# Introduction to game theory

Christos Dimitrakakis

Chalmers

May 16, 2014

# Deciding whether to take the bike to work

## Example 1 (Rain)

$\Omega = \{\mathrm{rain}, \mathrm{sun}\}$, $\mathcal{D} = \{\mathrm{bike}, \mathrm{tram}\}$
What we play depends on our own utility function, and the probability $P$ of different outcomes.

| $U(\omega, d)$ | $d_1$ | $d_2$ |
|---|---|---|
| $\omega_1$ | 0 | -10 |
| $\omega_2$ | -1 | 1 |

| | $d_1$ | $d_2$ |
|---|---|---|
| $U(P, d)$ | -0.8 | -1.2 |

Table : Utility and expected utility for 20% probability of rain.

# Deciding whether to take the bike to work

## Example 1 (Rain)

$\Omega = \{\text{rain}, \text{sun}\}$, $\mathcal{D} = \{\text{bike}, \text{tram}\}$
What we play depends on our own utility function, and the probability $P$ of different outcomes.

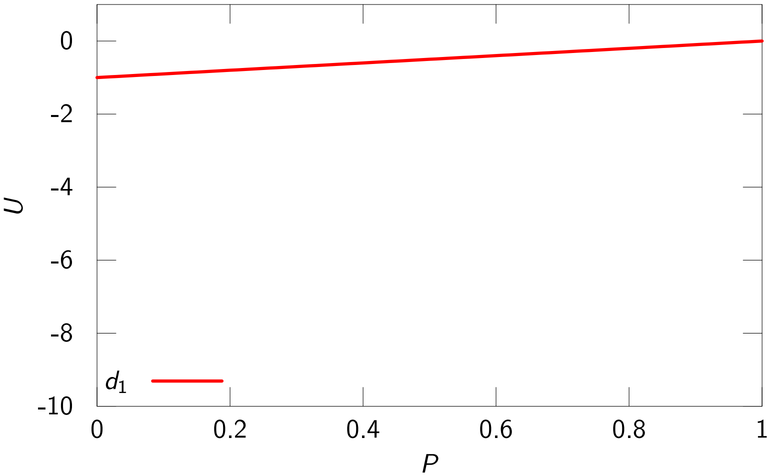
| $U(\omega, d)$ | $d_1$ | $d_2$ |
|---|---|---|
| $\omega_1$ | 0 | -10 |
| $\omega_2$ | -1 | 1 |

| | $d_1$ | $d_2$ |
|---|---|---|
| $U(P, d)$ | -0.8 | -1.2 |

Table : Utility and expected utility for 20% probability of rain.

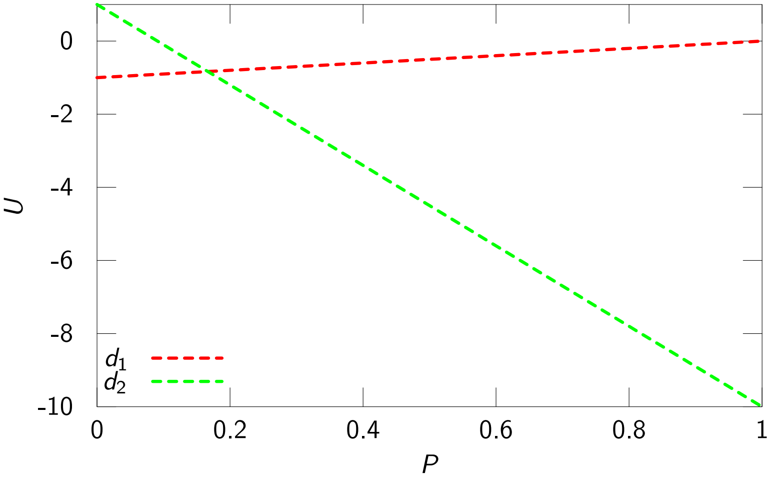
What if we $P$ is different, i.e. our belief is incorrect?

# Convexity of the Bayes utility



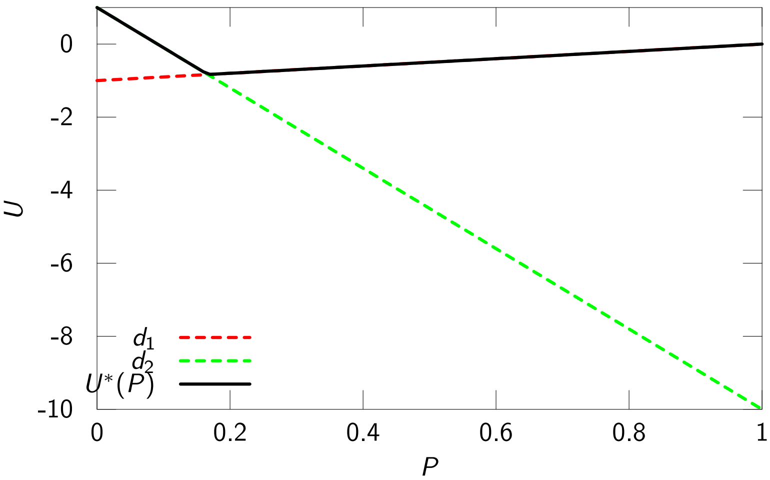

The expected utility of $d_1$

# Convexity of the Bayes utility




The expected utility of $d_1$ and $d_2$

# Convexity of the Bayes utility




The expected utility of $d_1$, $d_2$ and $\max_d U(P, d)$ – the optimal choice given $P$.

Consider two probability measures $P, Q$ on $(\Omega, \mathfrak{F}_\Omega)$.

$$Z_\alpha \triangleq \alpha P + (1-\alpha) Q \qquad \alpha \in [0,1]. \tag{1.1}$$

Consider two probability measures $P, Q$ on $(\Omega, \mathfrak{F}_\Omega)$.

$$Z_\alpha \triangleq \alpha P + (1-\alpha) Q \qquad \alpha \in [0,1]. \tag{1.1}$$

**Remark 1 (Linearity of the expected utility)**

*For any fixed $d \in \mathcal{D}$:*

$$U(Z_\alpha, d) = \alpha U(P, d) + (1-\alpha) U(Q, d). \tag{1.2}$$

Consider two probability measures $P, Q$ on $(\Omega, \mathfrak{F}_\Omega)$.

$$Z_\alpha \triangleq \alpha P + (1-\alpha)Q \qquad \alpha \in [0,1]. \tag{1.1}$$

**Remark 1 (Linearity of the expected utility)**

*For any fixed $d \in \mathcal{D}$:*

$$U(Z_\alpha, d) = \alpha U(P, d) + (1-\alpha) U(Q, d). \tag{1.2}$$

**Theorem 2 (The Bayes-optimal utility is convex)**

$$U^*[Z_\alpha] \leq \alpha U^*(P) + (1-\alpha) U^*(Q). \tag{1.3}$$

# Guarding against the worst-case

What if our belief $P$ is wrong? What would happen then?

## Minimax decisions

Solve:

$$\max_d \min_\omega U(\omega, d)$$

| $U(\omega, d)$ | $d_1$ | $d_2$ |
|---|---|---|
| $\omega_1$ | 0 | -10 |
| $\omega_2$ | -1 | 1 |
| $\min_\omega U(\omega, d)$ | | |

Table : The worst-case utility

# Guarding against the worst-case

What if our belief $P$ is wrong? What would happen then?

## Minimax decisions

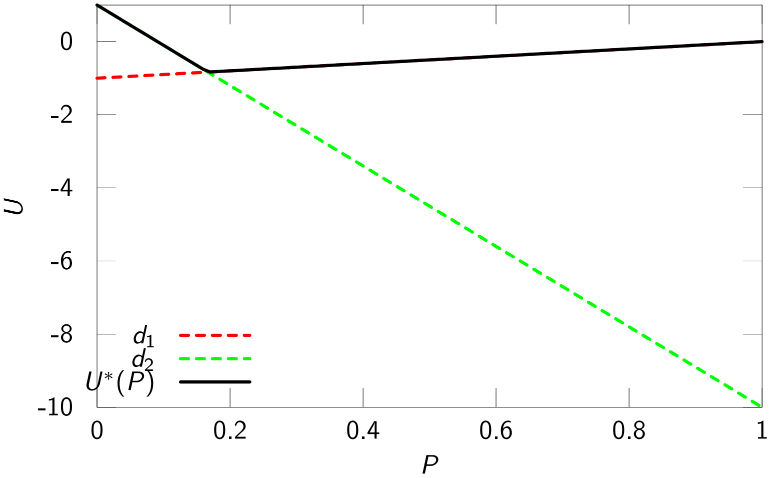
Solve:

$$\max_d \min_\omega U(\omega, d)$$

| $U(\omega, d)$ | $d_1$ | $d_2$ |
|---|---|---|
| $\omega_1$ | 0 | -10 |
| $\omega_2$ | -1 | 1 |
| $\min_\omega U(\omega, d)$ | -1 | -10 |

Table : The worst-case utility

# Convexity of the Bayes utility




The expected utility of $d_1$, $d_2$ and $\max_d U(P, d)$ – the optimal choice given $P$.

# Mixed decisions

Randomised decisions: Select decisions with probability $Q(d)$

# Mixed decisions

Randomised decisions: Select decisions with probability $Q(d)$

## Expected utility of a randomised decision

$$\begin{aligned} U(P, Q) &= \sum_{d \in \mathcal{D}} U(P, d) Q(d) \\ &= \sum_{d \in \mathcal{D}} \sum_{\omega \in \Omega} P(\omega) U(\omega, d) Q(d). \end{aligned}$$

# Mixed decisions

Randomised decisions: Select decisions with probability $Q(d)$

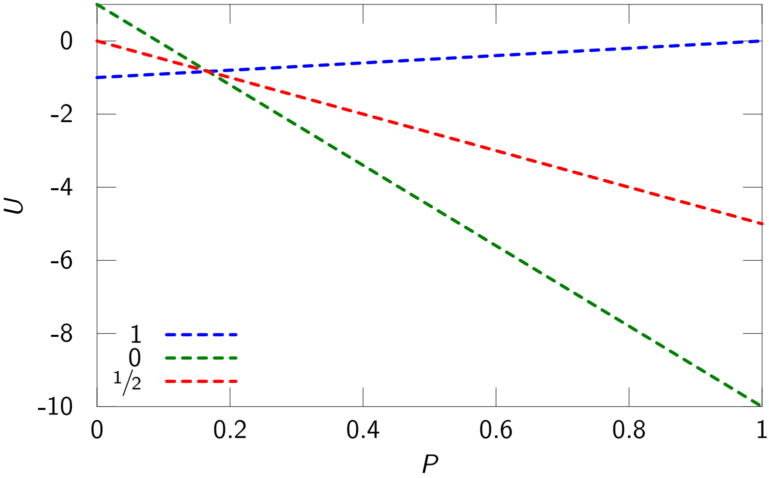
## Expected utility of a randomised decision

$$
\begin{aligned}
U(P, Q) &= \sum_{d \in \mathcal{D}} U(P, d) Q(d) \\
&= \sum_{d \in \mathcal{D}} \sum_{\omega \in \Omega} P(\omega) U(\omega, d) Q(d).
\end{aligned}
$$

## Maximin randomised decision

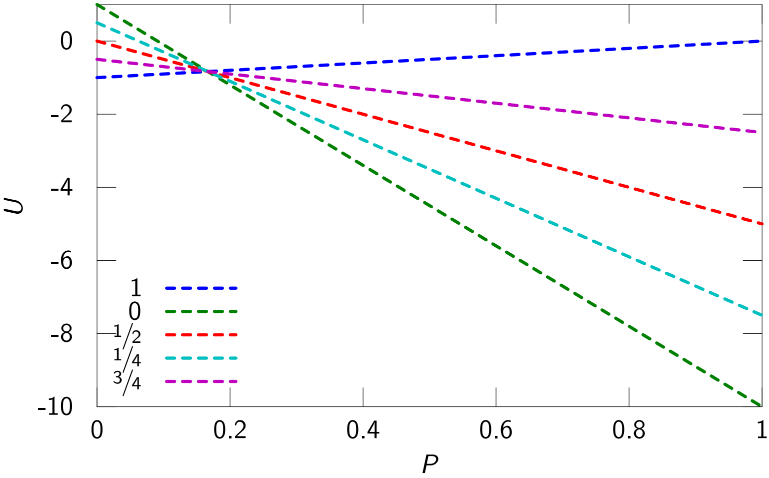
$$
\max_{Q \in \Delta(\mathcal{D})} \min_{P \in \Delta(\Omega)} U(P, Q)
$$

# Convexity of the Bayes utility



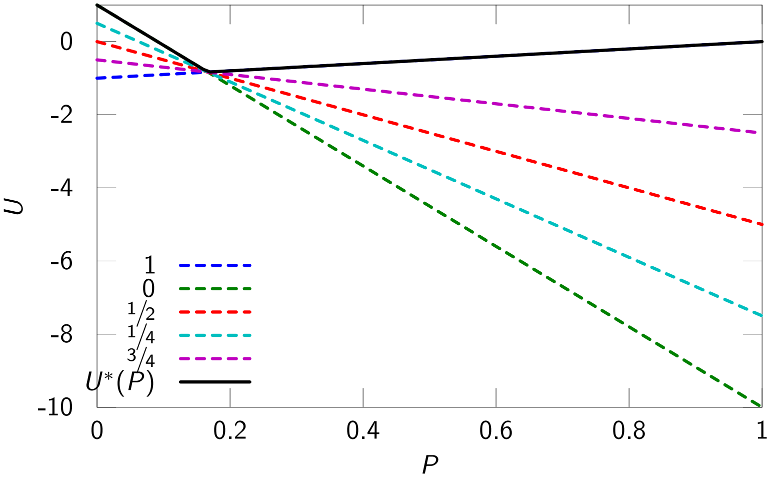

The expected utility of $d_1$, $d_2$ and mixed decision taking $d_1$ with probability $1/2$.

# Convexity of the Bayes utility




The expected utility of 5 different distributions $Q$ over $\mathcal{D}$.

# Convexity of the Bayes utility

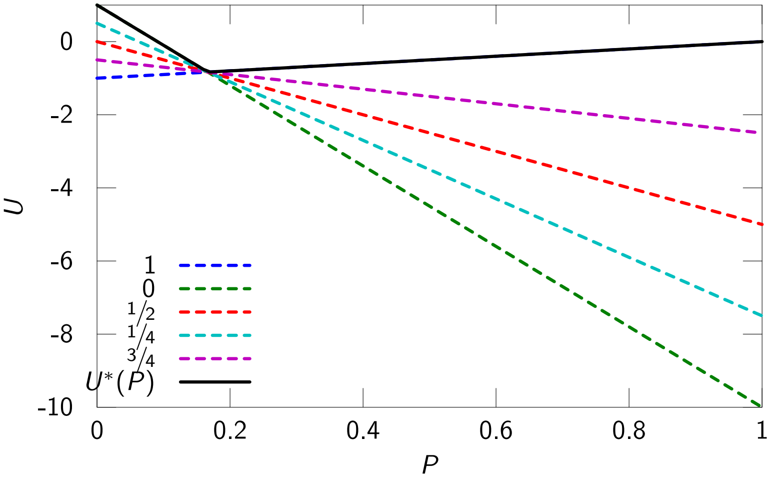



The expected utility of 5 different distributions $Q$ over $\mathcal{D}$ and the Bayes-optimal utility given $P$.

- If we know $P$, then we can play a single optimal decision $d^*(P)$.

- If we know $P$, then we can play a single optimal decision $d^*(P)$.
- There is a unique worst choice $P^*$ for the opponent. Can you guess what it is?

# Convexity of the Bayes utility

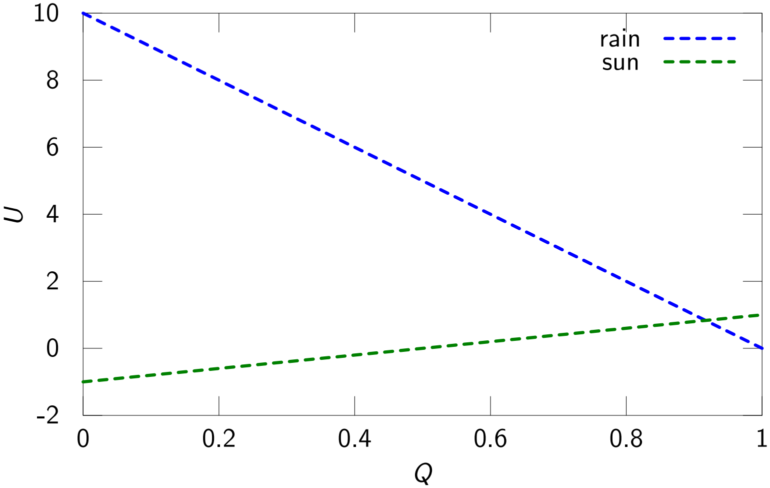



The expected utility of 5 different distributions $Q$ over $\mathcal{D}$ and the Bayes-optimal utility given $P$.

- If we know $P$, then we can play a single optimal decision $d^*(P)$.
- There is a unique worst choice $P^*$ for the opponent. Can you guess what it is?
- Under $P^*$, any of our decisions has the same expected utility.

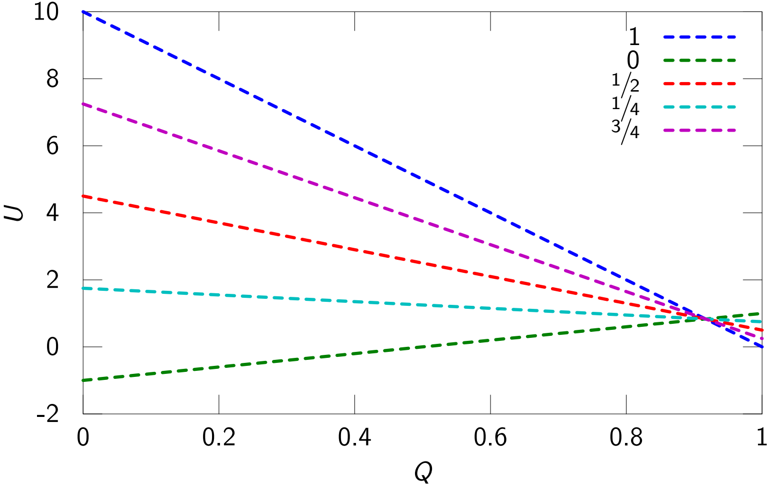
- If we know $P$, then we can play a single optimal decision $d^*(P)$.
- There is a unique worst choice $P^*$ for the opponent. Can you guess what it is?
- Under $P^*$, any of our decisions has the same expected utility.
- Is there a way to select $Q^*$ that is robust against Thor?

# The game from the point of view of Thor: $U_{\mathrm{Thor}} = -U$



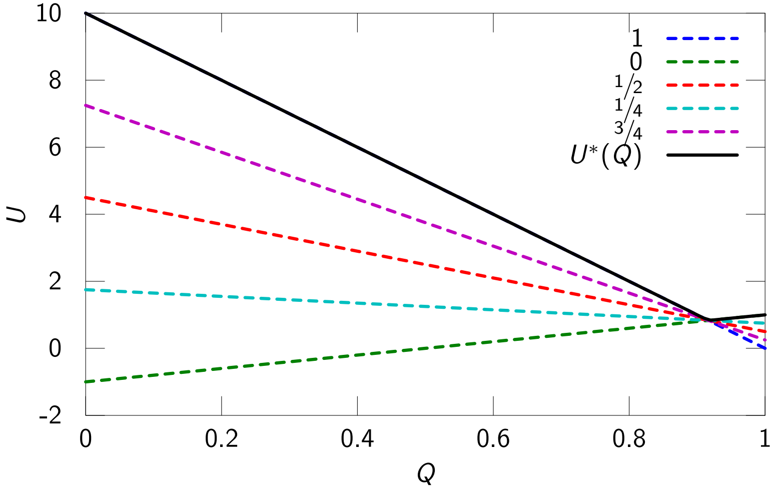

The expected utility of $\omega_1$, $\omega_2$ for different strategies.

# The game from the point of view of Thor: $U_{\text{Thor}} = -U$




The expected utility of various rain probabilities for different mixed strategies.

# The game from the point of view of Thor: $U_{\text{Thor}} = -U$




Thor's optimal decision for each mixed strategy we choose.

## Zero-sum one-shot game

A zero-sum game is a tuple $\langle \Omega, \mathcal{D}, U, \mathcal{P}, \mathcal{Q} \rangle$ where

- $\Omega$ are the pure strategies of the first player.

Player $Q$ wants to maximise $U$, and $P$ to minimise it (or vice-versa)

# Zero-sum one-shot game

A zero-sum game is a tuple $\langle \Omega, \mathcal{D}, U, \mathcal{P}, \mathcal{Q} \rangle$ where

- $\Omega$ are the pure strategies of the first player.
- $\mathcal{D}$ are the pure strategies of the second player.

Player $Q$ wants to maximise $U$, and $P$ to minimise it (or vice-versa)

# Zero-sum one-shot game

A zero-sum game is a tuple $\langle \Omega, \mathcal{D}, U, \mathcal{P}, \mathcal{Q} \rangle$ where

- $\Omega$ are the pure strategies of the first player.
- $\mathcal{D}$ are the pure strategies of the second player.
- $\mathcal{P}$ is the mixed strategy set of the first player.

Player $Q$ wants to maximise $U$, and $P$ to minimise it (or vice-versa)

# Zero-sum one-shot game

A zero-sum game is a tuple $\langle \Omega, \mathcal{D}, U, \mathcal{P}, \mathcal{Q} \rangle$ where

- $\Omega$ are the pure strategies of the first player.
- $\mathcal{D}$ are the pure strategies of the second player.
- $\mathcal{P}$ is the mixed strategy set of the first player.
- $\mathcal{Q}$ is the mixed strategy set of the second player.

Player $Q$ wants to maximise $U$, and $P$ to minimise it (or vice-versa)

# Zero-sum one-shot game

A zero-sum game is a tuple $\langle \Omega, \mathcal{D}, U, \mathcal{P}, \mathcal{Q} \rangle$ where

- $\Omega$ are the pure strategies of the first player.
- $\mathcal{D}$ are the pure strategies of the second player.
- $\mathcal{P}$ is the mixed strategy set of the first player.
- $\mathcal{Q}$ is the mixed strategy set of the second player.
- $U : \Omega \times \mathcal{D} \to \mathbb{R}$ is a utility function.

Player $Q$ wants to maximise $U$, and $P$ to minimise it (or vice-versa)

# Zero-sum one-shot game

A zero-sum game is a tuple $\langle \Omega, \mathcal{D}, U, \mathcal{P}, \mathcal{Q} \rangle$ where

- $\Omega$ are the pure strategies of the first player.
- $\mathcal{D}$ are the pure strategies of the second player.
- $\mathcal{P}$ is the mixed strategy set of the first player.
- $\mathcal{Q}$ is the mixed strategy set of the second player.
- $U : \Omega \times \mathcal{D} \to \mathbb{R}$ is a utility function.

Player $Q$ wants to maximise $U$, and $P$ to minimise it (or vice-versa)

## Game structure

1. The players choose $P, Q$ and don't reveal it.

# Zero-sum one-shot game

A zero-sum game is a tuple $\langle \Omega, \mathcal{D}, U, \mathcal{P}, \mathcal{Q} \rangle$ where

- $\Omega$ are the pure strategies of the first player.
- $\mathcal{D}$ are the pure strategies of the second player.
- $\mathcal{P}$ is the mixed strategy set of the first player.
- $\mathcal{Q}$ is the mixed strategy set of the second player.
- $U : \Omega \times \mathcal{D} \to \mathbb{R}$ is a utility function.

Player $Q$ wants to maximise $U$, and $P$ to minimise it (or vice-versa)

## Game structure

1. The players choose $P, Q$ and don't reveal it.
2. The players randomly select $\omega, d$ from $P, Q$.

# Zero-sum one-shot game

A zero-sum game is a tuple $\langle \Omega, \mathcal{D}, U, \mathcal{P}, \mathcal{Q} \rangle$ where

- $\Omega$ are the pure strategies of the first player.
- $\mathcal{D}$ are the pure strategies of the second player.
- $\mathcal{P}$ is the mixed strategy set of the first player.
- $\mathcal{Q}$ is the mixed strategy set of the second player.
- $U : \Omega \times \mathcal{D} \to \mathbb{R}$ is a utility function.

Player $Q$ wants to maximise $U$, and $P$ to minimise it (or vice-versa)

## Game structure

1. The players choose $P, Q$ and don't reveal it.
2. The players randomly select $\omega, d$ from $P, Q$.
3. $\omega, d$ is revealed and the players get $U(\omega, d)$ and $-U(\omega, d)$.

# Properties of zero-sum games

$$U_* \triangleq \max_{Q \in \mathcal{Q}} \min_{\omega \in \Omega} U(\omega, Q) \leq \min_{P \in \mathcal{P}} \max_{d \in \mathcal{D}} U(P, d) \triangleq U^*.$$

Recall that we don't need to randomise if we know $P$ !

**Theorem 3**

*If $\mathcal{P}, \mathcal{Q}$ include all probability distributions over pure strategies then*

$$U_* = U^*$$

*and is the value of the game. In fact, if $P^*$ and $Q^*$ are the corresponding optimal strategies, then:*

$$U_* = U(P^*, Q^*) = U^*.$$

# Linear programming

- We have seen that we can always find the best mixed strategy $P^*$ and $Q^*$ for either player.

# Linear programming

- We have seen that we can always find the best mixed strategy $P^*$ and $Q^*$ for either player.
- This entails solving a set of linear inequalities.

# Linear programming

- We have seen that we can always find the best mixed strategy $P^*$ and $Q^*$ for either player.
- This entails solving a set of linear inequalities.
- Consequently, the problem can be solved with linear programming.

# Linear programming

- We have seen that we can always find the best mixed strategy $P^*$ and $Q^*$ for either player.
- This entails solving a set of linear inequalities.
- Consequently, the problem can be solved with linear programming.
- This means the complexity of zero-sum games is polynomial.

# Linear programming

- We have seen that we can always find the best mixed strategy $P^*$ and $Q^*$ for either player.
- This entails solving a set of linear inequalities.
- Consequently, the problem can be solved with linear programming.
- This means the complexity of zero-sum games is polynomial.
- However, there are conceptually simpler ways, which can give incremental solutions, with polynomial complexity.

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.
- A learning rate $\eta$.

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.
- A learning rate $\eta$.
- A set of weights, $\boldsymbol{w}_t = (w_{t,i})_{i=1}^n$.

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.
- A learning rate $\eta$.
- A set of weights, $\boldsymbol{w}_t = (w_{t,i})_{i=1}^n$.
- The sum of the weights $W_t = \sum_i w_{t,i}$.

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0,1]$.
- A learning rate $\eta$.
- A set of weights, $\boldsymbol{w}_t = (w_{t,i})_{i=1}^n$.
- The sum of the weights $W_t = \sum_i w_{t,i}$.

For each round $t$:

1. $Q$ normalises the weights to get a distribution $\boldsymbol{q}_t = \boldsymbol{w}_t / W_t$.

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.
- A learning rate $\eta$.
- A set of weights, $\boldsymbol{w}_t = (w_{t,i})_{i=1}^{n}$.
- The sum of the weights $W_t = \sum_i w_{t,i}$.

For each round $t$:

1. $Q$ normalises the weights to get a distribution $\boldsymbol{q}_t = \boldsymbol{w}_t / W_t$.
2. $Q$ plays $d_t \sim \boldsymbol{q}_t$ from the distribution.

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.
- A learning rate $\eta$.
- A set of weights, $\boldsymbol{w}_t = (w_{t,i})_{i=1}^n$.
- The sum of the weights $W_t = \sum_i w_{t,i}$.

For each round $t$:

1. $Q$ normalises the weights to get a distribution $\boldsymbol{q}_t = \boldsymbol{w}_t / W_t$.
2. $Q$ plays $d_t \sim \boldsymbol{q}_t$ from the distribution.
3. $P$ plays the best response: $\omega_t \in \arg\min_\omega U(\omega, \boldsymbol{q}_t)$

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.
- A learning rate $\eta$.
- A set of weights, $\boldsymbol{w}_t = (w_{t,i})_{i=1}^n$.
- The sum of the weights $W_t = \sum_i w_{t,i}$.

For each round $t$:

1. $Q$ normalises the weights to get a distribution $\boldsymbol{q}_t = \boldsymbol{w}_t / W_t$.
2. $Q$ plays $d_t \sim \boldsymbol{q}_t$ from the distribution.
3. $P$ plays the best response: $\omega_t \in \arg\min_\omega U(\omega, \boldsymbol{q}_t)$
4. $Q$ gets a reward $r_t = U(\omega_t, d_t)$ and calculates $r_{t,d} = U(\omega_t, d)$ for all pure strategies $d$.

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.
- A learning rate $\eta$.
- A set of weights, $\boldsymbol{w}_t = (w_{t,i})_{i=1}^{n}$.
- The sum of the weights $W_t = \sum_i w_{t,i}$.

For each round $t$:

1. $Q$ normalises the weights to get a distribution $\boldsymbol{q}_t = \boldsymbol{w}_t / W_t$.
2. $Q$ plays $d_t \sim \boldsymbol{q}_t$ from the distribution.
3. $P$ plays the best response: $\omega_t \in \arg\min_\omega U(\omega, \boldsymbol{q}_t)$
4. $Q$ gets a reward $r_t = U(\omega_t, d_t)$ and calculates $r_{t,d} = U(\omega_t, d)$ for all pure strategies $d$.
5. $Q$ calculates the next set of weights for all $d$

$$w_{t+1,d} = w_{t,d}(1 + \eta r_{t,d}).$$

# The weighted majority algorithm

- $n$ pure strategies for player $Q$, utility $U \in [0, 1]$.
- A learning rate $\eta$.
- A set of weights, $\boldsymbol{w}_t = (w_{t,i})_{i=1}^n$.
- The sum of the weights $W_t = \sum_i w_{t,i}$.

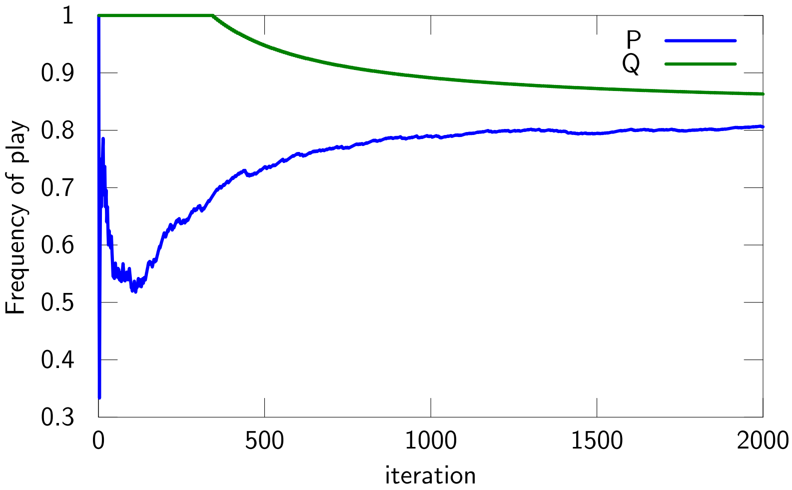
For each round $t$:

1. $Q$ normalises the weights to get a distribution $\boldsymbol{q}_t = \boldsymbol{w}_t / W_t$.
2. $Q$ plays $d_t \sim \boldsymbol{q}_t$ from the distribution.
3. $P$ plays the best response: $\omega_t \in \arg\min_\omega U(\omega, \boldsymbol{q}_t)$
4. $Q$ gets a reward $r_t = U(\omega_t, d_t)$ and calculates $r_{t,d} = U(\omega_t, d)$ for all pure strategies $d$.
5. $Q$ calculates the next set of weights for all $d$

$$w_{t+1,d} = w_{t,d}(1 + \eta r_{t,d}).$$

6. A near-optimal strategy for $Q$ is the empirical frequency of $d_t$!

# Convergence of run play frequencies



# General-sum two-player games

## Definition 4

A zero-sum game is a tuple $\langle \mathcal{D}, U, \mathcal{P} \rangle$ where

- $\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2$ are the pure strategies of the two players.

## Properties of general-sum games

# General-sum two-player games

## Definition 4

A zero-sum game is a tuple $\langle \mathcal{D}, U, \mathcal{P} \rangle$ where

- $\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2$ are the pure strategies of the two players.
- $\mathcal{P} = \mathcal{P}_1 \times \mathcal{P}_2$ are the mixed strategy set of the two players.

## Properties of general-sum games

# General-sum two-player games

## Definition 4

A zero-sum game is a tuple $\langle \mathcal{D}, U, \mathcal{P} \rangle$ where

- $\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2$ are the pure strategies of the two players.
- $\mathcal{P} = \mathcal{P}_1 \times \mathcal{P}_2$ are the mixed strategy set of the two players.
- $U : \mathcal{D} \to \mathbb{R}^2$ is a utility function with components $U_i$. The $i$-th player wants to maximise $U_i$.

## Properties of general-sum games

# General-sum two-player games

## Definition 4

A zero-sum game is a tuple $\langle \mathcal{D}, U, \mathcal{P} \rangle$ where

- $\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2$ are the pure strategies of the two players.
- $\mathcal{P} = \mathcal{P}_1 \times \mathcal{P}_2$ are the mixed strategy set of the two players.
- $U : \mathcal{D} \to \mathbb{R}^2$ is a utility function with components $U_i$. The $i$-th player wants to maximise $U_i$.

## Properties of general-sum games

- If $\mathcal{P}$ is the set of all mixed strategies, then a solution $\boldsymbol{p}^* = (p_1^*, p_2^*)$ exists such that

$$U_2(p_1^*, p_2^*) \geq U_2(p_1^*, p_2), \quad U_1(p_1^*, p_2^*) \geq U_1(p_1, p_2^*), \quad (2.1)$$

for any $p_1, p_2$. This is called a Nash equilibrium.

# General-sum two-player games

## Definition 4

A zero-sum game is a tuple $\langle \mathcal{D}, U, \mathcal{P} \rangle$ where

- $\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2$ are the pure strategies of the two players.
- $\mathcal{P} = \mathcal{P}_1 \times \mathcal{P}_2$ are the mixed strategy set of the two players.
- $U : \mathcal{D} \to \mathbb{R}^2$ is a utility function with components $U_i$. The $i$-th player wants to maximise $U_i$.

## Properties of general-sum games

- If $\mathcal{P}$ is the set of all mixed strategies, then a solution $\boldsymbol{p}^* = (p_1^*, p_2^*)$ exists such that

$$U_2(p_1^*, p_2^*) \geq U_2(p_1^*, p_2), \quad U_1(p_1^*, p_2^*) \geq U_1(p_1, p_2^*), \quad (2.1)$$

  for any $p_1, p_2$. This is called a Nash equilibrium.
- Finding a Nash equilibrium is in NP (in fact PPAD).

# Nash as a solution concept

### The prisoner's dilemma

| $U_1$, $U_2$ | Co-operate | Defect |
|---|---|---|
| Co-operate | 1, 1 | -1, 2 |
| Defect | 2, -1 | 0,0 |

In this case, both player defecting is a dominant strategy, even though both players co-operating would be better for both!

# Sequential games

- We already hinted at the fact that players may take turns.

### Definition 5

The value of a state $s_t$ in a zero-sum Markov game for a policy $p$, against a minimising player with policy $q$ is

$$V^*(s) = \max_p \min_q \mathbb{E}_p^q(U \mid s_t = s)$$

# Sequential games

- We already hinted at the fact that players may take turns.
- In fact, players may make a series of moves in a game (e.g. in chess)

### Definition 5

The value of a state $s_t$ in a zero-sum Markov game for a policy $p$, against a minimising player with policy $q$ is

$$V^*(s) = \max_p \min_q \mathbb{E}_p^q(U \mid s_t = s)$$

# Sequential games

- We already hinted at the fact that players may take turns.
- In fact, players may make a series of moves in a game (e.g. in chess)
- Then strategies are defined as functions from an observation history to a next move.

## Definition 5

The value of a state $s_t$ in a zero-sum Markov game for a policy $p$, against a minimising player with policy $q$ is

$$V^*(s) = \max_p \min_q \mathbb{E}_p^q(U \mid s_t = s)$$

# Sequential games

- We already hinted at the fact that players may take turns.
- In fact, players may make a series of moves in a game (e.g. in chess)
- Then strategies are defined as functions from an observation history to a next move.
- A specifically interesting case is that of Markov games, where the strategy only depends on the game state.

## Definition 5

The value of a state $s_t$ in a zero-sum Markov game for a policy $p$, against a minimising player with policy $q$ is

$$V^*(s) = \max_p \min_q \mathbb{E}_p^q(U \mid s_t = s)$$

# Sequential games

- We already hinted at the fact that players may take turns.
- In fact, players may make a series of moves in a game (e.g. in chess)
- Then strategies are defined as functions from an observation history to a next move.
- A specifically interesting case is that of Markov games, where the strategy only depends on the game state.
- These can be solved with backwards induction.

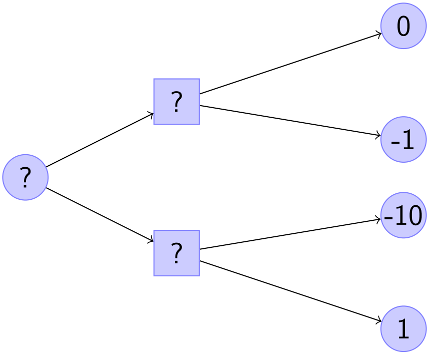
### Definition 5

The value of a state $s_t$ in a zero-sum Markov game for a policy $p$, against a minimising player with policy $q$ is

$$V^*(s) = \max_p \min_q \mathbb{E}_p^q(U \mid s_t = s)$$

# Backwards induction for Markov games

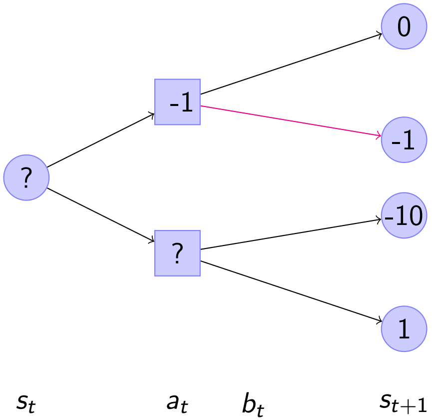
$$v_t(s_t) = \max_{a_t \in \mathcal{D}_1} \left\{ \min_{b_t \in \mathcal{D}_2} v_{t+1}(s_{t+1}) \right\}$$




$s_t$ $a_t$ $b_t$ $s_{t+1}$

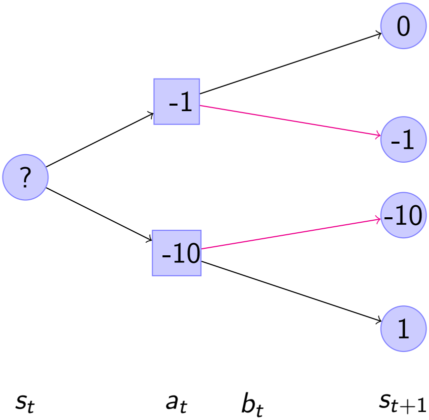
# Backwards induction for Markov games

$$v_t(s_t) = \max_{a_t \in \mathcal{D}_1} \left\{ \min_{b_t \in \mathcal{D}_2} v_{t+1}(s_{t+1}) \right\}$$



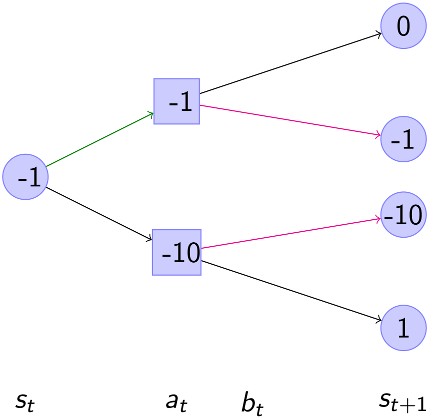
# Backwards induction for Markov games

$$v_t(s_t) = \max_{a_t \in \mathcal{D}_1} \left\{ \min_{b_t \in \mathcal{D}_2} v_{t+1}(s_{t+1}) \right\}$$



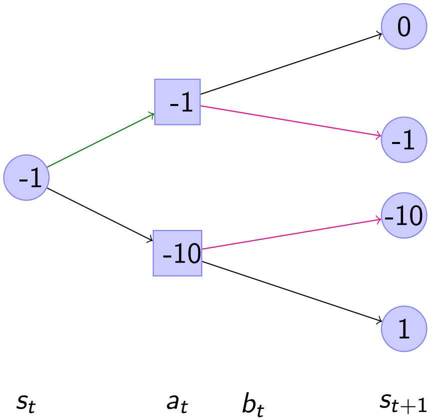
# Backwards induction for Markov games

$$v_t(s_t) = \max_{a_t \in \mathcal{D}_1} \left\{ \min_{b_t \in \mathcal{D}_2} v_{t+1}(s_{t+1}) \right\}$$



# Backwards induction for Markov games

$$v_t(s_t) = \max_{a_t \in \mathcal{D}_1} \left\{ \min_{b_t \in \mathcal{D}_2} v_{t+1}(s_{t+1}) \right\}$$




Extension to stochastic Markov games is easy!

# Properties of Markov games

- Generally the same as a Markov decision process:

# Properties of Markov games

- Generally the same as a Markov decision process:
- Easy to solve as long as the number of states is finite and the utility function is additive.

# Properties of Markov games

- Generally the same as a Markov decision process:
- Easy to solve as long as the number of states is finite and the utility function is additive.
- If moves are simultaneous at every round, then they can still be solved if they are zero-sum.

# Properties of Markov games

- Generally the same as a Markov decision process:
- Easy to solve as long as the number of states is finite and the utility function is additive.
- If moves are simultaneous at every round, then they can still be solved if they are zero-sum.
- Otherwise, they are PPAD by reduction to non-zero-sum games.

# Information states

What we know about a game, and the state of the game, comprises our information state. These pieces of knowledge may include:

- A prior distribution $P$ on $\omega$.
- The utility function of the game for all players.
- Any random variables defined on the space of $P$.
- The moves played by the players so far.
- The utility obtained by the players so far.

In general, the more information, the better we can do, and the simpler the algorithms we can use.

# Unknown utility games

- What if we do not know the utility of anybody else in the game?

# Unknown utility games

- What if we do not know the utility of anybody else in the game?
- One idea is to take a worst-case approach:

# Unknown utility games

- What if we do not know the utility of anybody else in the game?
- One idea is to take a worst-case approach:
- This results in the standard minimax framework and a zero-sum two-player game.

# Unknown utility games

- What if we do not know the utility of anybody else in the game?
- One idea is to take a worst-case approach:
- This results in the standard minimax framework and a zero-sum two-player game.
- But what if we have some idea about what they want?

# Unknown utility games

- What if we do not know the utility of anybody else in the game?
- One idea is to take a worst-case approach:
- This results in the standard minimax framework and a zero-sum two-player game.
- But what if we have some idea about what they want?
- We could use a subjective probability distribution to model our uncertainty. This is the topic of Bayesian games (not covered here).

# Multi-player games

## Definition 6

A general $n$-player game is a tuple $\langle \mathcal{D}, U, \mathcal{P} \rangle$ where

- $\mathcal{D} = \prod_{i=1}^{n} \mathcal{D}_i$ are the pure strategies of the $n$ players.

A game is co-operative if $U_i = U_j$ for all players. These games are slightly easier (exponential in the number of players).

▸ ROBOTS

# Multi-player games

## Definition 6

A general $n$-player game is a tuple $\langle \mathcal{D}, U, \mathcal{P} \rangle$ where

- $\mathcal{D} = \prod_{i=1}^{n} \mathcal{D}_i$ are the pure strategies of the $n$ players.
- $\mathcal{P} = \prod_{i=1}^{n} \mathcal{P}_i$ are the mixed strategy sets of the players.

A game is co-operative if $U_i = U_j$ for all players. These games are slightly easier (exponential in the number of players).

▸ ROBOTS

# Multi-player games

### Definition 6

A general $n$-player game is a tuple $\langle \mathcal{D}, U, \mathcal{P} \rangle$ where

- $\mathcal{D} = \prod_{i=1}^{n} \mathcal{D}_i$ are the pure strategies of the $n$ players.
- $\mathcal{P} = \prod_{i=1}^{n} \mathcal{P}_i$ are the mixed strategy sets of the players.
- $U : \mathcal{D} \to \mathbb{R}^n$ is a utility function. The $i$-th player wants to maximise $U_i$.

A game is co-operative if $U_i = U_j$ for all players. These games are slightly easier (exponential in the number of players).

▸ ROBOTS

# Categories of games

## Move structure

- One-shot; Repeated; Sequential.
- All moves observed; Only some moves known.

## Utility

- Zero-sum; Collaborative; Additive; Arbitray.
- Fully known; Only for the player; Only individual rewards;

## Stochasticity

- World: Deterministic, stochastic.
- Players: Deterministic, stochastic.