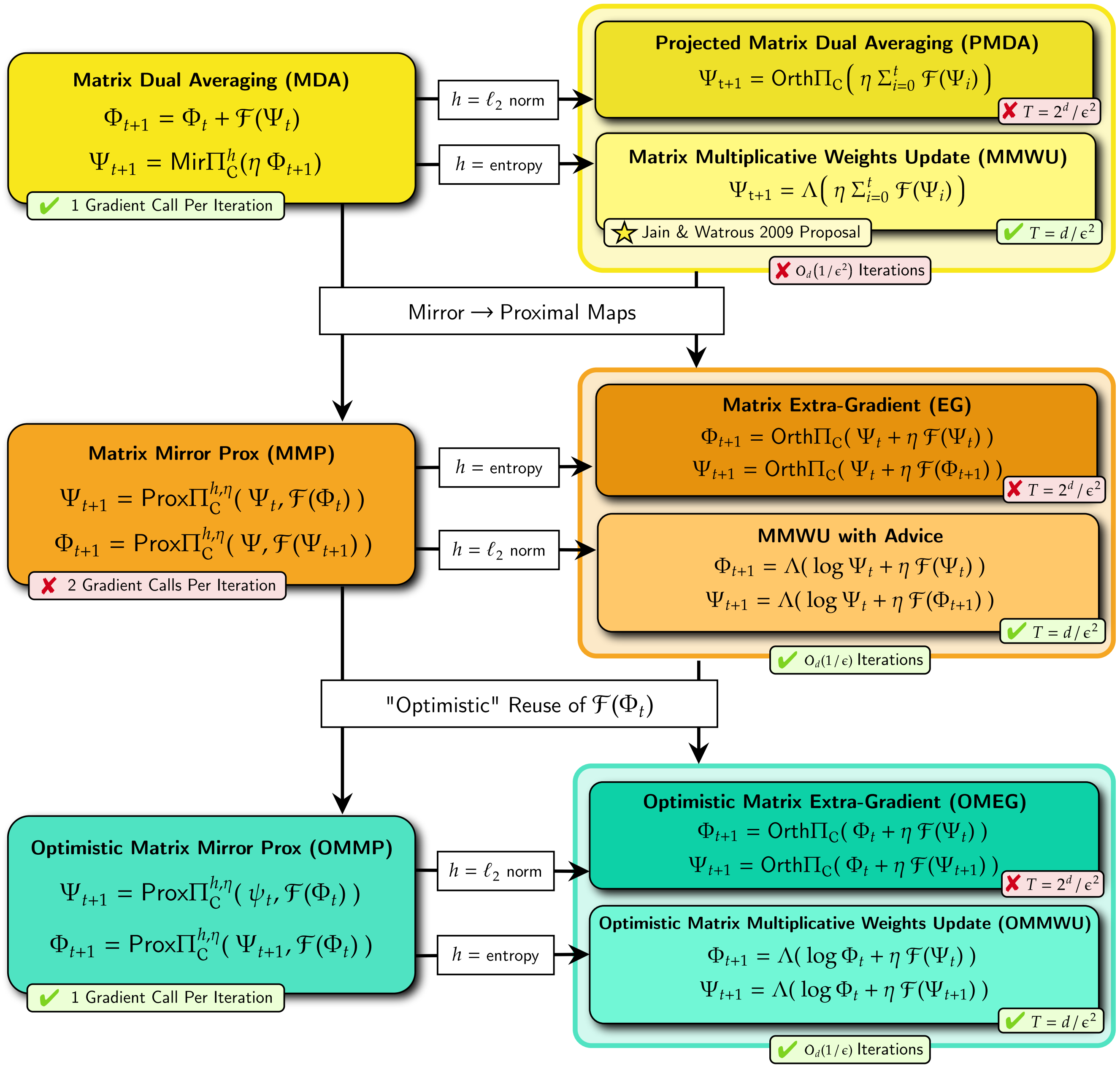
QZSG Algorithm Design



QZSG Algorithm Design



QZSG Algorithm Design



A Quadratic Speedup in Finding Nash Equilibria of Quantum Zero-Sum Games

[arXiv:2311.10859]

Thank you! Questions?

References

[BW22] John Bostanci and John Watrous. "Quantum game theory and the complexity of approximating quantum Nash equilibria". Quantum 6, 882 (2022).

[DK18] Pierre-Luc Dallaire-Demers and Nathan Killoran. "Quantum generative adversarial networks". Phys. Rev. A 98, 012324 (2018).

[EN20] Alina Ene and Huy Lê Nguyên. "Adaptive and Universal Algorithms for Variational Inequalities with Optimal Convergence". Proceedings of the AAAI Conference on Artificial Intelligence 36, 6559–6567 (2022).

[JW09] Rahul Jain and John Watrous. "Parallel Approximation of Non-interactive Zero-sum Quantum Games". In 2009 24th Annual IEEE Conference on Computational Complexity. Pages 243–253. (2009).

[M99] David A. Meyer. "Quantum Strategies". Phys. Rev. Lett. 82, 1052–1055 (1999).