TWO-PLAYER ZERO-SUM GAMES

Tomáš Kroupa

AI Center
Department of Computer Science
Faculty of Electrical Engineering
Czech Technical University in Prague

MORRA

- Each player raises between 1 and 3 fingers and simultaneously makes a guess about how many fingers the opponent will raise
- There is no payoff unless *exactly one player predicts correctly*
- The correct guesser wins an amount from the other player, which is equal to the total number of fingers raised by both players

MORRA MATRIX

	1-1	1-2	1-3	2-1	2-2	2-3	3-1	3-2	3-3
1-1	0	2	2	-3	0	0	-4	0	0
1-2	-2	0	0	0	3	3	-4	0	0
1-3	-2	0	0	-3	0	0	0	4	4
2-1	3	0	3	0	-4	0	0	-5	0
2-2	0	-3	0	4	0	4	0	-5	0
2-3	0	-3	0	0	-4	0	5	0	5
3-1	4	4	0	0	0	-5	0	0	-6
3-2	0	0	-4	5	5	0	0	0	-6
3-3	0	0	-4	0	0	-5	6	6	0

MATRIX GAMES

1. $N = \{1, 2\}$
2. Finite strategy sets S_1 and S_2
3. Utility functions satisfy $u_1 + u_2 = 0$

Remarks

- Notation $u := u_1$
- We can view $u = -u_2$ as the loss function of player 2
- Terminology: player 1 is *maximizing* while player 2 is *minimizing*

MINIMAX THEOREM

SOLVING MATRIX GAMES

- A Nash equilibrium $(p_1^*, p_2^*) \in \boldsymbol{\Delta}$ exists for any matrix game:

$$U(p_1, p_2^*) \leq U(p_1^*, p_2^*) \leq U(p_1^*, p_2) \qquad \forall (p_1, p_2) \in \boldsymbol{\Delta}$$

- We derive this result from fundamental principles
- This will lead naturally to a linear programming (LP) problem

PURE EQUILIBRIA IN MATRIX GAMES

- A pure Nash equilibrium $(s_1^*, s_2^*) \in \mathbf{S}$ in a matrix game is defined by

$$u(s_1, s_2^*) \leq u(s_1^*, s_2^*) \leq u(s_1^*, s_2) \qquad \forall (s_1, s_2) \in \mathbf{S}$$

- The strategy profile (s_1^*, s_2^*) is also called a saddle point

	d	e	f
a	1	2	3
b	4	5	6
c	7	8	9

$$\max_{s_1 \in S_1} \min_{s_2 \in S_2} u(s_1, s_2) = 7 = \min_{s_2 \in S_2} \max_{s_1 \in S_1} u(s_1, s_2)$$

LOWER/UPPER VALUE

Lower bound on the utility

- Given p_1, player 2 computes

$$\min_{p_2 \in \Delta_2} U(p_1, p_2)$$

- Player 1 then computes

$$\underline{v} := \max_{p_1 \in \Delta_1} \min_{p_2 \in \Delta_2} U(p_1, p_2)$$

$$= \max_{p_1 \in \Delta_1} \min_{s_2 \in S_2} U(p_1, s_2)$$

Upper bound on the loss

- Given p_2, player 1 computes

$$\max_{p_1 \in \Delta_1} U(p_1, p_2)$$

- Player 2 then computes

$$\overline{v} := \min_{p_2 \in \Delta_2} \max_{p_1 \in \Delta_1} U(p_1, p_2)$$

$$= \min_{p_2 \in \Delta_2} \max_{s_1 \in S_1} U(s_1, p_2)$$

LP FORMULATION

Player 1

$$\max_{p_1 \in \Delta_1} \min_{s_2 \in S_2} U(p_1, s_2)$$

Maximize v_1

subject to

$U(p_1, s_2) \geq v_1 \qquad \forall s_2 \in S_2$

$p_1 \in \Delta_1$

$v_1 \in \mathbb{R}$

Player 2

$$\min_{p_2 \in \Delta_2} \max_{s_1 \in S_1} U(s_1, p_2)$$

Minimize v_2

subject to

$U(s_1, p_2) \leq v_2 \qquad \forall s_1 \in S_1$

$p_2 \in \Delta_2$

$v_2 \in \mathbb{R}$

Minimax theorem

The two LPs are dual and their optimal value is $v := \underline{v} = \overline{v}$.

VALUE OF THE GAME

- The optimal solutions (p_1^*, p_2^*) of the two dual LPs are called the maximin strategy and the minimax strategy, respectively
- The value of the game v represents a unique outcome of the game associated with the strategy profile (p_1^*, p_2^*),

$$\max_{p_1 \in \Delta_1} \min_{p_2 \in \Delta_2} U(p_1, p_2) = \underbrace{U(p_1^*, p_2^*)}_{v} = \min_{p_2 \in \Delta_2} \max_{p_1 \in \Delta_1} U(p_1, p_2)$$

THE SOLUTION OF MORRA

	1-1	1-2	1-3	2-1	2-2	2-3	3-1	3-2	3-3
1-1	0	2	2	-3	0	0	-4	0	0
1-2	-2	0	0	0	3	3	-4	0	0
1-3	-2	0	0	-3	0	0	0	4	4
2-1	3	0	3	0	-4	0	0	-5	0
2-2	0	-3	0	4	0	4	0	-5	0
2-3	0	-3	0	0	-4	0	5	0	5
3-1	4	4	0	0	0	-5	0	0	-6
3-2	0	0	-4	5	5	0	0	0	-6
3-3	0	0	-4	0	0	-5	6	6	0

$v = 0$

The support of any maximin strategy is $\{1\text{-}3, 2\text{-}2, 3\text{-}1\}$, e.g.

$p_{13}^* = \frac{5}{12}$, $p_{22}^* = \frac{4}{12}$, $p_{31}^* = \frac{3}{12}$

Maximize v_1

subject to

$$-2p_{12} - 2p_{13} + 3p_{21} + 4p_{31} \geq v_1$$

$$\vdots$$

$$p_{ij} \geq 0 \qquad i, j = 1, 2, 3$$

$$\sum_{i=1}^{3} \sum_{j=1}^{3} p_{ij} = 1$$

$$v_1 \in \mathbb{R}$$

MAXIMIN/MINIMAX STRATEGIES AND NASH EQUILIBRIA

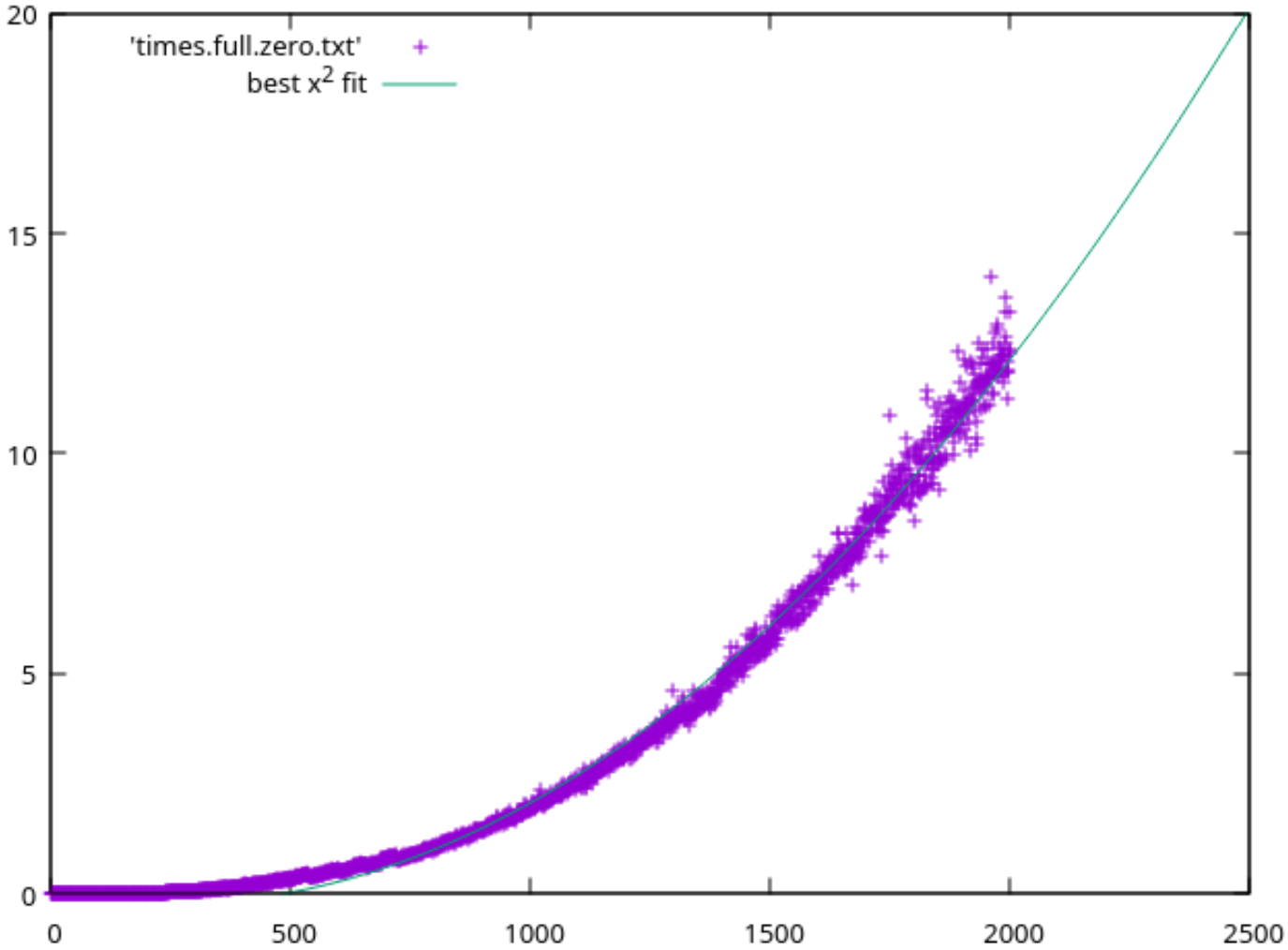
Proposition

Let (p_1^*, p_2^*) be a strategy profile in a matrix game. The following are equivalent:

1. *p_1^* is a maximin strategy and p_2^* is a minimax strategy.*
2. *(p_1^*, p_2^*) is a Nash equilibrium.*

COMPUTATIONAL EXPERIMENTS

- Julia + JuMP + Gurobi, randomly generated matrix games
- Number of strategies vs Solve time in Gurobi



COMPARISON

	Zero-sum	*General-sum*
Nash equilibrium	exists	exists
maxmin/minmax strategies	equivalent to NE	different
unique value	yes	no
equilibrium selection problem	no	yes
computable in $\mathbb{Q}$	yes	no
optimization problem	LP	non-convex POP

DOUBLE ORACLE ALGORITHM

MOTIVATION

- Certain matrix games are too large to solve directly using the baseline linear programming approach
- We will discuss strategy generation method which gradually expands the sets of currently used strategies

WHAT PATH SHOULD THE ROBOT FOLLOW TO AVOID CCTV?

The position of cameras is known.

WHAT PATH SHOULD THE ROBOT FOLLOW TO AVOID CCTV?

The adversary deploys cameras.

WHAT PATH SHOULD THE ROBOT FOLLOW TO AVOID CCTV?

Motion planner

- Path π for the robot
- Finite set of paths Π
- Mixed strategy $p \in \Delta_{\Pi}$
- Loss $\ell(\pi, \mathbf{c})$
- Expected loss

$$\sum_{\pi \in \Pi} \sum_{\mathbf{c} \in C} p(\pi) \cdot q(\mathbf{c}) \cdot \ell(\pi, \mathbf{c})$$

Adversary

- Cost vector $\mathbf{c}$
- Finite set of cost vectors C
- Mixed strategy $q \in \Delta_C$

PLANNING PATHS: EXPERIMENTS

McMahan, Gordon, Blum (ICML 2003)

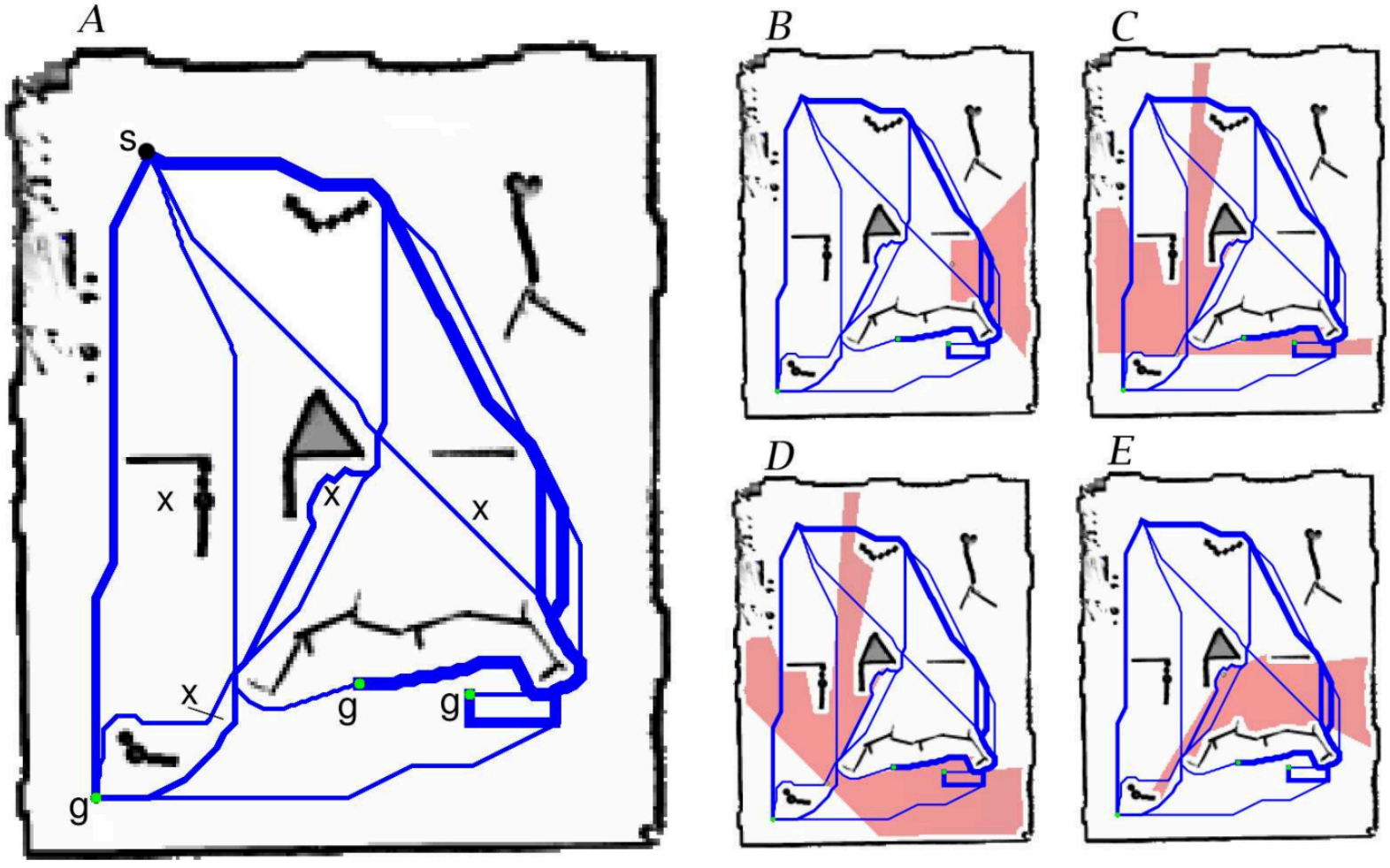
- The gridworld of size up to 269×226
- The robot can move in any of 16 compas directions
- Each cell has cost 1 and a cost proportional to the distance of camera

Computational limits

- Sets Π and C should be reasonably small
- Already $\binom{100}{2} = 4\,950$ positions for 2 cameras in the gridworld 10×10

EXAMPLE OF SOLUTION

McMahan, Gordon, Blum (ICML 2003)



DOUBLE ORACLE ALGORITHM

Input: Any matrix game with strategy sets S_1 and S_2

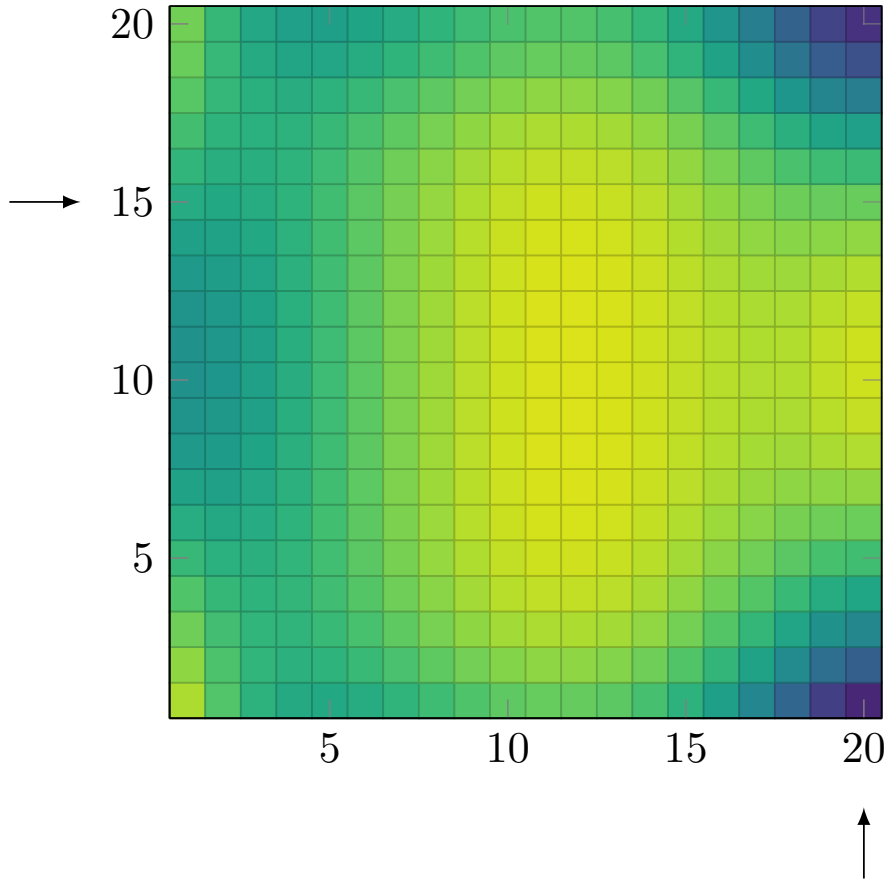
Initialize: Pick small strategy sets $T_1 \subseteq S_1$ and $T_2 \subseteq S_2$

1. Solve the subgame with T_1 and $T_2 \longrightarrow (q_1^*, q_2^*)$
2. Compute the pure best responses

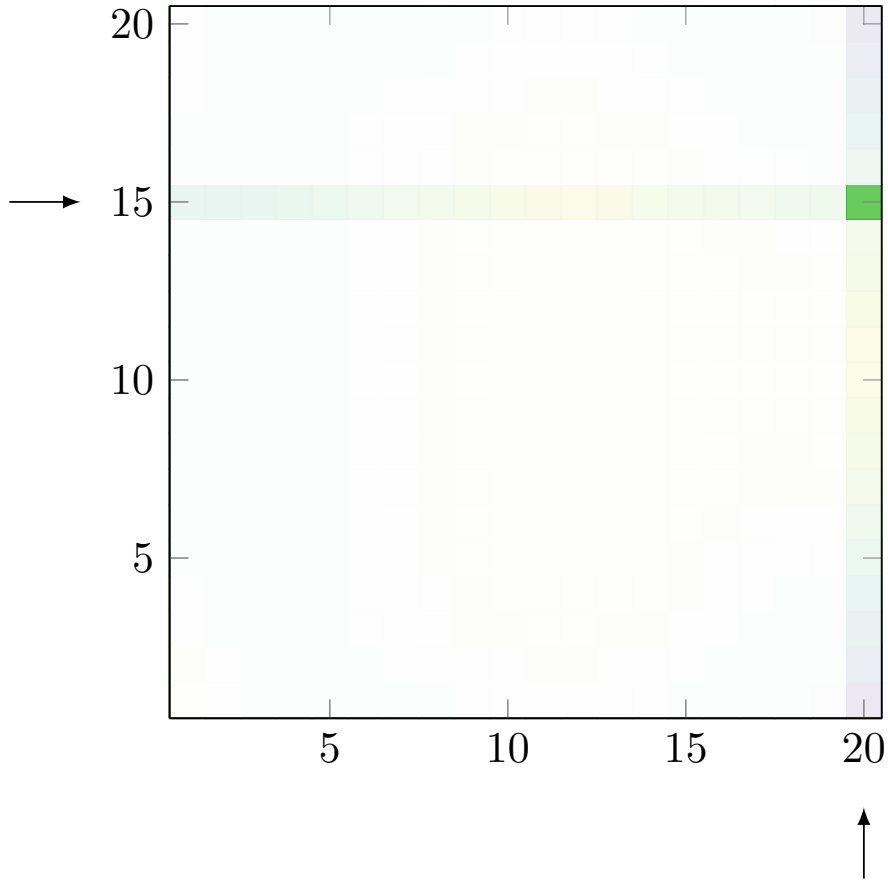
$$s_1 \in \underset{s_1' \in S_1}{\operatorname{argmax}}\, U(s_1', q_2^*) \qquad \text{and} \qquad s_2 \in \underset{s_2' \in S_2}{\operatorname{argmin}}\, U(q_1^*, s_2')$$

3. If $s_1 \in T_1$ and $s_2 \in T_2$, then stop
4. Otherwise add s_1 to T_1 or s_2 to T_2, and go to 1.

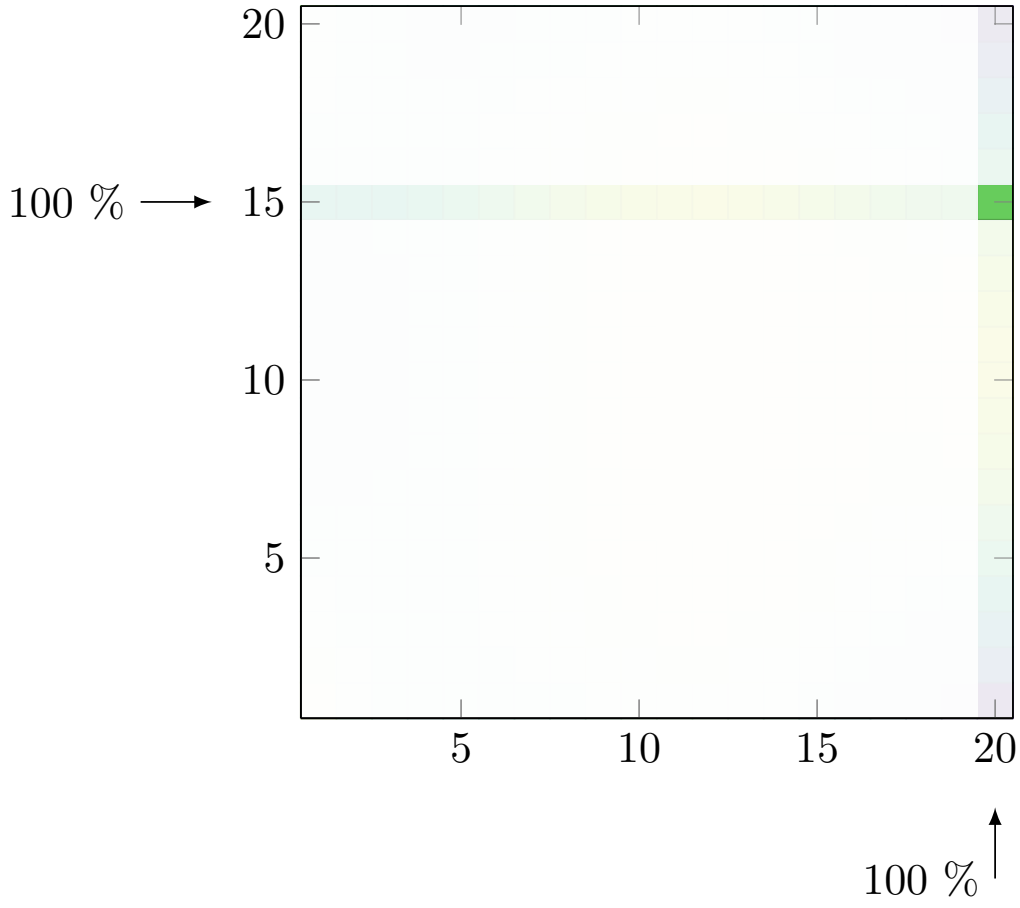
DOUBLE ORACLE ALGORITHM



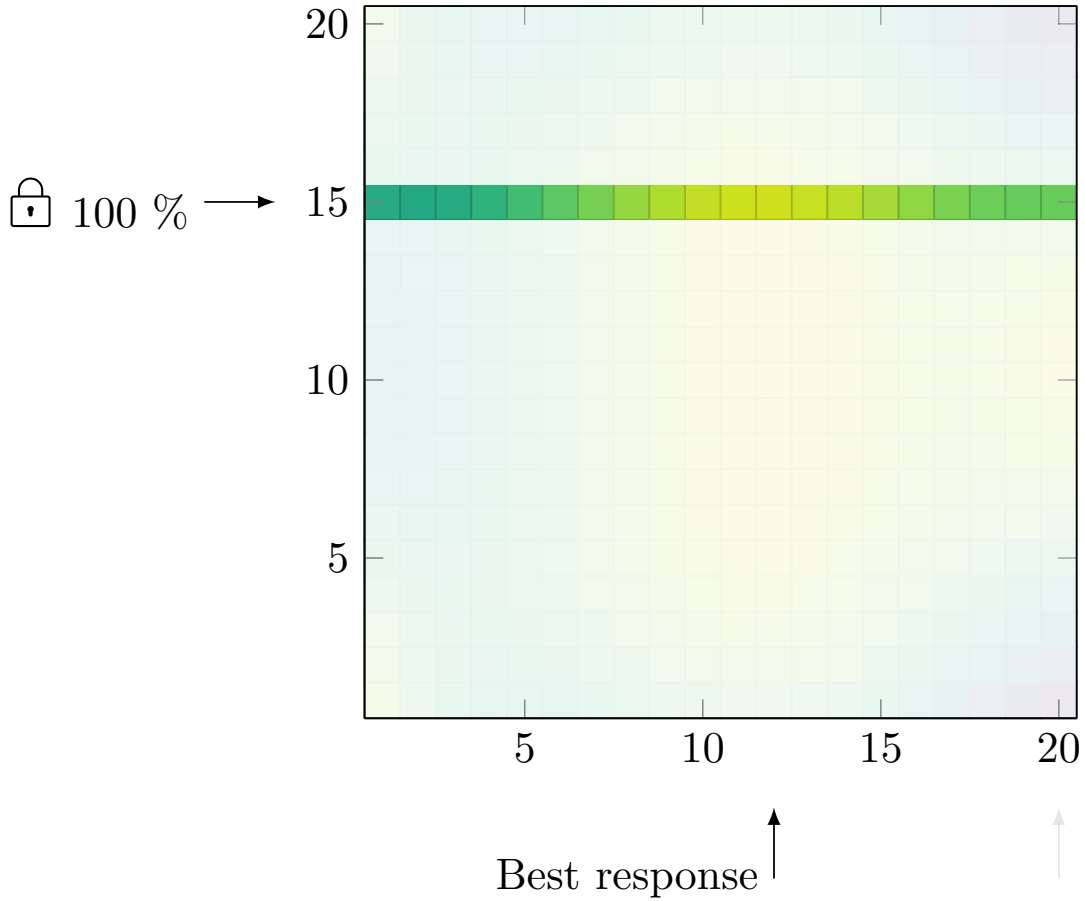
DOUBLE ORACLE ALGORITHM



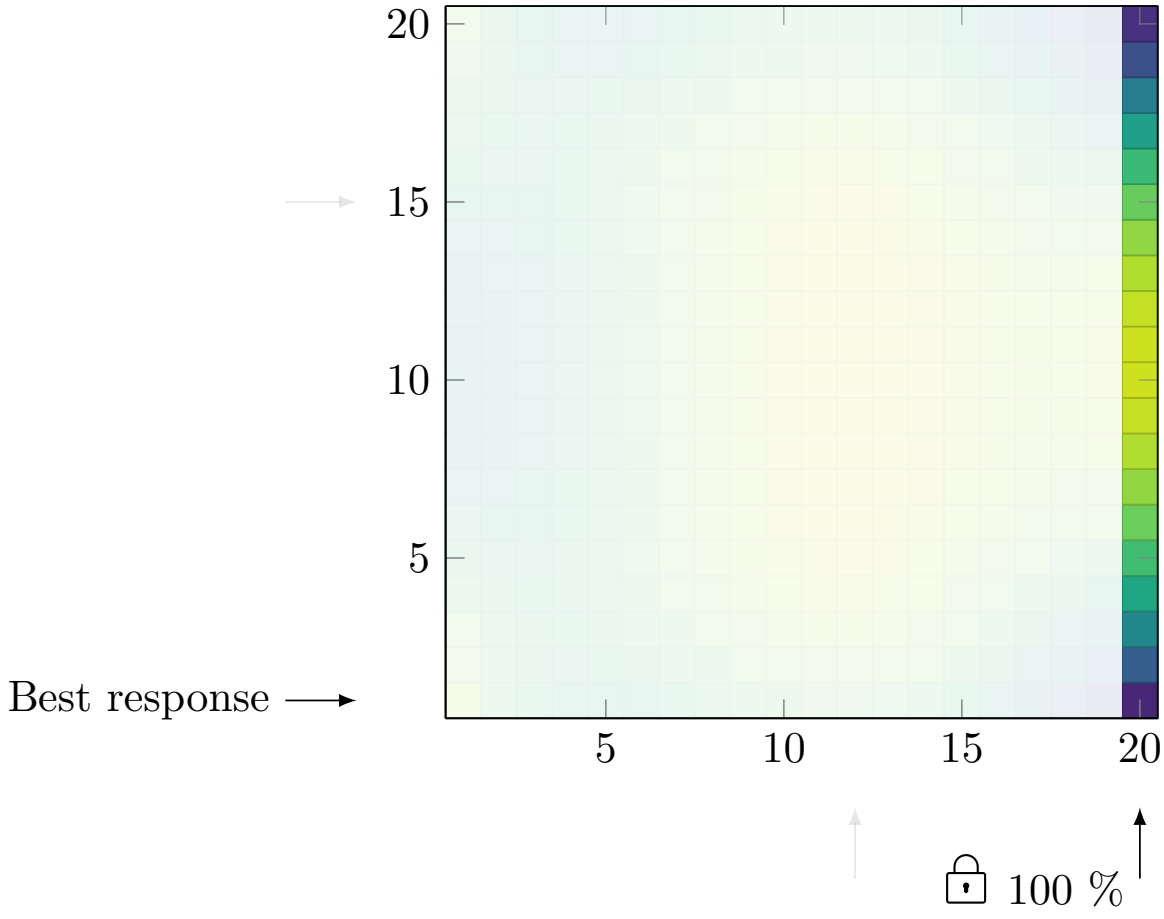
DOUBLE ORACLE ALGORITHM



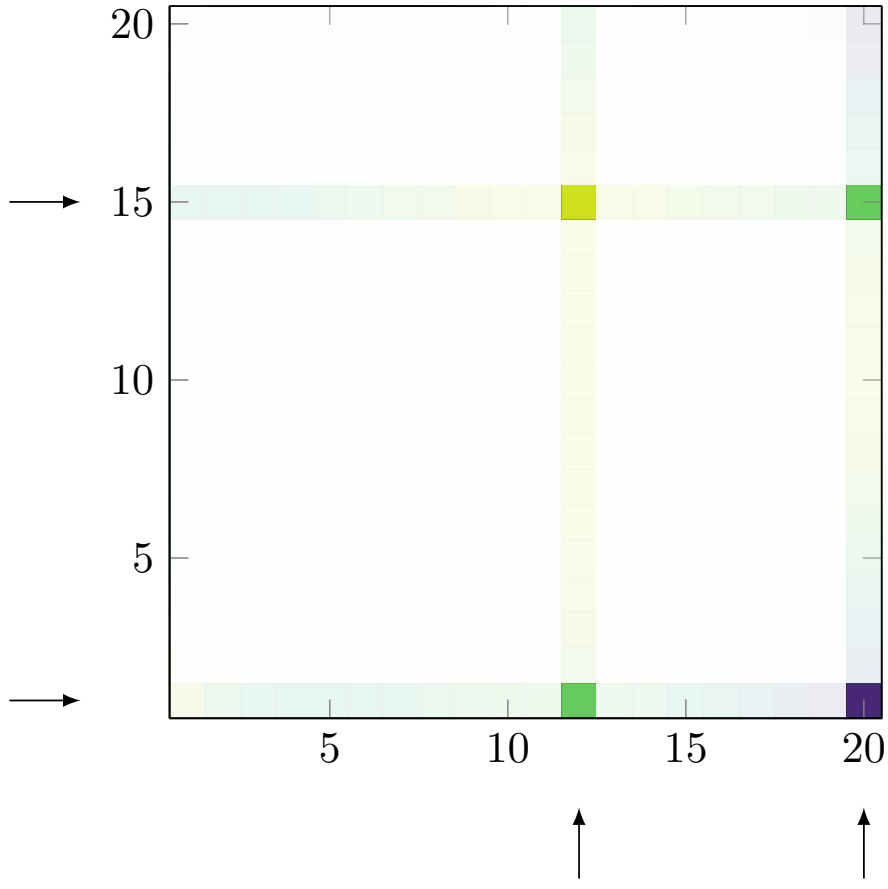
DOUBLE ORACLE ALGORITHM



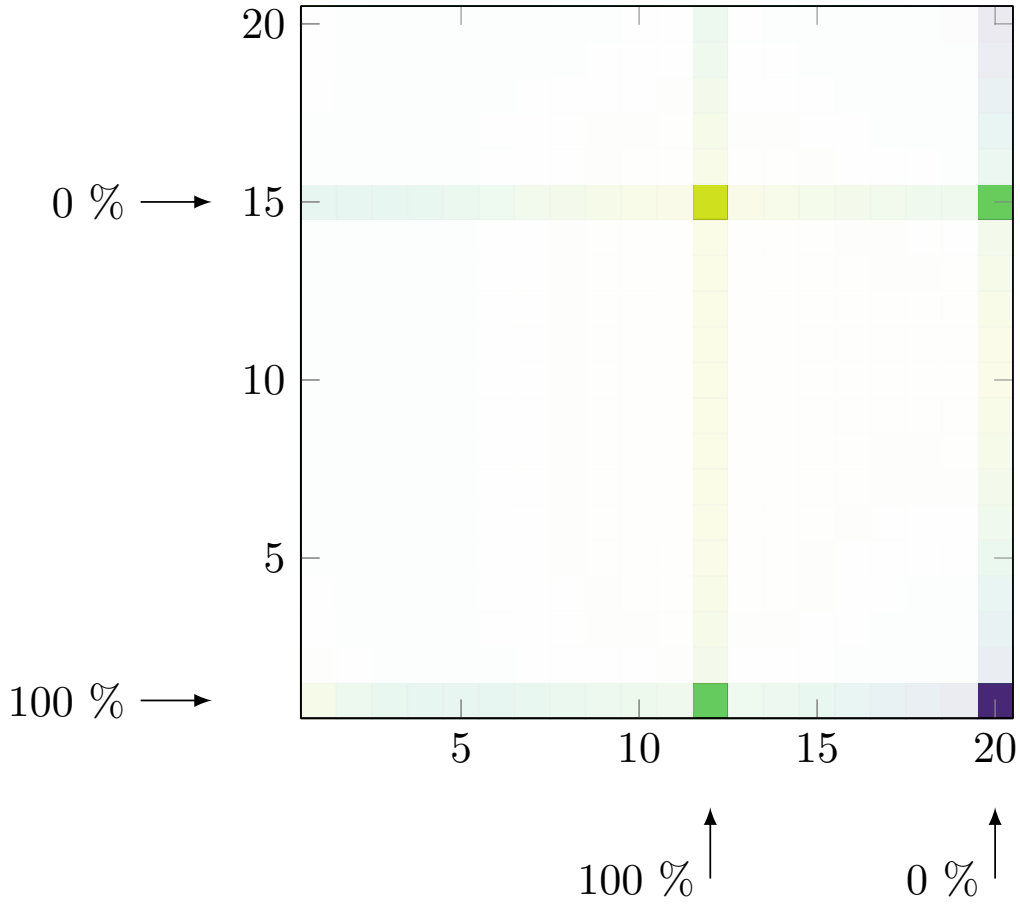
DOUBLE ORACLE ALGORITHM



DOUBLE ORACLE ALGORITHM



DOUBLE ORACLE ALGORITHM



TERMINATION AND CORRECTNESS

Proposition

The DO algorithm terminates and returns a Nash equilibrium (q_1^*, q_2^*).

In each iteration:

1. $v \leq v_u := U(s_1, q_2^*)$
2. $v \geq v_\ell := U(q_1^*, s_2)$
3. If $s_1 \in T_1$ and $s_2 \in T_2$, then $v_\ell = v = v_u$ and (q_1^*, q_2^*) is a NE

ALTERNATIVE TERMINATING CONDITION

- Choose some $\varepsilon > 0$ and stop when

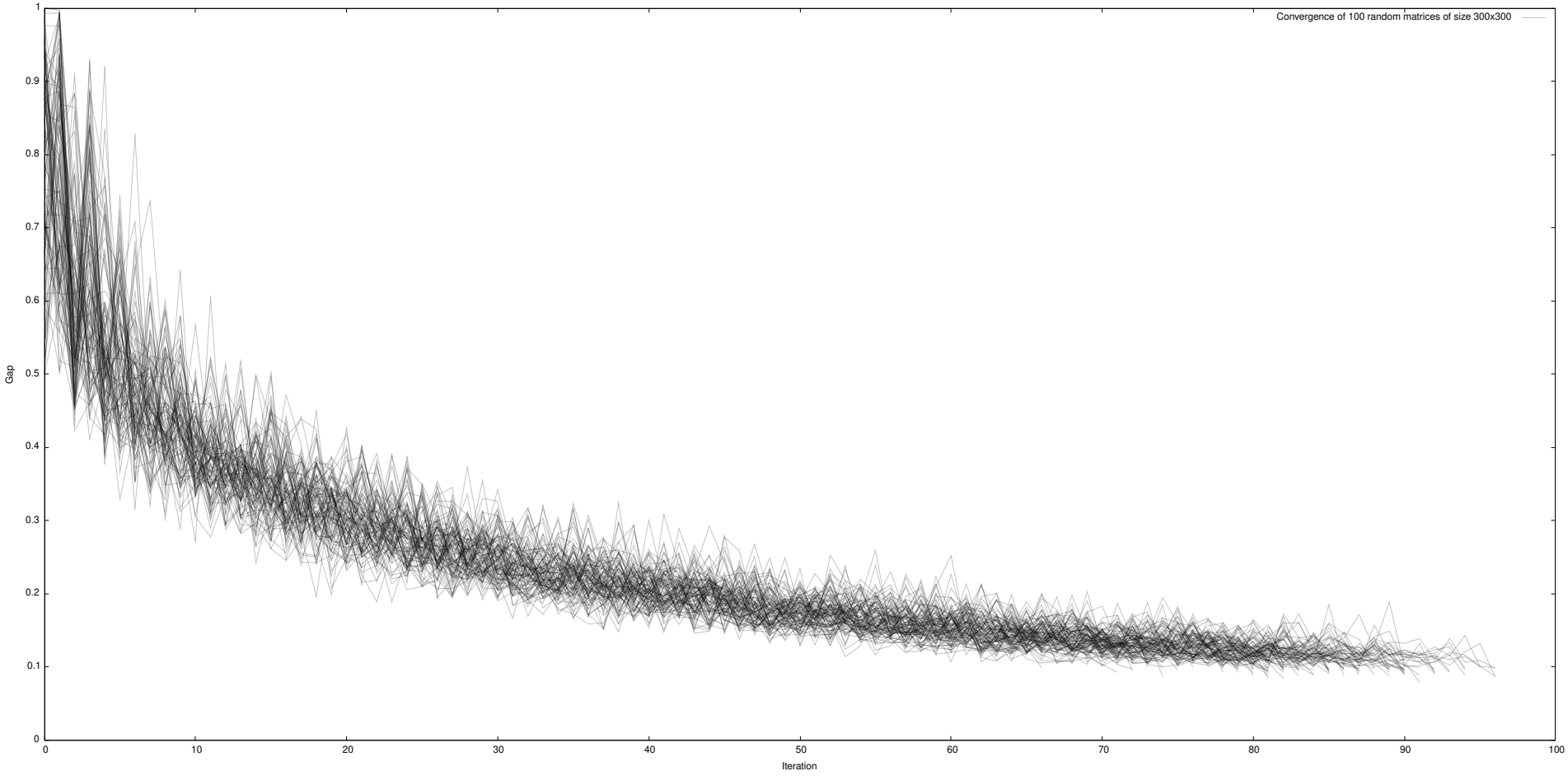
$$v_u - v_\ell \leq \varepsilon$$

- The output (q_1^*, q_2^*) is an ε-Nash equilibrium,

$$U(p_1, q_2^*) - \varepsilon \leq U(q_1^*, q_2^*) \leq U(q_1^*, p_2) + \varepsilon \qquad \forall (p_1, p_2) \in \boldsymbol{\Delta}$$

CONVERGENCE TO ε-EQUILIBRIUM

#iterations vs convergence criterion for 300×300 games



CONVERGENCE TO ε-EQUILIBRIUM