Distributed algorithm for routing multi-flows

Pierre Coucheney, Bruno Gaujal, Panayotis Mertikopoulos

Univ. Versailles Inria CNRS

ICPP, Oct. 2013

Game theory

Founded by J. Von Neumann (also a pioneer in computers) and O. Morgenstern *Theory of Games and Economic Behavior*, published in 1944.

Definition (Roger Myerson, *Game Theory, Analysis of Conflicts*)

Game theory can be defined as the study of mathematical models of conflict and cooperation between intelligent rational decision-makers. Game theory provides general mathematical techniques for analyzing situations in which two or more individuals make decisions that will influence one another's welfare.

Mainly used in mathematical economy.

Game theory in economy, Nobel prizes

- Alvin E. Roth and Lloyd S. Shapley (2012): cooperative games
- Roger B. Myerson (2007, 1951): eq. in dynamic games
- Leonid Hurwicz (2007, 1917-2008): incentives
- Eric S. Maskin (2007, 1950): mechanism design
- Robert J. Aumann (2005, 1930): correlated equilibria
- Thomas C. Schelling (2005, 1921): bargaining
- William Vickrey (1996, 1914-1996): pricing
- Robert E. Lucas Jr. (1995, 1937): rational expectations
- John C. Harsanyi (1994, 1920-2000): Bayesian games, eq. selection
- John F. Nash Jr. (1994, 1928): NE, NBS
- Reinhard Selten (1994, 1930): Subgame perf. eq., bounded rationality
- Kenneth J. Arrow (1972, 1921): Impossibility theorem
- Paul A. Samuelson (1970, 1915-2009): thermodynamics to econ.

Game Theory in networks

Game theory has been used in communication networks for the last 15 years, with increasing popularity.
Number of documents found in Google scholar when searching the following keywords (from E. Altman, 2011)

	Power control	Flow control	Rate control	Access control	Jamming	Routing
networks, G.T.	391,000	174,000	264,000	298,000	17,000	38,000

Game Theory in networks

Game theory has been used in communication networks for the last 15 years, with increasing popularity.
Number of documents found in Google scholar when searching the following keywords (from E. Altman, 2011)

	Power control	Flow control	Rate control	Access control	Jamming	Routing
networks, G.T.	391,000	174,000	264,000	298,000	17,000	38,000

Evolution of the number of documents on routing games (from Publish or Perish):

	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
N. docs.	51	49	190	150	216	324	358	459	459	539	570

Game Theory and distributed algorithms

A recent interest in distributed algorithm community.
Game theoretical vision of distributed systems:
A set of processes (players) compete over ressources. Each process takes its own decisions and want to maximize its own good. The behavior of the whole system is the superposition of the individual behavior of all players.

The ideal case is when the common good (social optimum) coincides with the selfish interest of each player.

- Sheds a new light on issues like: stability, self-optimization, malicious behaviors.
- Allows one to design new distributed algorithms.

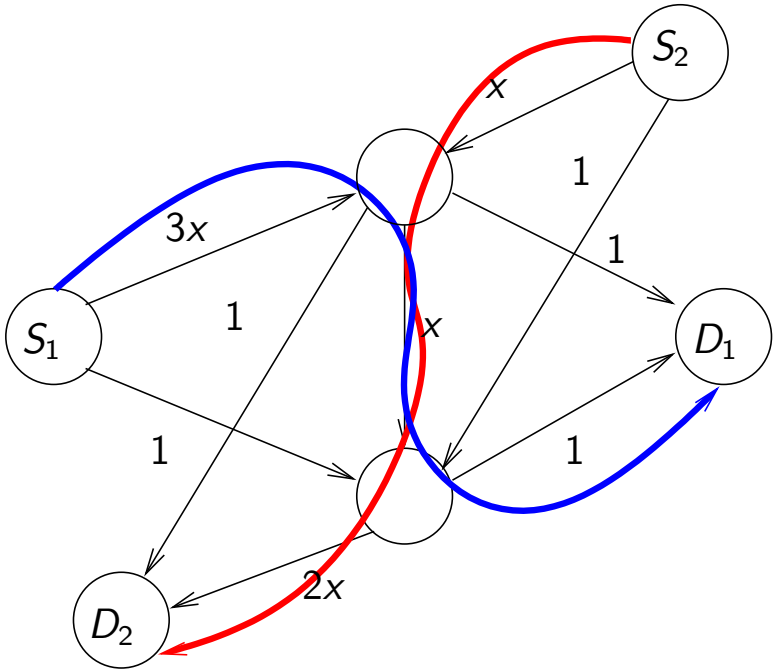
Multi-flow routing

- Consider a directed graph $G = (V, E)$;
- A set of K connections, given by source-sink couples $(S_1, D_1), \ldots (S_K, D_K)$ and rates $r_1, \ldots r_K$;
- Latencies on the edges $\ell_e(x)$.

End to end latencies:

$c_1 = 3r_1 + (r_1 + r_2) + 1$

$c_2 = r_2 + (r_1 + r_2) + 2r_2$



Find a traffic flow that minimizes end-to-end latencies of the connections.

Latency functions

- The affine case: $\ell_e(x) = ax + b$.
- The queueing case: If each edge has a limited capacity (μ_e), then the latency is
  $\ell_e(x) = \dfrac{x}{\mu_e - x}$
- The case of a wifi network

$$\ell_e(x) = \frac{x\,(T_{DATA} + T_{ACK} + 2T_{TBO}(x) + 2T_W(x))}{L_{TCP}}$$

  $L_{TCP} = 8000$ bits is the size of a TCP packet, T_{ACK} is the transmission times of TCP ACK (approx. 1.091 ms), T_{DATA} the transmission times of a TCP data packet (about 1.785 ms), T_W and T_{TBO} are the mean total time lost due to collisions and backoffs.
- The real case: The latency function does not have a closed formula, may decrease with the load and is not convex in general. It can only be measured.

Several types of connections

- Atomic case: K connections, each connection sends all its traffic along one path.
- Atomic-splittable case: K connections, each connection can split its traffic along several paths
- Non-atomic case: K connection types (one type is a pair (S, D)), each type is made of an infinite number of small packets, each one having an infinitesimal impact.

Applications

These kinds of problems appear in

- Ground transportation design (non-atomic),
- communication networks (wifi networks, ad-hoc networks) (atomic splittable),
- vertical handover in cellular networks (atomic),

  bipartite graph,
  delay replaced by throughput,
  congestion replaced by interference,
- spectrum management in MIMO systems (atomic splittable).

Game Model

This problem can be seen as a routing game where
the players are the connections (also called users),
the actions of one connection are the flows allocated in each path from source to sink
and the costs are the end-to-end latencies.

Game Model

This problem can be seen as a routing game where
the players are the connections (also called users),
the actions of one connection are the flows allocated in each path from source to sink
and the costs are the end-to-end latencies.

Connections are non-cooperative: they want to minimize their personnal latency and do not care about the other connections.
A Nash equilibrium is a configuration where no connection can improve on iys latency by changing its path.

Nash Equilibria Properties

- Congestion games always admit Nash equilibria (not necessarily unique). [Nash], [Beckmann et al. 56],
- Efficient computation of Nash and optimal flows [Dafermos-Sparrow, 69],
- Quantification of the cost of a lack of cooperation (price of anarchy) [Roughgarden-Tardos, 02]

Nash Equilibria Properties

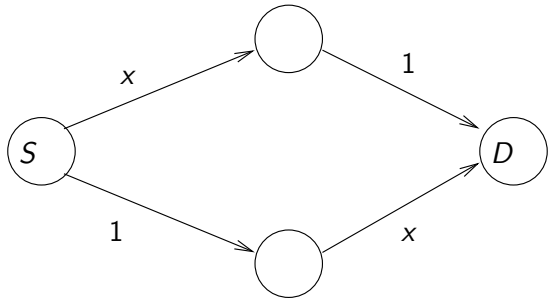
- Congestion games always admit Nash equilibria (not necessarily unique). [Nash], [Beckmann et al. 56],
- Efficient computation of Nash and optimal flows [Dafermos-Sparrow, 69],
- Quantification of the cost of a lack of cooperation (price of anarchy) [Roughgarden-Tardos, 02]

Non-intuitive behavior of Nash equilibria:

- Price of anarchy is unbounded (4/3 for affine latencies).
- Braess paradox: adding routes may increase the latencies for all the connections.
- Adding cooperation between connections may increase the latencies for all the connections.

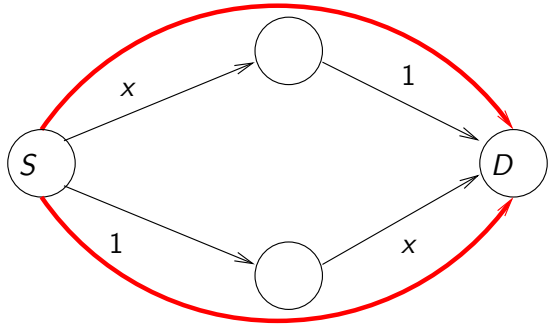
Braess Paradox (from Roughgarden, 2000)

One non-atomic connection type (with rate 1).



Braess Paradox (from Roughgarden, 2000)

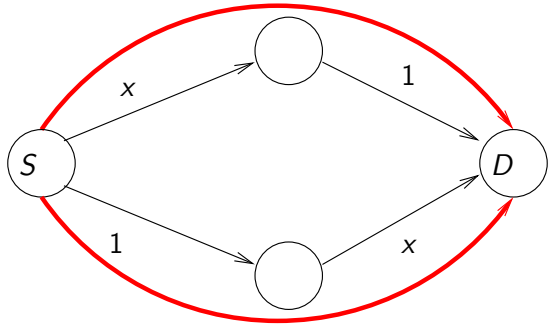
One non-atomic connection type (with rate 1).



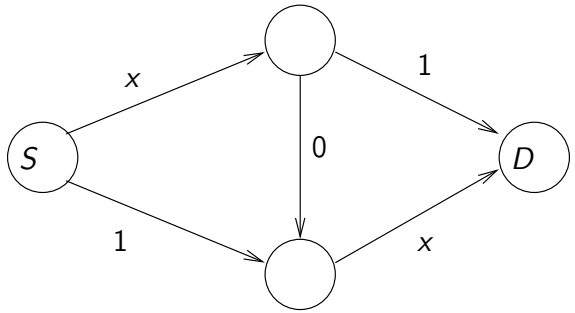
In Nash equilibrium the flow splits evenly, packet latency is 1.5

Braess Paradox (from Roughgarden, 2000)

One non-atomic connection type (with rate 1).

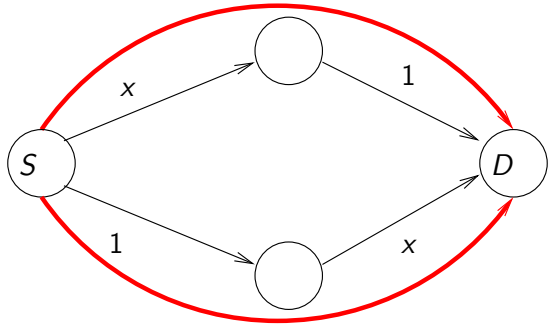


In Nash equilibrium the flow splits evenly, packet latency is 1.5

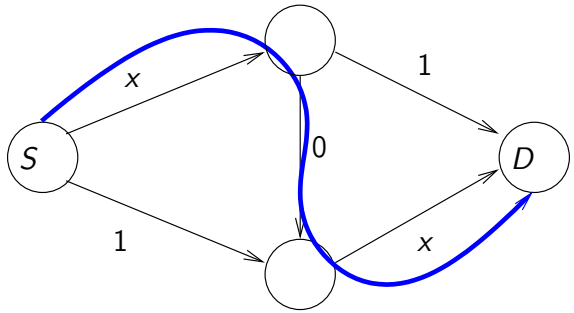


Braess Paradox (from Roughgarden, 2000)

One non-atomic connection type (with rate 1).

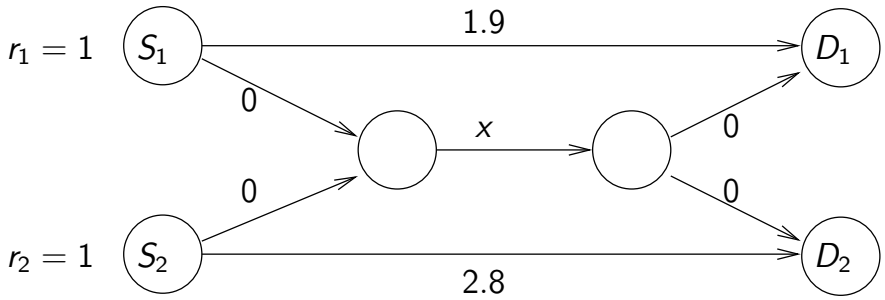


In Nash equilibrium the flow splits evenly, packet latency is 1.5

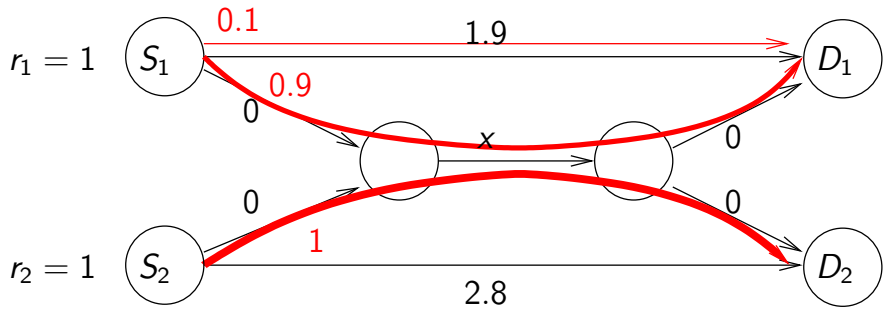


By adding a fast link (0 latency), packet latency is 2 at Nash equilibrium

Cooperation paradox (Cominetti et al. 2009)

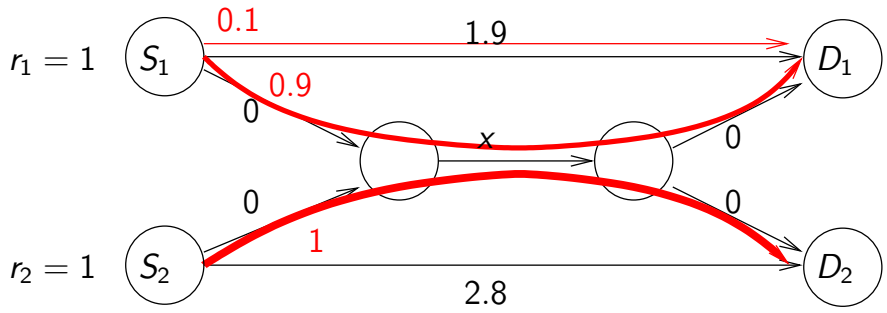


Cooperation paradox (Cominetti et al. 2009)

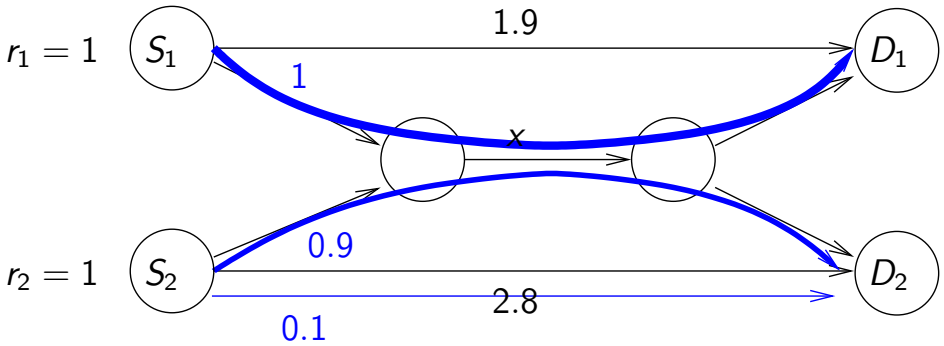


In Nash equilibrium, the latencies for flows of types 1 and 2 is 1.9

Cooperation paradox (Cominetti et al. 2009)



In Nash equilibrium, the latencies for flows of types 1 and 2 is 1.9



By adding cooperation within type 2 (they want to minimize their total latency instead of individual ones), the new Nash equilibrium has latency 1.9 for type 1 and 1.99 for type 2.

Learning algorithm

We want to design a distributed algorithm (executed by each user) that learns the Nash equilibria. Each user computes its route according to a sequential process that should satisfies the following desired properties:

- only uses local information (each connection only sees its own latency) (stateless).
- does not use coordination between the users (time-oblivious)
- tolerates outdated measurements on the value of the current end-to-end latency (delay-oblivious)
- tolerates random perturbations or inacuracies on the values of the latencies (robust)
- converges fast even if the number of users is very large (scalable)

Main Ideas

Here are the main ingredients to design the algorithm:

- Randomize the choices of actions
- Learn from past mistakes
- Go back and forth between time-continuous models and discrete models.

Atomic routing game

Main characteristics:

- Let us consider a game $\mathfrak{G} = (\mathcal{K}, \mathcal{A}, g)$ with a finite number of players.
- each player $k \in \mathcal{K}$ chooses an action $a_k \in \mathcal{A}_k$ (finite).
- the gain for user k is denoted $g_k(a_k, a_{-k})$.

Atomic routing game

Main characteristics:

- Let us consider a game $\mathfrak{G} = (\mathcal{K}, \mathcal{A}, g)$ with a finite number of players.
- each player $k \in \mathcal{K}$ chooses an action $a_k \in \mathcal{A}_k$ (finite).
- the gain for user k is denoted $g_k(a_k, a_{-k})$.

Routing game

- We consider a routing game.
- players are connections;
- actions are routes from source to destination;
- gains are opposite of end-to-end latencies.

Best Response algorithm

BR

Repeat

Choose randomly a user $k \in \mathcal{K}$

For All route $\alpha \in \mathcal{A}_k$

compute the gain $g_k(\alpha, a_{-k})$

Choose one α^* that maximizes the gain

Theorem

Best Response (BR) converges to a Nash equilibrium in finite time a.s..

The game $\mathfrak{G}$ is a potential game with potential $F(a) = \sum_k g_k(a)$.

Best Response algorithm assessment

- Stateless: yes and no
- Robust: no
- Time-oblivious: no
- Delay-oblivious: no
- Scalable: no

Futhermore, it may converge to any Nash equilibria, the ratio with the social optimal can be arbitrarily large.

Gibbs Sampling Algorithm

Let us randomize the choices:

GSA

For ever

Choose randomly a user $k \in \mathcal{K}$;

For All action $\alpha \in \mathcal{A}_k$

$$r_\alpha^k := \exp(\frac{1}{\tau} g_k(a, s_{-k}));$$

Pick $\alpha \in \mathcal{A}_k$ with probability $\frac{r_\alpha^k}{\sum_{\beta \in \mathcal{A}_k} r_\beta^k}$;

Theorem ([Blume, 93])

The Gibbs Sampling Algorithm GSA, *finds an "optimal" equilibrium with high probability, when* τ *is close to* 0.

Gibbs Sampling Algorithm assessment

- Stateless: yes and no
- Time-oblivious: no
- Robust: no
- Delay-oblivious: no
- Scalable: no
- Convergence becomes very slow when τ gets close to 0.

Non-Asynchronous Revisions

A partial synchronous revision process is given by probability measures ρ_a over the set $2^{\mathcal{K}}$,
(Example: each player decides to play independently)

GSA-Rev

For All time step n

Choose a set of users V according to ρ_a.

For All user $u \in V$

For All action $\alpha \in \mathcal{A}_u$

$$r_\alpha^u := \exp(\frac{1}{T} g_u(\alpha, a_{-u}));$$

Pick action $\alpha \in \mathcal{A}_u$ with probability $\frac{r_\alpha^u}{\sum_{\beta \in \mathcal{A}_u} r_\beta^u}$;

Relation with best response

Theorem (Coucheney, 2010)

The asymptotically stable states of `GSA-Rev` *are recurrent vertices of the best response graph $\mathcal{B}$ (not Nash equilibria in general).*

Corollary (Durand, 2012)

If the Nash equilibria are strict and if the support of ρ_a is isolable, then the asymptotically stable states of `GSA-Rev` *are Nash equilibria.*

Let $\mathcal{B}$ be the best response graph of the game: there is an arc between (a, b) if there exists a revision set V such that $a_{-V} = b_{-V}$ and

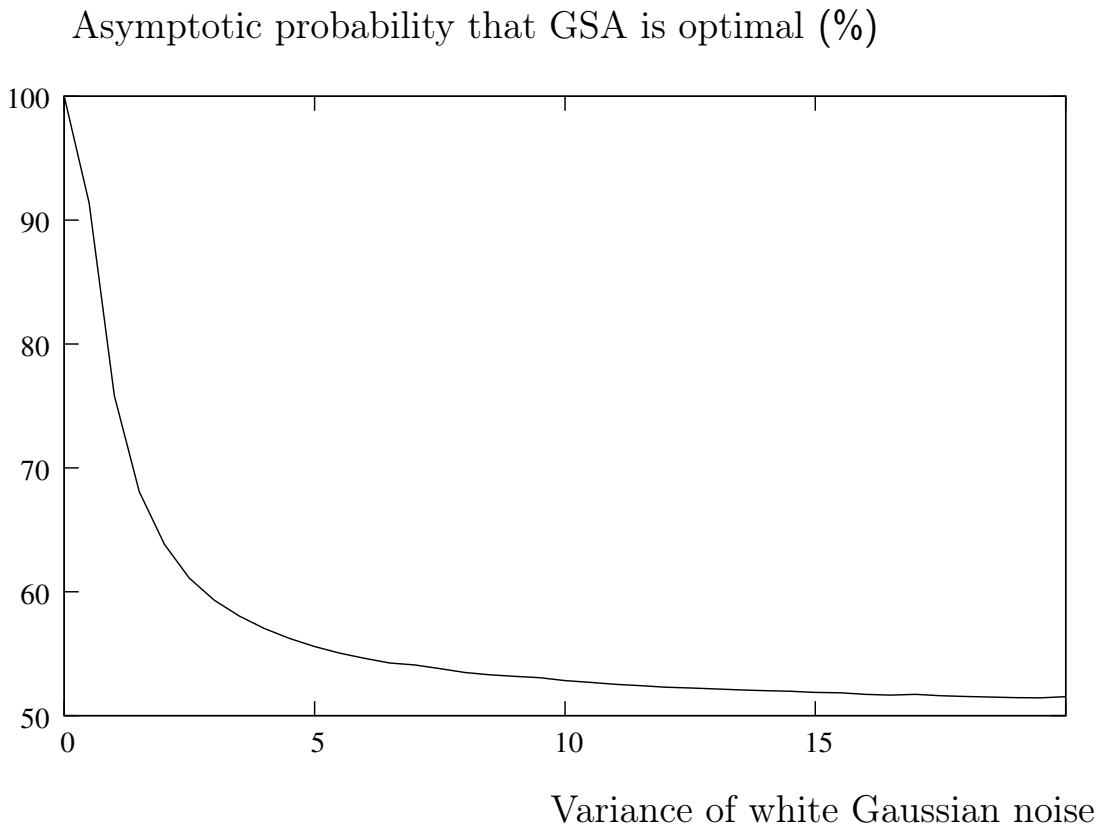
$$\forall u \in V, b_u \in BR_u(a).$$

Noisy measurements

Player k can only measure her gain $g_k(a)$ up to some noise (*e.g.* i.i.d. white Gaussian noise).

Noisy measurements

Player k can only measure her gain $g_k(a)$ up to some noise (*e.g.* i.i.d. white Gaussian noise).
Consider a game with one player and two actions (with gains 2 and 1).



Score and Choice

The main problem of the Gibbs sampling algorithm comes from its Markovian nature. One way to make the learning procedure robust is to take into account the history of the process.

Score and Choice

The main problem of the Gibbs sampling algorithm comes from its Markovian nature. One way to make the learning procedure robust is to take into account the history of the process.

Score: Each user plays at discrete times $s = 0, 1, \ldots, t$. The score is a discounted average of the payoff:

$$y_\alpha(t) = g_\alpha(t) + \lambda y_\alpha(t-1),$$

or

$$y_\alpha(t) = \sum_{s=0}^{t} \lambda^{t-s} g_\alpha(s),$$

where $\lambda \in (0, +\infty)$ is the model's discounting parameter.

Score and Choice

The main problem of the Gibbs sampling algorithm comes from its Markovian nature. One way to make the learning procedure robust is to take into account the history of the process.

Score: Each user plays at discrete times $s = 0, 1, \ldots, t$. The score is a discounted average of the payoff:

$$y_\alpha(t) = g_\alpha(t) + \lambda y_\alpha(t-1),$$

or

$$y_\alpha(t) = \sum_{s=0}^{t} \lambda^{t-s} g_\alpha(s),$$

where $\lambda \in (0, +\infty)$ is the model's discounting parameter.

Choice: The choice of the next action is still based on a Gibbs distribution. The probability to select route α is:

$$x_\alpha = \frac{\exp(y_\alpha)}{\sum_\beta \exp(y_\beta)} = Gibbs_\alpha(y), \quad u \in \mathcal{A}$$

Continuous time

The continuous time version of the (score,choice) equation

$$y(t) = \sum_{s=0}^{t} \lambda^{t-s} g(s), \quad x(t) = Gibbs(y(t))$$

is

$$y(t) = \int_0^t \lambda^{(t-s)} g(s) ds,$$
$$x(t) = Gibbs(y(t)),$$

By differentiating w.r.t. time , (ED1)

$$\dot{y}(t) = g(t) - Ty(t)$$
$$x(t) = Gibbs(y(t))$$

by setting $T \stackrel{\text{def}}{=} \log(1/\lambda)$.

Entropy Driven dynamics (II)

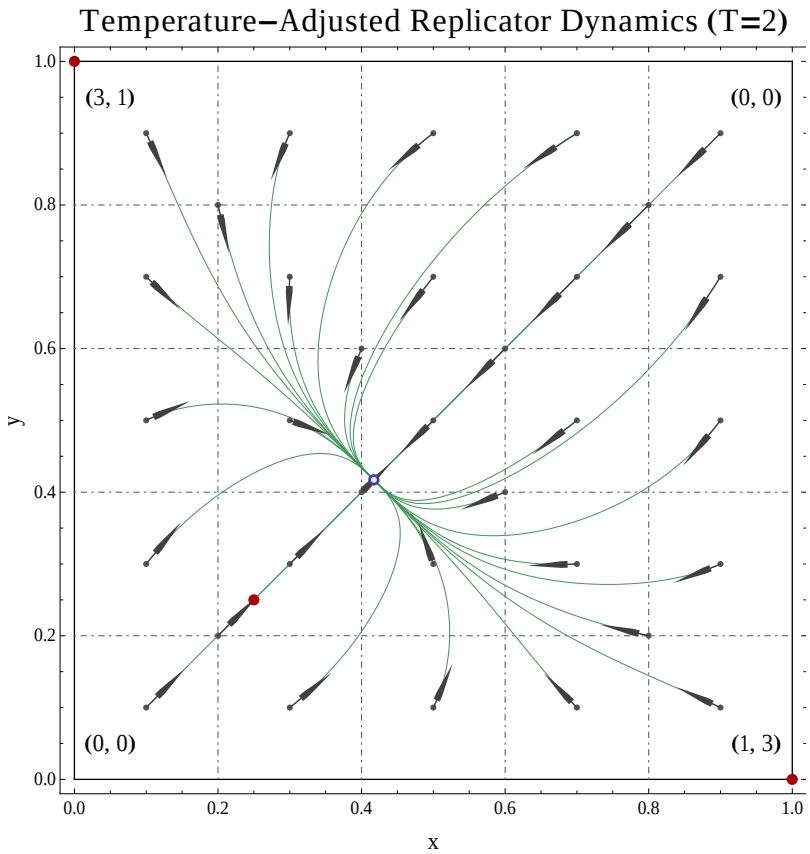
The two stage dynamics can be written in x instead (with some algebra).Entropy-driven dynamics (ED2):

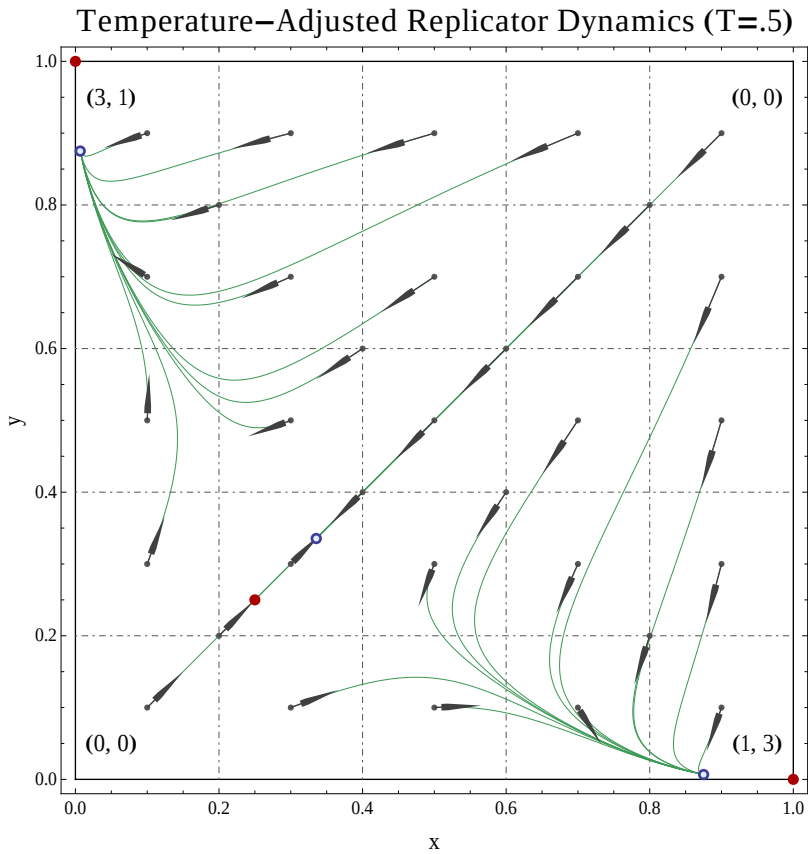
$$\dot{x}_\alpha = x_\alpha \left(g_\alpha(x) - \sum_\beta x_\beta g_\beta(x) \right) - T x_\alpha \left(\log x_\alpha - \sum_\beta x_\beta \log x_\beta \right)$$

for $T = 0$, (ED2) freezes to the ordinary replicator dynamics:

$$\dot{x}_\alpha = x_\alpha \left(g_\alpha(x) - \sum_\beta x_\beta g_\beta(x) \right) .$$

Entropy Driven dynamics (III)





Lyapunov function and convergence

The game $\mathfrak{G}$ is a finite potential game with potential F.
The function $U(x) \stackrel{\text{def}}{=} Th(x) - F(x)$ is Lyapunov for (ED):
for any interior orbit $x(t)$ of (ED), $\frac{d}{dt}U(x(t)) \leq 0$ with equality iff $x(0)$ is a rest point.

Lyapunov function and convergence

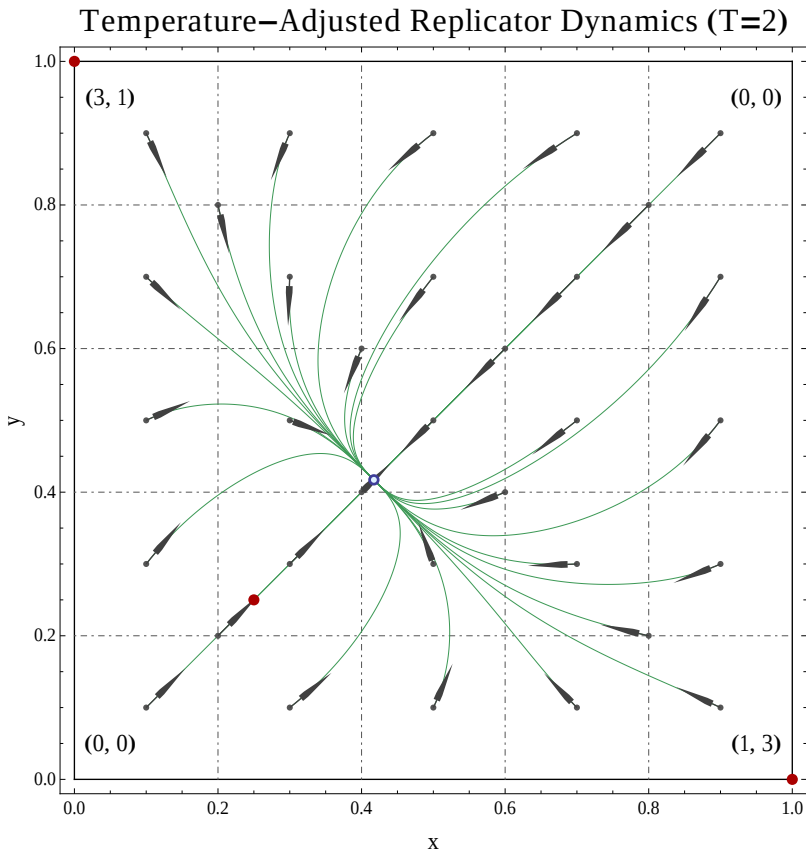
The game $\mathfrak{G}$ is a finite potential game with potential F.
The function $U(x) \stackrel{\text{def}}{=} Th(x) - F(x)$ is Lyapunov for (ED):
for any interior orbit $x(t)$ of (ED), $\frac{d}{dt}U(x(t)) \leq 0$ with equality iff $x(0)$ is a rest point.

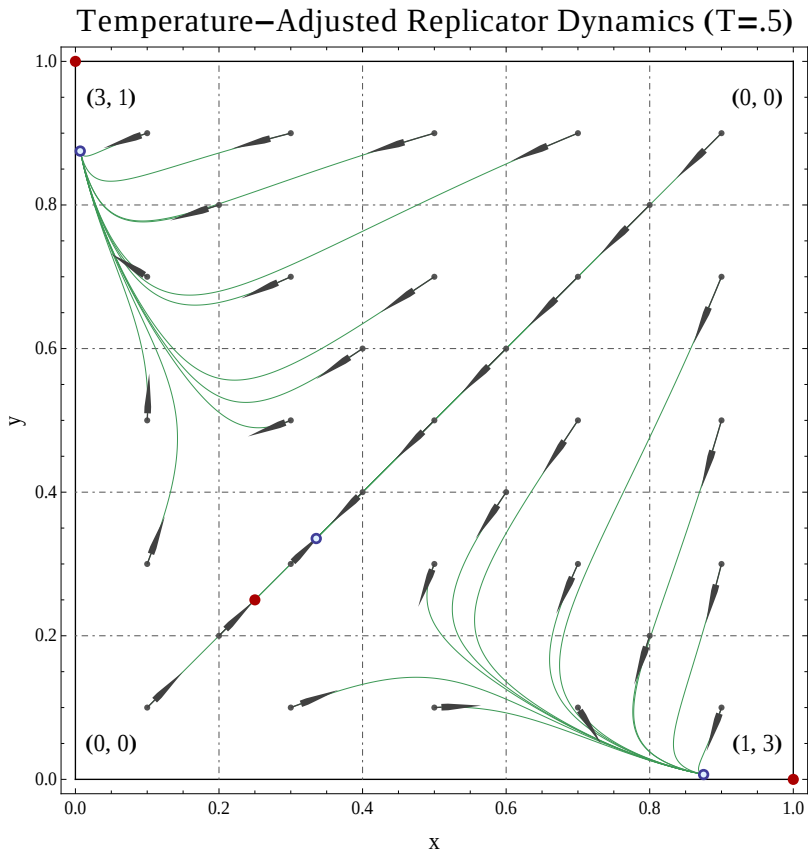
Theorem (Folk theorem for (ED))

Let $x(t)$ be a solution orbit of the entropic dynamics (ED) for the potential game $\mathfrak{G}$. Then:
For $T > 0$ and $x(0) > 0$, $x(t)$ converges to the rest points of (ED).

The rest points of the entropy-driven dynamics (ED) converge to the Nash equilibria of $\mathfrak{G}$ when the temperature approaches 0.

Convergence of Driven dynamics





Discrete Algorithm on Scores

Stochastic discrete version of (ED1): $\dot{y} = g - Ty, \quad x = Gibbs(y)$

Discrete Algorithm on Scores

Stochastic discrete version of (ED1): $\dot{y} = g - Ty, \quad x = Gibbs(y)$

ED-Score

Repeat

For each player $k \in \mathcal{U}$

Choose action $a_k(n)$ according to $Gibbs\left(Y_\beta^k(n)\right)$

compute the new score:

$$Y_{a_k}^k(n+1) := Y_{a_k}^k + \varepsilon_{n+1}\frac{1}{X_{a_k}}\left(g_k(a) - TY_{a_k}^k\right).$$

The discrete process $(Y(n))$ generated by Algorithm ED-Score is a stochastic approximation of the Entropy Driven dynamics (ED1):

$$\frac{\mathbb{E}\left[Y(n+1) - Y(n)|Y(n)\right]}{\varepsilon_{n+1}} = g(X(n)) - TY(n)$$

Discrete Algorithm on Strategies (ED2)

ED-Strat

Repeat

For Each Player $k \in \mathcal{N}$ simultaneously

Pick action $\hat{\alpha}_k$ according to the mixed strategy X_k

For Each Player $k \in \mathcal{N}$

$\hat{g}_k \leftarrow g_k(\hat{\alpha})$ # current payoff

For Each action $\alpha \in \mathcal{A}_k$

$$X_\alpha \leftarrow X_\alpha + \varepsilon_{n+1}\left[-TX_\alpha\left(\log X_\alpha - \sum\nolimits_\beta X_\beta \log X_\beta\right) + \hat{g}\left(\mathbf{1}_{\hat{\alpha}_k=\alpha} - X_\alpha\right)\right]$$

The discrete process $(X(n))$ generated by Algorithm ED-Strat is a stochastic approximation of the entropy driven dynamics.

Stability Properties

For every positive learning temperature T, the sequence of iterates generated by Algorithm ED-Strat are stochastically stable, i.e. $\sup_n ||Y(n)|| < \infty$ almost surely.

This is not true for ED-Score.

Stability Properties

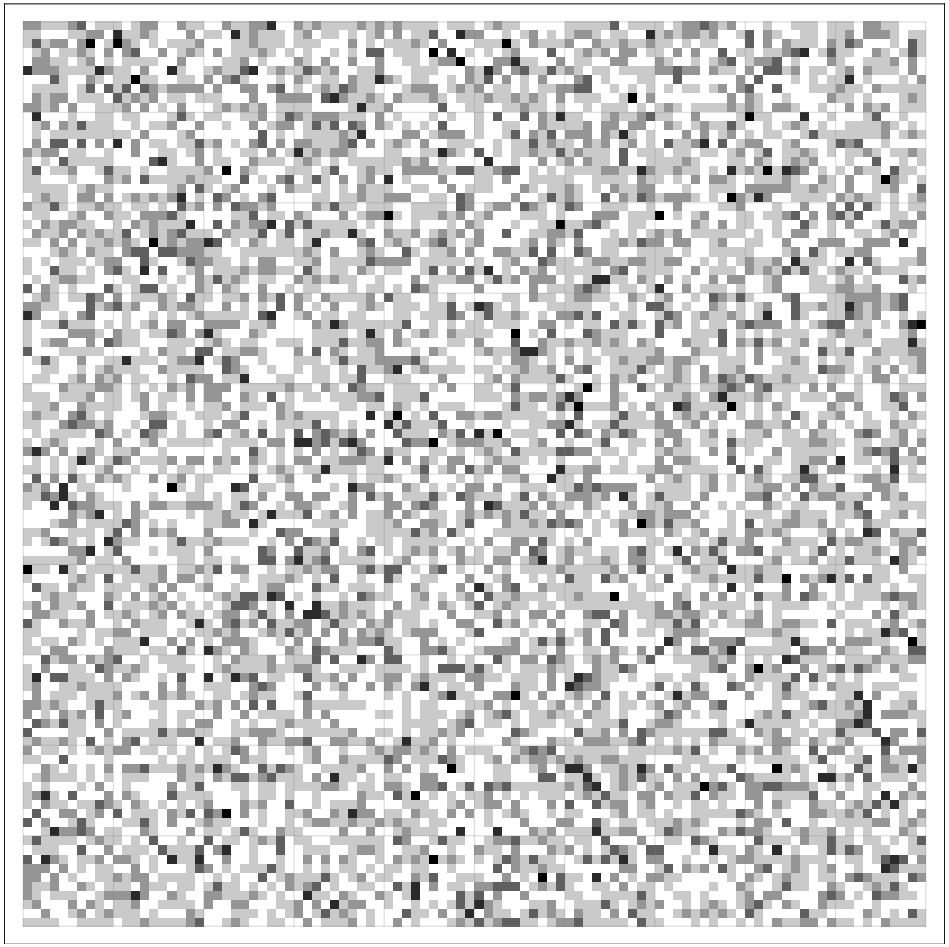
For every positive learning temperature T, the sequence of iterates generated by Algorithm ED-Strat are stochastically stable, i.e. $\sup_n ||Y(n)|| < \infty$ almost surely.

This is not true for ED-Score.

Theorem

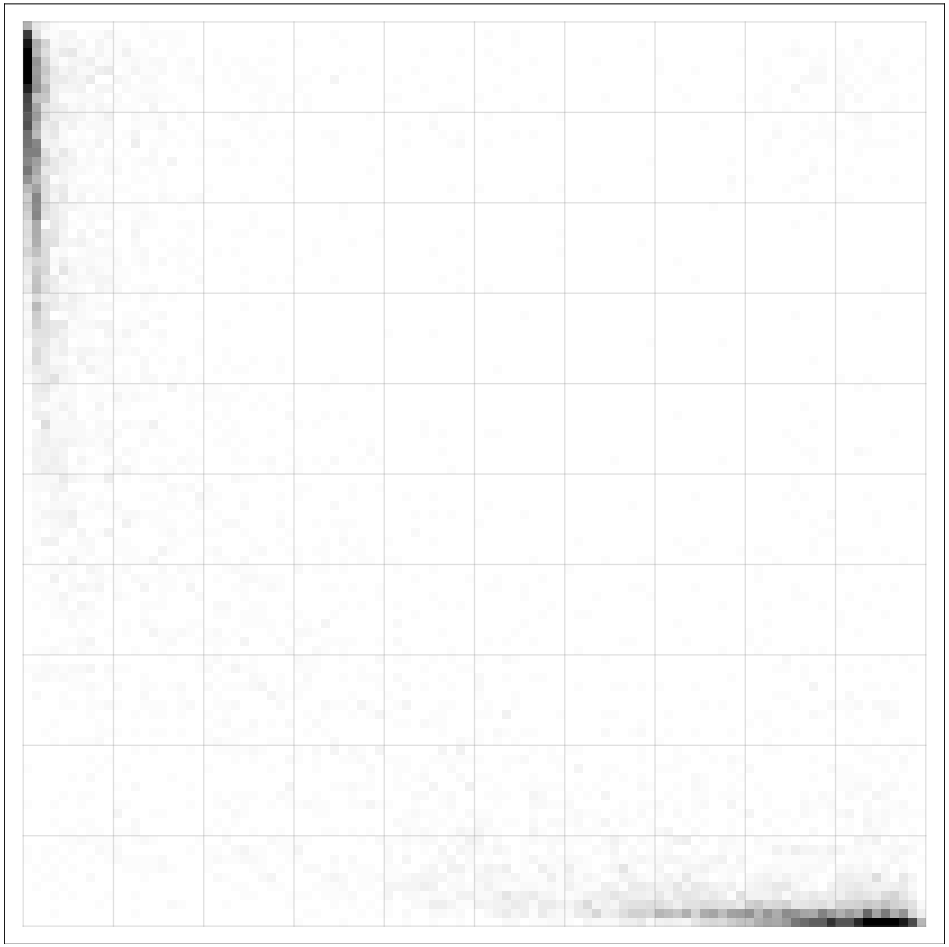
The discrete algorithm ED-Strat converges to the stable rest points of the continuous ED dynamics (that are very close to the Nash equilibria of the game when T is small) .

Convergence rates



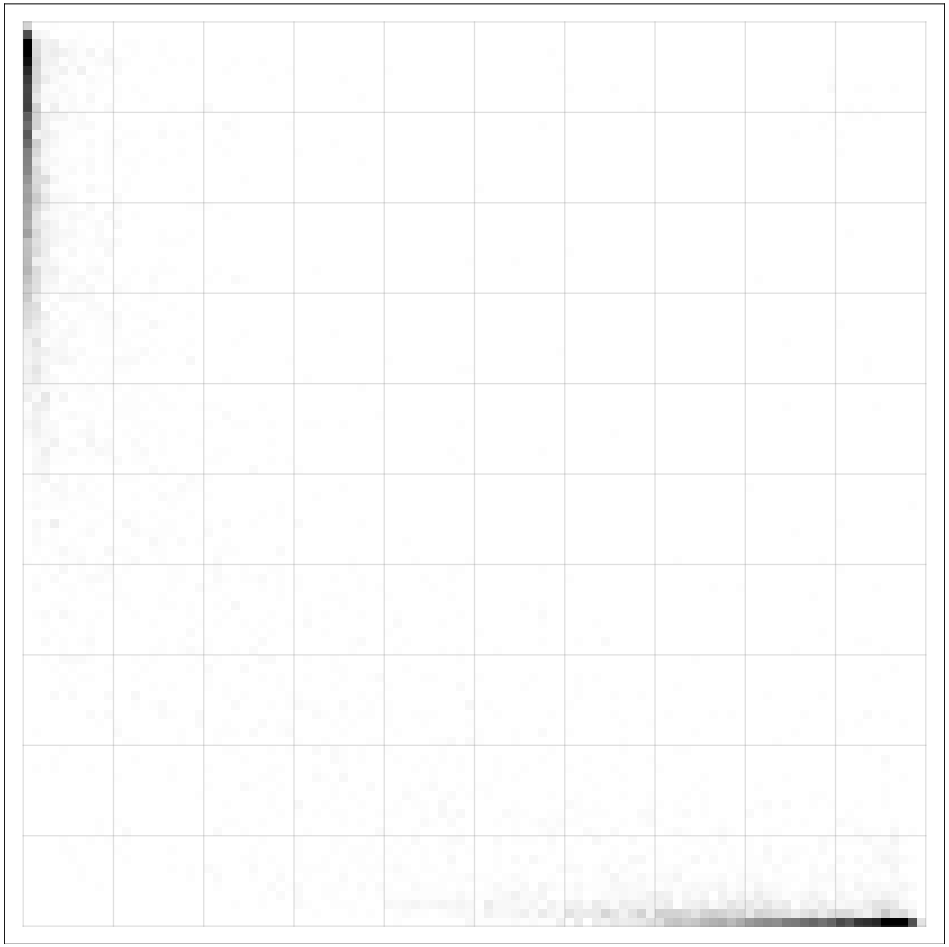
Density for 10^4 executions, after 0,10,20 or 30 iterations. (play a film).

Convergence rates



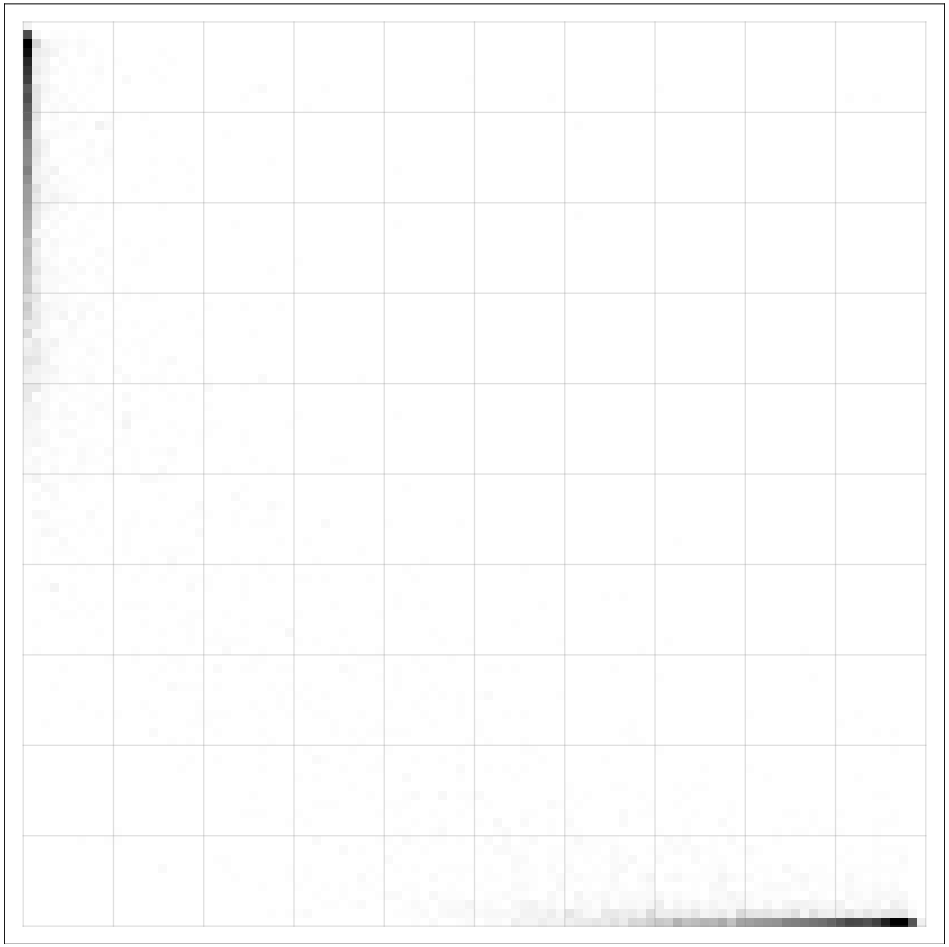
Density for 10^4 executions, after 0,10,20 or 30 iterations. (play a film).

Convergence rates



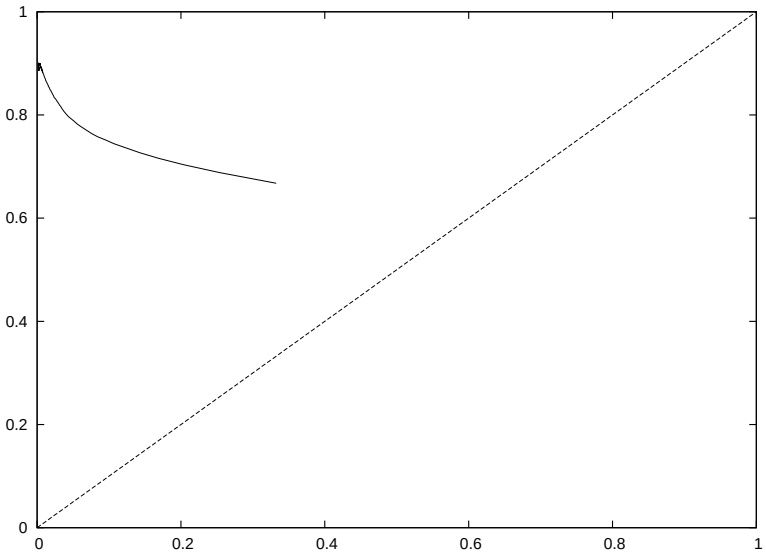
Density for 10^4 executions, after 0,10,20 or 30 iterations. (play a film).

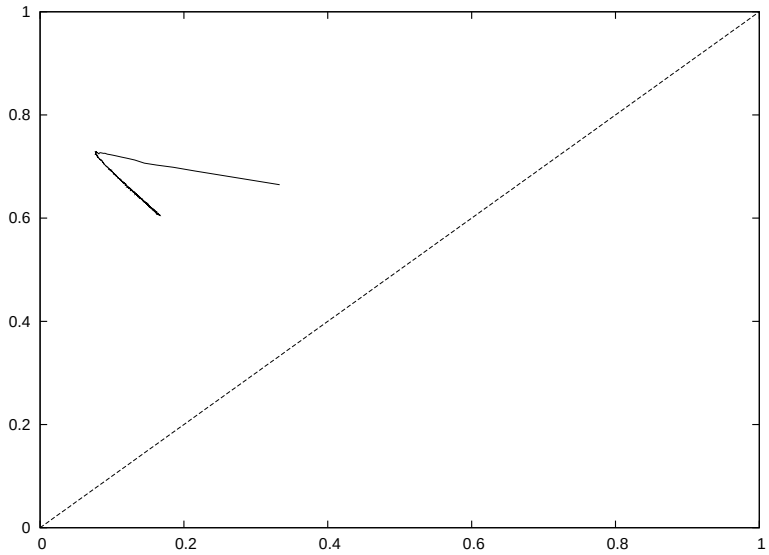
Convergence rates



Density for 10^4 executions, after 0,10,20 or 30 iterations. (play a film).

Average trajectory





Average trajectories do not follow the continuous vector field (esp. when the tempereture is high).

Robustness w.r.t. asynchronous revision and delays

Let us consider asynchronous versions of algorithm ED-Strat.

- Let R_n be the random set of players that update their choice at time n,
- $\delta_{ij}(n)$ be the delay faced by player i in receiving outputs from j at step n.

If $(R_n)_{n\in\mathbb{N}}$ is an irreducible Markov chain and $\delta_{i,j}(n)$ are bounded a.s. then, the modified algorithm is a stochastic approximation of a modified differential equation $\dot{x} = \lambda(t)ED(x)$, where $\lambda_k(t)$ is the rate at which player k updates its strategy.
The modified ED has the same stable rest points as the original one.

Robustness w.r.t. noisy payoffs

Assume that the payoffs g_k are only known up to some noise $\xi_k(\alpha)$, that forms a difference of maringales.
If $\xi_k(\alpha)$ is bounded a.s., and independent of α_k, the action chosen by player k, then the theorem holds when the current payoff $g_k(\hat{\alpha})$ is replaced, by $g_k(\hat{\alpha}) + \xi_k(\hat{\alpha})$.

The noise can depend on the entire history of the game. Also, the independence hypothesis is true when the randomness comes from uncertainty on the actions taken by the other players.

Revised algorithm ED-Strat

ED-Strat for player k

Repeat

At local time $t_k(n)$, Pick action $\hat{\alpha}_k$ according to the mixed strategy X_k

Measure (or compute) $\hat{g} \leftarrow g_k(\hat{\alpha})$

For Each action $\alpha \in \mathcal{A}_k$

$$X_\alpha \leftarrow X_\alpha + \varepsilon_{n+1}\left[-TX_\alpha\left(\log X_\alpha - \sum\nolimits_\beta X_\beta \log X_\beta\right) + \hat{g}\left(\mathbf{1}_{\alpha=\hat{\alpha_k}} - X_\alpha\right)\right]$$

Algorithm ED-Strat assessment

- Stateless: yes and yes
- Time-oblivious:yes
- Robust: yes
- Delay-oblivious: yes
- Scalable: yes (no formal proof in general)
- speed of convergence does not decrease too fast with the temperature.

Its only parameter (T) should be easy to tune: A small temperature gives better results but slows the speed of convergence and alters the numerical stability (the scores grow larger and although they do not appear explicitly, they still affect the values of X). The optimal choice of the temperature giving a good compromise between accuracy and speed of convergence, is problem dependent.

Application to Spectrum management in MIMO systems

We consider a MIMO system made of K non-cooperative MIMO transmitters who upload data to a receiver. Each transmitter wants to improve its individual achievable rate g_k by unilaterally changing its signal covariance matrix $\mathbf{Q}_k$, where

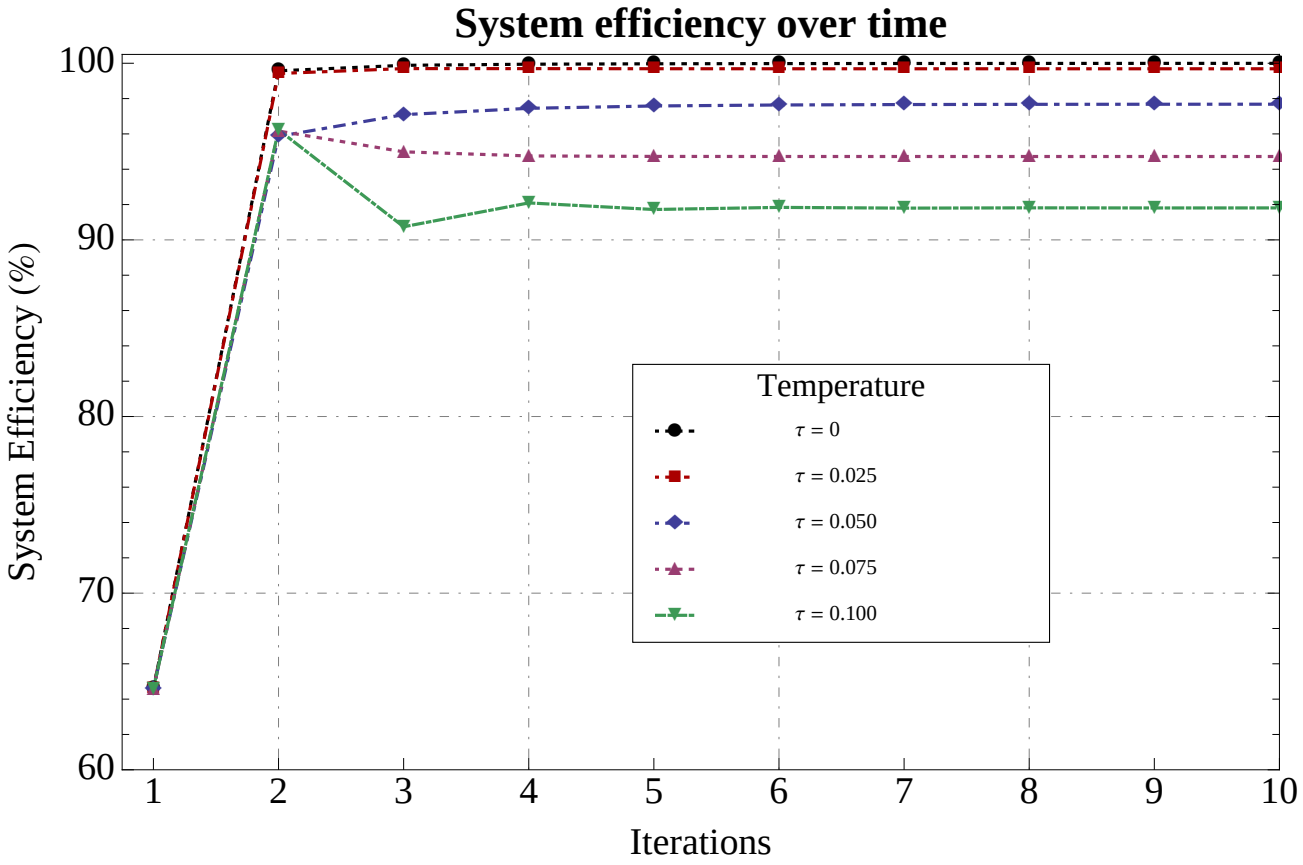
$$g_k(\mathbf{Q}) = \log\det\left(\mathbf{W}_k + \mathbf{H}_k\mathbf{Q}_k\mathbf{H}_k^\dagger\right) - \log\det\left(\mathbf{W}_k\right), \tag{1}$$

where

$$\mathbf{W}_k = \mathbf{I} + \sum_{\ell\neq k}\mathbf{H}_\ell\mathbf{Q}_\ell\mathbf{H}_\ell^\dagger \tag{2}$$

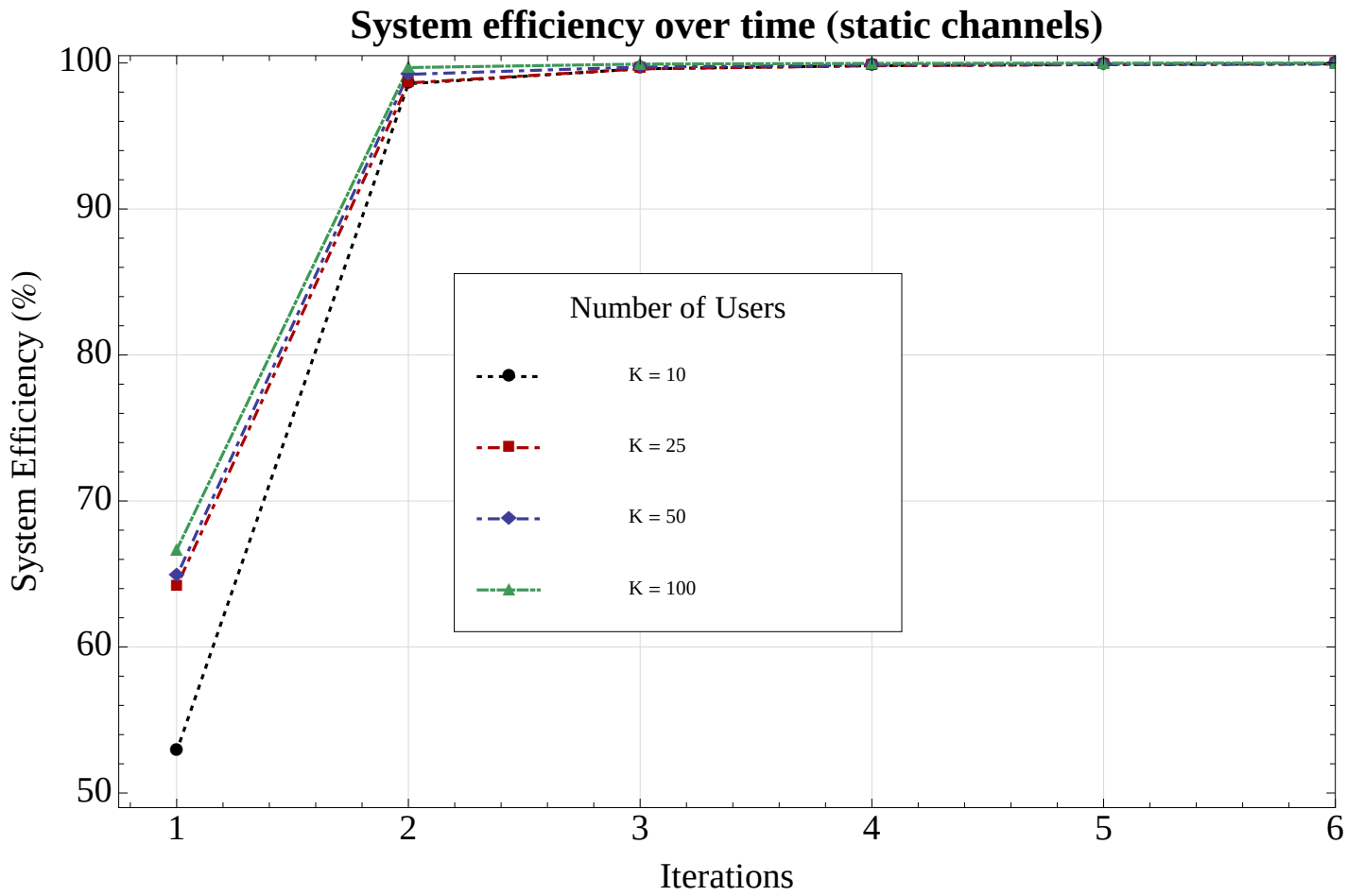
This is a congestion (routing) game, so let us check the performance of our algorithm in this case.

Tuning the temperature



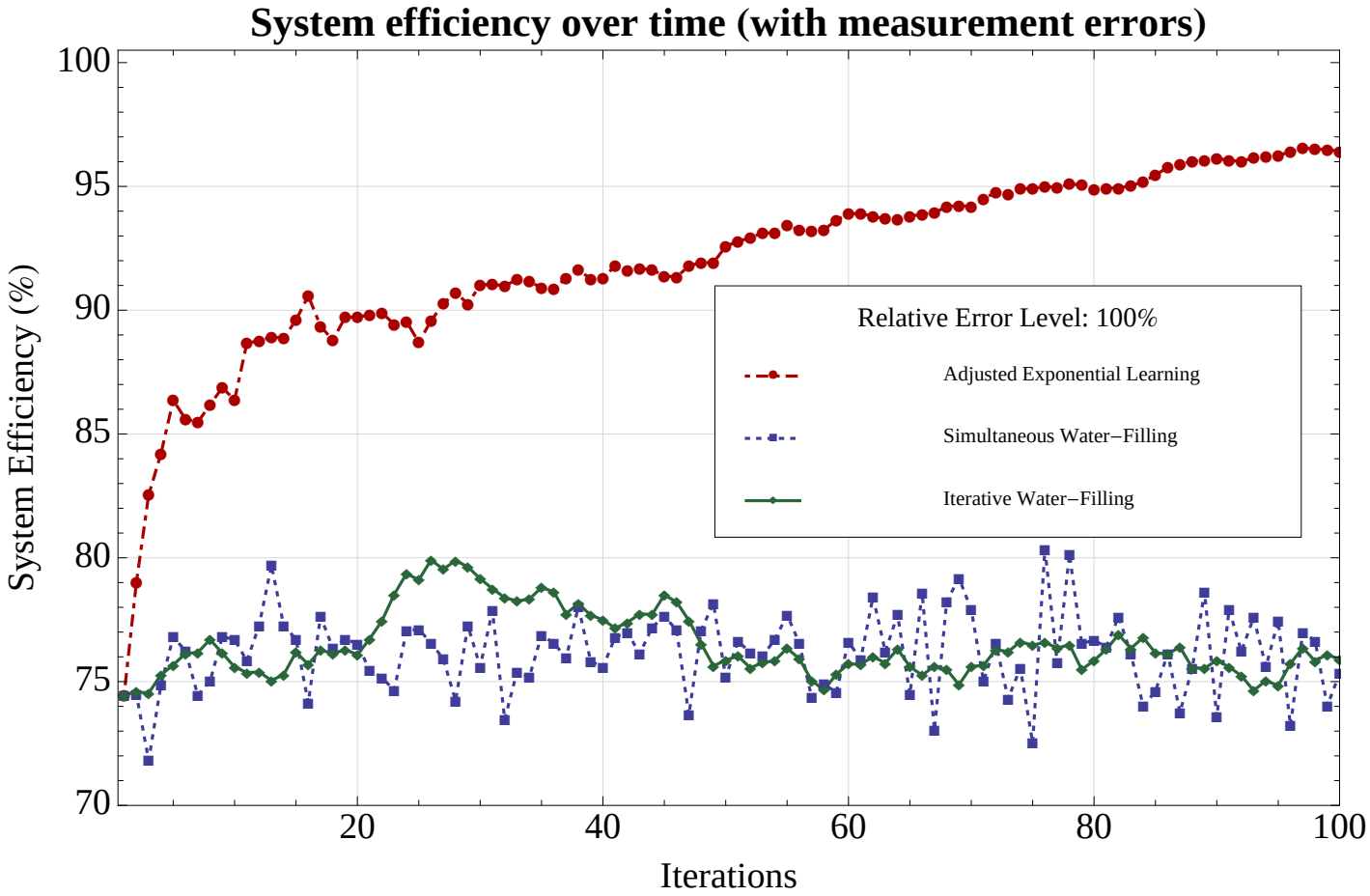
MIMO system consisting of a wireless base receiver with 5 antennas and $K = 25$ transmitters, each with a random number m_k of transmit antennas picked uniformly between 2 and 6.

Scalability



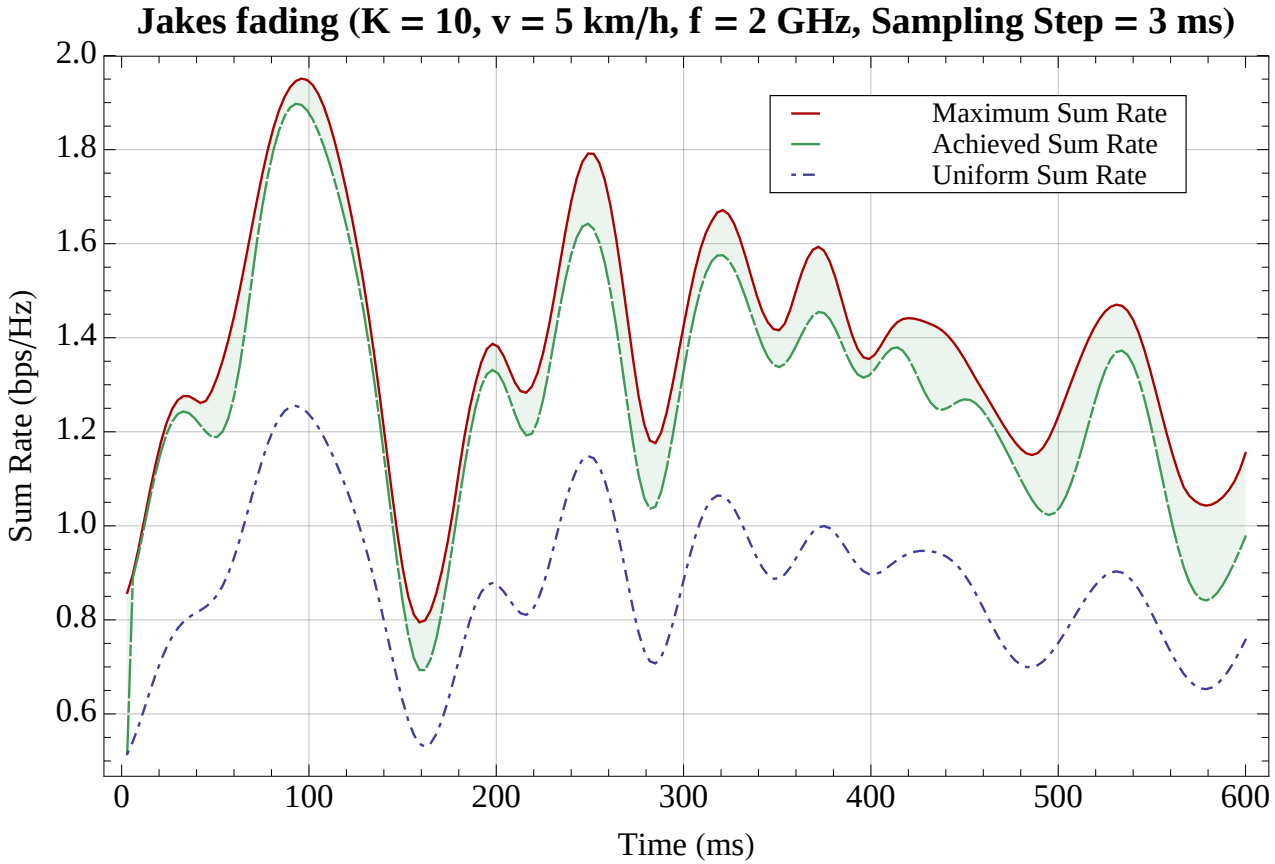
Temperature is fixed to $\tau = 10^{-3}$, the number of users varies: $K = 10$, 25, 50 and 100.

Efficiency with measurement errors



Gaussian noise is added to all measurements. The relative error is 100% (average magnitude of the error is same as the true value). ED-Score is compared with classical techniques (water filling and iterative water filling).

Speed of convergence and tracking



The performance of entropy-driven learning under changing channel conditions (with Jakes fading).

Thank you for your attention.