



# Learning in Nonlinear Games

P. Mertikopoulos

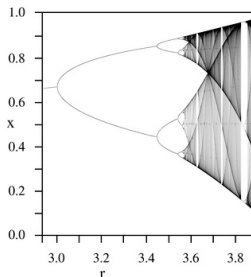
École Polytechnique, Department of Economics

AlgoGT 2011 – June 20, 2011

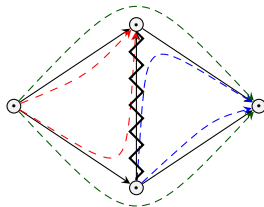
## Motivation

Many game-like interactions have highly nonlinear utilities:

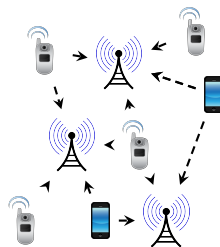
Population Growth Models



Congestion Models



Wireless Networks

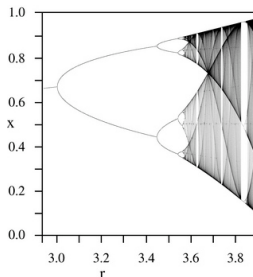




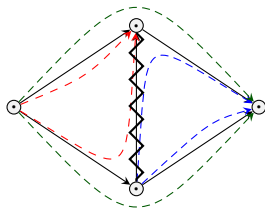
## Motivation

Many game-like interactions have highly nonlinear utilities:

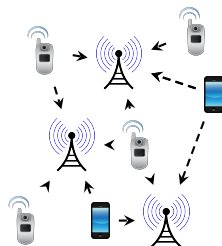
Population Growth Models



Congestion Models



Wireless Networks



- ▶ Learning methods usually work with finite, affine or convex structures.
- ▶ **Can we control completely nonlinear games?**



# Outline

Semilinear Models

Fully Nonlinear Models



## ***Simplicial State Spaces and Nonlinear Utilities I: Routing***

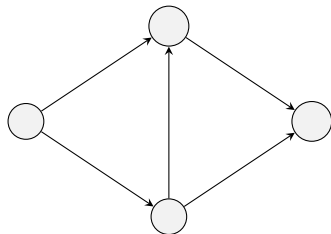
Large-scale (nonatomic) congestion models consist of:



## Simplicial State Spaces and Nonlinear Utilities I: Routing

Large-scale (nonatomic) congestion models consist of:

- ▶ An arbitrary directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ .

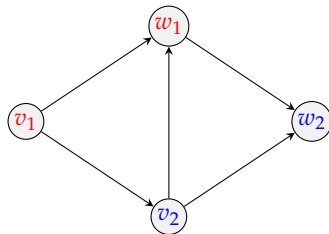




## Simplicial State Spaces and Nonlinear Utilities I: Routing

Large-scale (nonatomic) congestion models consist of:

- ▶ An arbitrary directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ .
- ▶ A collection of **origin-destination pairs**  $\mathcal{K} = \{1, \dots, N\}$  with traffic rates  $\rho_k$ .

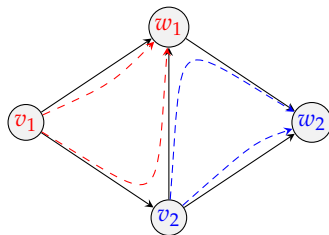




## Simplicial State Spaces and Nonlinear Utilities I: Routing

Large-scale (nonatomic) congestion models consist of:

- ▶ An arbitrary directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ .
- ▶ A collection of **origin-destination pairs**  $\mathcal{K} = \{1, \dots, N\}$  with traffic rates  $\rho_k$ .
- ▶ A collection of **paths**  $\mathcal{A}_k$  joining each pair.





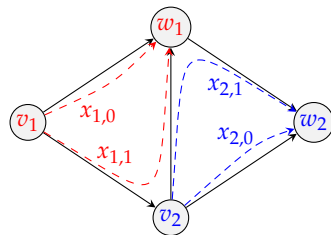


## Simplicial State Spaces and Nonlinear Utilities I: Routing

Large-scale (nonatomic) congestion models consist of:

- ▶ An arbitrary directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ .
- ▶ A collection of **origin-destination pairs**  $\mathcal{K} = \{1, \dots, N\}$  with traffic rates  $\rho_k$ .
- ▶ A collection of **paths**  $\mathcal{A}_k$  joining each pair.
- ▶ The space of each pair's **traffic flows**:

$$\Delta_k = \{x_k \in \mathbb{R}^{\mathcal{A}_k} : x_{k\alpha} \geq 0 \text{ and } \sum_{\alpha} x_{k\alpha} = \rho_k\}.$$





## Simplicial State Spaces and Nonlinear Utilities I: Routing

Large-scale (nonatomic) congestion models consist of:

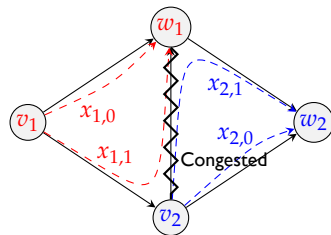
- ▶ An arbitrary directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ .
- ▶ A collection of **origin-destination pairs**  $\mathcal{K} = \{1, \dots, N\}$  with traffic rates  $\rho_k$ .
- ▶ A collection of **paths**  $\mathcal{A}_k$  joining each pair.
- ▶ The space of each pair's **traffic flows**:

$$\Delta_k = \{x_k \in \mathbb{R}^{\mathcal{A}_k} : x_{k\alpha} \geq 0 \text{ and } \sum_{\alpha} x_{k\alpha} = \rho_k\}.$$

- ▶ The network's **latency functions**:

$$\omega_{k\alpha}(x) \equiv \sum_{r \in \alpha} \phi_r(y_r)$$

where  $y_r = \sum_{\alpha \ni r} x_{k\alpha}$  is the **load** on link  $r$  and  $\phi_r$  is the **expected delay**.



$$\begin{aligned} \omega_{1,1}(x) = & \phi_2(x_{1,1}) + \phi_5(x_{1,1} + x_{2,1}) \\ & \dots \end{aligned}$$



## Simplicial State Spaces and Nonlinear Utilities I: Routing

Large-scale (nonatomic) congestion models consist of:

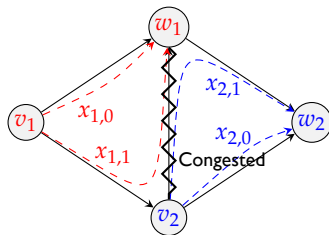
- ▶ An arbitrary directed graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ .
- ▶ A collection of **origin-destination pairs**  $\mathcal{K} = \{1, \dots, N\}$  with traffic rates  $\rho_k$ .
- ▶ A collection of **paths**  $\mathcal{A}_k$  joining each pair.
- ▶ The space of each pair's **traffic flows**:

$$\Delta_k = \{x_k \in \mathbb{R}^{\mathcal{A}_k} : x_{k\alpha} \geq 0 \text{ and } \sum_{\alpha} x_{k\alpha} = \rho_k\}.$$

- ▶ The network's **latency functions**:

$$\omega_{k\alpha}(x) \equiv \sum_{r \in \alpha} \phi_r(y_r)$$

where  $y_r = \sum_{\alpha \ni r} x_{k\alpha}$  is the **load** on link  $r$   
and  $\phi_r$  is the **expected delay**.



$$\begin{aligned} \omega_{1,1}(x) = & \phi_2(x_{1,1}) + \phi_5(x_{1,1} + x_{2,1}) \\ & \dots \end{aligned}$$

**Assumption** O/D latencies  $\omega_k(x) = \sum_{\alpha} x_{k\alpha} \omega_{k\alpha}(x)$  are increasing.



## Routing efficiency notions

Main objective: minimize delays. *But what does this really mean?*



## Routing efficiency notions

Main objective: minimize delays. *But what does this really mean?*

- ▶ **Optimum Distributions** – traffic allocations which minimize the network's aggregate latency  $\Omega(x) = \sum_{k,\alpha} x_{k\alpha} \omega_{k\alpha}(x)$ :

$$\Omega(q) \leq \Omega(x) \text{ for all flow profiles } x \in \Delta.$$



## Routing efficiency notions

Main objective: minimize delays. *But what does this really mean?*

- ▶ **Optimum Distributions** – traffic allocations which minimize the network's aggregate latency  $\Omega(x) = \sum_{k,\alpha} x_{k\alpha} \omega_{k\alpha}(x)$ :

$$\Omega(q) \leq \Omega(x) \text{ for all flow profiles } x \in \Delta.$$

- ▶ **Nash/Wardrop Equilibrium** – there are no profitable *unilateral* deviations:

$$\omega_{k\alpha}(q) \leq \omega_{k\beta}(q) \text{ for all } \alpha, \beta \in \mathcal{A}_k \text{ such that } q_{k\alpha} > 0.$$

"The journey times in all routes actually used are equal and less than those which would be experienced by a single vehicle on any unused route." (J. G. Wardrop, 1952)



## Routing efficiency notions

Main objective: minimize delays. *But what does this really mean?*

- ▶ **Optimum Distributions** – traffic allocations which minimize the network's aggregate latency  $\Omega(x) = \sum_{k,\alpha} x_{k\alpha} \omega_{k\alpha}(x)$ :

$$\Omega(q) \leq \Omega(x) \text{ for all flow profiles } x \in \Delta.$$

- ▶ **Nash/Wardrop Equilibrium** – there are no profitable *unilateral* deviations:

$$\omega_{k\alpha}(q) \leq \omega_{k\beta}(q) \text{ for all } \alpha, \beta \in \mathcal{A}_k \text{ such that } q_{k\alpha} > 0.$$

"The journey times in all routes actually used are equal and less than those which would be experienced by a single vehicle on any unused route." (J. G. Wardrop, 1952)

These two notions are dual to each other:

|         | Minimize                           | Equilibrate                      |
|---------|------------------------------------|----------------------------------|
| Optimum | $\sum_r y_r \phi_r(y_r)$           | $\phi_r(y_r) + y_r \phi_r'(y_r)$ |
| Wardrop | $\sum_r \int_0^{y_r} \phi_r(w) dw$ | $\phi_r(y_r)$                    |



## ***Semilinear Games II: Power allocation***

For a different example, consider the following wireless network model:





## Semilinear Games II: Power allocation

For a different example, consider the following wireless network model:

- ▶ A set of receivers  $\mathcal{A} = \{1, \dots, A\}$  with non-overlapping channels.





## Semilinear Games II: Power allocation

For a different example, consider the following wireless network model:

- ▶ A set of receivers  $\mathcal{A} = \{1, \dots, A\}$  with non-overlapping channels.
- ▶ A set of users  $\mathcal{K} = \{1, \dots, K\}$  who transmit over multiple bands.



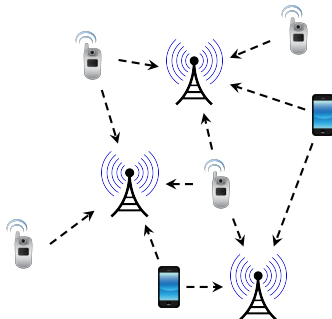


## Semilinear Games II: Power allocation

For a different example, consider the following wireless network model:

- ▶ A set of receivers  $\mathcal{A} = \{1, \dots, A\}$  with non-overlapping channels.
- ▶ A set of users  $\mathcal{K} = \{1, \dots, K\}$  who transmit over multiple bands.
- ▶ Users split power among nodes:

$$p_{k\alpha} \geq 0, \quad \sum_{\alpha}^k p_{k\alpha} \leq P_k.$$





## Semilinear Games II: Power allocation

For a different example, consider the following wireless network model:

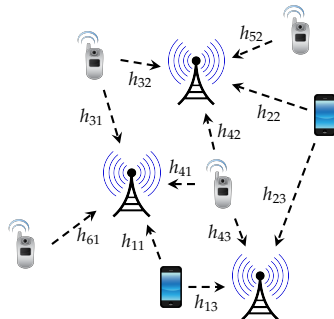
- ▶ A set of receivers  $\mathcal{A} = \{1, \dots, A\}$  with non-overlapping channels.
- ▶ A set of users  $\mathcal{K} = \{1, \dots, K\}$  who transmit over multiple bands.
- ▶ Users split power among nodes:

$$p_{k\alpha} \geq 0, \quad \sum_{\alpha}^k p_{k\alpha} \leq P_k.$$

- ▶ A popular performance metric is the users' **spectral efficiency**:

$$u_k(p) = \sum_{\alpha \in \mathcal{A}} \log(1 + \text{snr}_{k\alpha}(p)) = \sum_{\alpha \in \mathcal{A}} \log \left( 1 + \frac{g_{k\alpha} p_{k\alpha}}{\sigma_{\alpha}^2 + \sum_{\ell \neq k} g_{\ell\alpha} p_{\ell\alpha}} \right)$$

where  $g_{k\alpha} = |h_{k\alpha}|^2$  are the channel coefficients and  $\sigma_{\alpha}^2$  is the noise.





## ***Bird's Eye View of Semilinear Models***

Similarities between these two games:

- ▶ The state space is a product of simplices  $\Delta = \prod_k \Delta_k$ .
- ▶ The users' utilities are nonlinear over  $\Delta$ .
- ▶ Both games admit a potential function.



## Bird's Eye View of Semilinear Models

Similarities between these two games:

- ▶ The state space is a product of simplices  $\Delta = \prod_k \Delta_k$ .
- ▶ The users' utilities are nonlinear over  $\Delta$ .
- ▶ Both games admit a potential function.

Differences:

- ▶ The games are potential in different ways (Sandholm v. Monderer-Shapley)
- ▶ Different equilibrial structure (but both at the minimum of the potential)
- ▶ State-specific v. undecomposable utilities:
  - ▶  $u_k(x) = \sum_{\alpha} x_{k\alpha} u_{k\alpha}(x)$  in the routing model
  - ▶ no such meaningful decomposition in the wireless network model



## Complexity of Calculating Efficient States

Unfortunately, efficient states are not very easy to calculate:

- ▶ Nonlinear convex problem with exponentially many variables.
- ▶ Proposed algorithms (e.g. flow deviation or water-filling) have important drawbacks:
  - ▶ Computationally intensive (e.g. solve MFDL at each step)
  - ▶ Not wholly distributed (must perform a global linear search)
  - ▶ Reduced convergence rates: search directions quickly become suboptimal



## Complexity of Calculating Efficient States

Unfortunately, efficient states are not very easy to calculate:

- ▶ Nonlinear convex problem with exponentially many variables.
- ▶ Proposed algorithms (e.g. flow deviation or water-filling) have important drawbacks:
  - ▶ Computationally intensive (e.g. solve MFDL at each step)
  - ▶ Not wholly distributed (must perform a global linear search)
  - ▶ Reduced convergence rates: search directions quickly become suboptimal

It is thus natural to apply ideas from **distributed learning in games**:

- ▶ Agents only possess and process a limited amount of information.
- ▶ No exogenous forcing needed: evolution is steered by individual objectives.





## Complexity of Calculating Efficient States

Unfortunately, efficient states are not very easy to calculate:

- ▶ Nonlinear convex problem with exponentially many variables.
- ▶ Proposed algorithms (e.g. flow deviation or water-filling) have important drawbacks:
  - ▶ Computationally intensive (e.g. solve MFDL at each step)
  - ▶ Not wholly distributed (must perform a global linear search)
  - ▶ Reduced convergence rates: search directions quickly become suboptimal

It is thus natural to apply ideas from **distributed learning in games**:

- ▶ Agents only possess and process a limited amount of information.
- ▶ No exogenous forcing needed: evolution is steered by individual objectives.

**The catch:** the learning process must actually work!



## **Exponential Learning**

A scheme which fits the bill particularly well is that of "exponential learning":



## Exponential Learning

A scheme which fits the bill particularly well is that of "exponential learning":

1. Take the agents' marginal utilities:

$$v_{k\alpha}(x) = -\frac{\partial\Phi}{\partial x_{k\alpha}}$$

(which coincide for both decomposable and state-specific utilities)



## Exponential Learning

A scheme which fits the bill particularly well is that of "exponential learning":

1. Take the agents' marginal utilities:

$$v_{k\alpha}(x) = -\frac{\partial\Phi}{\partial x_{k\alpha}}$$

(which coincide for both decomposable and state-specific utilities)

2. Agents record their choices' performance:

$$V_{k\alpha}(t + dt) = V_{k\alpha}(t) + v_{k\alpha}(x(t)) dt$$



## Exponential Learning

A scheme which fits the bill particularly well is that of "exponential learning":

1. Take the agents' marginal utilities:

$$v_{k\alpha}(x) = -\frac{\partial\Phi}{\partial x_{k\alpha}}$$

(which coincide for both decomposable and state-specific utilities)

2. Agents record their choices' performance:

$$V_{k\alpha}(t + dt) = V_{k\alpha}(t) + v_{k\alpha}(x(t)) dt$$

3. State variables are updated according to the Gibbs distribution:

$$x_{k\alpha}(t) = \frac{e^{\lambda_k V_{k\alpha}(t)}}{\sum_{\beta} e^{\lambda_k V_{k\beta}(t)'}}$$

where the "learning parameters"  $\lambda_k$  are player-specific inverse temperatures



## Exponential Learning

A scheme which fits the bill particularly well is that of "exponential learning":

1. Take the agents' marginal utilities:

$$v_{k\alpha}(x) = -\frac{\partial\Phi}{\partial x_{k\alpha}}$$

(which coincide for both decomposable and state-specific utilities)

2. Agents record their choices' performance:

$$V_{k\alpha}(t + dt) = V_{k\alpha}(t) + v_{k\alpha}(x(t)) dt$$

3. State variables are updated according to the Gibbs distribution:

$$x_{k\alpha}(t) = \frac{e^{\lambda_k V_{k\alpha}(t)}}{\sum_{\beta} e^{\lambda_k V_{k\beta}(t)'}}$$

where the "learning parameters"  $\lambda_k$  are player-specific inverse temperatures

4. Decouple to obtain the (rate-adjusted) **replicator dynamics**:

$$\frac{dx_{k\alpha}}{dt} = \lambda_k x_{k\alpha} \left( v_{k\alpha}(x) - \sum_{\beta}^k x_{k\beta} v_{k\beta}(x) \right)$$



## ***Convergence Properties of the Replicator Dynamics***

So, how well does replicator learning fare?



## Convergence Properties of the Replicator Dynamics

So, how well does replicator learning fare?

### Theorem (Standard stuff)

*The minimum sets of the potential are Lyapunov stable (and usually attracting).* [Sandholm '01]





## Convergence Properties of the Replicator Dynamics

So, how well does replicator learning fare?

### Theorem (Standard stuff)

*The minimum sets of the potential are Lyapunov stable (and usually attracting).* [Sandholm '01]

### Theorem (A bit better)

*In stable games, interior solution trajectories always converge to a minimum **point**.* [M '10]



## Convergence Properties of the Replicator Dynamics

So, how well does replicator learning fare?

### Theorem (Standard stuff)

*The minimum sets of the potential are Lyapunov stable (and usually attracting).* [Sandholm '01]

### Theorem (A bit better)

*In stable games, interior solution trajectories always converge to a minimum **point**.* [M '10]

And for a more practical question: how **fast** is this convergence?



## Convergence Properties of the Replicator Dynamics

So, how well does replicator learning fare?

### Theorem (Standard stuff)

The minimum sets of the potential are Lyapunov stable (and usually attracting). [Sandholm '01]

### Theorem (A bit better)

In stable games, interior solution trajectories always converge to a minimum **point**. [M '10]

And for a more practical question: how **fast** is this convergence?

### Theorem (M 2010)

Let  $x(t)$  be an interior solution orbit, and let  $q = \lim_{t \rightarrow \infty} x(t)$ . Then:

$$D_{KL}(q \parallel x(t)) \leq h_0 e^{-ct},$$

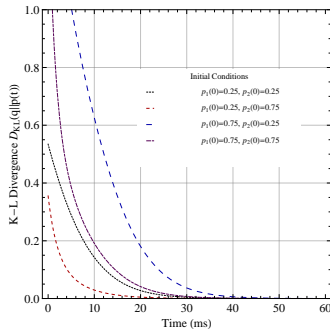
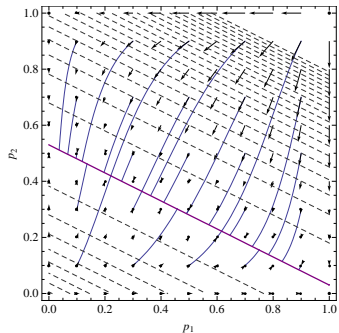
where  $D_{KL}$  denotes the Kullback-Leibler divergence,  $h_0 = D_{KL}(q \parallel x(0))$  is the initial relative entropy, and  $c$  is a positive constant.

In other words, exponential learning equilibrates exponentially quickly:

*solution orbits hit an  $\varepsilon$ -neighborhood of an optimum distribution in time  $\mathcal{O}(\log(1/\varepsilon))$ .*

## Convergence of the Replicator Dynamics

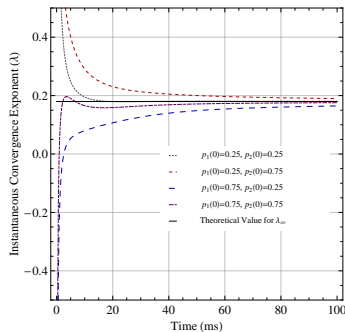
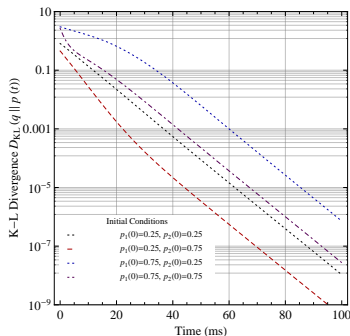
Thus, almost all initial conditions converge to an efficient state





## Convergence of the Replicator Dynamics

Thus, almost all initial conditions converge to an efficient state in time which is at most  $\mathcal{O}(\log(1/\varepsilon))$ .





## Outline

Semilinear Models

Fully Nonlinear Models



## ***Beyond Simplicies: an example***

A natural extension of our wireless network example: multi-antenna systems.



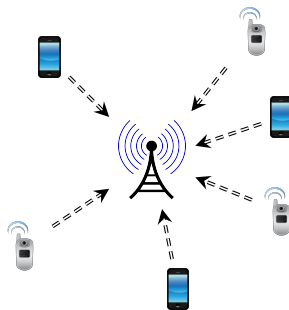
## Beyond Simplicies: an example

A natural extension of our wireless network example: multi-antenna systems.

- ▶ The received signal is:

$$\mathbf{y} = \sum_k \mathbf{H}_k \cdot \mathbf{x}_k + \mathbf{z}$$

where  $\mathbf{x}_k$  is  $k$ 's transmitted vector







## Beyond Simplicies: an example

A natural extension of our wireless network example: multi-antenna systems.

- ▶ The received signal is:

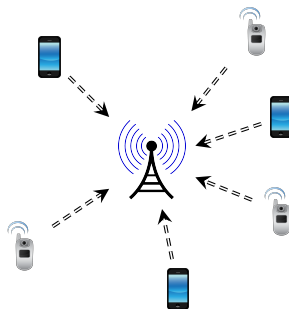
$$\mathbf{y} = \sum_k \mathbf{H}_k \cdot \mathbf{x}_k + \mathbf{z}$$

where  $\mathbf{x}_k$  is  $k$ 's transmitted vector

- ▶ Users choose a **covariance matrix**

$$\mathbf{Q}_k = \mathbf{E} [\mathbf{x}_k \cdot \mathbf{x}_k^\dagger]$$

(single antenna  $\implies$  diagonal  $\mathbf{Q}$ )





## Beyond Simplicies: an example

A natural extension of our wireless network example: multi-antenna systems.

- ▶ The received signal is:

$$\mathbf{y} = \sum_k \mathbf{H}_k \cdot \mathbf{x}_k + \mathbf{z}$$

where  $\mathbf{x}_k$  is  $k$ 's transmitted vector

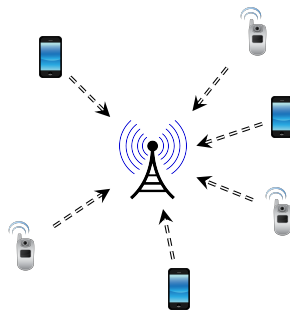
- ▶ Users choose a **covariance matrix**

$$\mathbf{Q}_k = \mathbf{E} [\mathbf{x}_k \cdot \mathbf{x}_k^\dagger]$$

(single antenna  $\implies$  diagonal  $\mathbf{Q}$ )

- ▶ Strategy spaces are now **cones**:

$$M_k = \{\mathbf{Q}_k : \mathbf{Q}_k \succeq 0, \text{tr}(\mathbf{Q}_k) \leq P_k\}$$





## Beyond Simplicies: an example

A natural extension of our wireless network example: multi-antenna systems.

- ▶ The received signal is:

$$\mathbf{y} = \sum_k \mathbf{H}_k \cdot \mathbf{x}_k + \mathbf{z}$$

where  $\mathbf{x}_k$  is  $k$ 's transmitted vector

- ▶ Users choose a **covariance matrix**

$$\mathbf{Q}_k = \mathbf{E} [\mathbf{x}_k \cdot \mathbf{x}_k^\dagger]$$

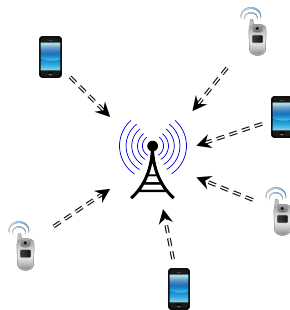
(single antenna  $\implies$  diagonal  $\mathbf{Q}$ )

- ▶ Strategy spaces are now **cones**:

$$M_k = \{ \mathbf{Q}_k : \mathbf{Q}_k \succeq 0, \text{tr}(\mathbf{Q}_k) \leq P_k \}$$

- ▶ Utility functions are undecomposable:

$$u_k(\mathbf{Q}) = \log \det \left( \mathbf{I} + \sum_k \mathbf{H}_k \mathbf{Q}_k \mathbf{H}_k^\dagger \right) - \log \det \left( \mathbf{I} + \sum_{\ell \neq k} \mathbf{H}_\ell \mathbf{Q}_\ell \mathbf{H}_\ell^\dagger \right)$$



How can we control this totally nonlinear (but still relevant) system?



## ***Beyond Simplices: an example***

The challenge: **non-simplicial state space**  $\implies$  **doom for replicator dynamics** 😞



## Beyond Simplicies: an example

The challenge: **non-simplicial state space**  $\implies$  **doom for replicator dynamics** 😞

Vanilla game-theoretic solution:

1. Discretize the state space.
2. Take (multi)linearized versions of the game's utilities.
3. Employ your favorite discrete learning algorithm.
4. Converge to a state which is efficient ``on average''.

Downside: efficiency  $\implies$  fine discretization  $\implies$  high complexity



## Beyond Simplicies: an example

The challenge: **non-simplicial state space**  $\implies$  **doom for replicator dynamics** 😞

Vanilla game-theoretic solution:

1. Discretize the state space.
2. Take (multi)linearized versions of the game's utilities.
3. Employ your favorite discrete learning algorithm.
4. Converge to a state which is efficient "on average".

Downside: efficiency  $\implies$  fine discretization  $\implies$  high complexity

"Distributed" optimization workaround: water-filling and its friends

- + Converges to equilibrium in many scenarios.
- High complexity.
- Not entirely distributed.



## ***Beyond Simplices: an example***

Still, can we salvage replicator-type approaches?

- ▶ Can we write a dynamical system which respects the game's state space?
- ▶ Can we obtain good convergence properties?



## Beyond Simplicies: an example

Still, can we salvage replicator-type approaches?

- ▶ Can we write a dynamical system which respects the game's state space?
- ▶ Can we obtain good convergence properties?

**Yes, we can!**

$$\frac{d\mathbf{Q}_k}{dt} = \mathbf{Q}_k \cdot \left( \mathbf{V}_k - \frac{\text{tr}(\mathbf{Q}_k \mathbf{V}_k)}{\text{tr}(\mathbf{Q}_k)} \mathbf{I} \right)$$

where  $\mathbf{V}_k = -\frac{\partial \Phi}{\partial \mathbf{Q}_k}$  is the matrix derivative of the game's potential w.r.t.  $\mathbf{Q}_k$ .

(Diagonal  $\mathbf{Q}_k$ 's reduce to the standard replicator dynamics ✓)





## Beyond Simplices: an example

Still, can we salvage replicator-type approaches?

- ▶ Can we write a dynamical system which respects the game's state space?
- ▶ Can we obtain good convergence properties?

**Yes, we can!**

$$\frac{d\mathbf{Q}_k}{dt} = \mathbf{Q}_k \cdot \left( \mathbf{V}_k - \frac{\text{tr}(\mathbf{Q}_k \mathbf{V}_k)}{\text{tr}(\mathbf{Q}_k)} \mathbf{I} \right)$$

where  $\mathbf{V}_k = -\frac{\partial \Phi}{\partial \mathbf{Q}_k}$  is the matrix derivative of the game's potential w.r.t.  $\mathbf{Q}_k$ .

(Diagonal  $\mathbf{Q}_k$ 's reduce to the standard replicator dynamics ✓)

**Theorem (M-Moustakas '11)**

1. *The matrix replicator dynamics leave the game's state space invariant.*
2. *Solution trajectories converge to equilibrium.*



## Nonlinear Learning — Non-convex Optimization

General optimization problem formulation:

- ▶ Let  $M$  be a smooth manifold with boundary embedded in  $\mathbb{R}^n$ :

$$M = \{x \in \mathbb{R}^n : f_k(x) = 0, g_r(x) \geq 0, k = 1, \dots, K, r = 1, \dots, R\}$$

- ▶ Let  $\phi: \mathbb{R}^n \rightarrow \mathbb{R}$  be a smooth function.



## Nonlinear Learning — Non-convex Optimization

General optimization problem formulation:

- ▶ Let  $M$  be a smooth manifold with boundary embedded in  $\mathbb{R}^n$ :

$$M = \{x \in \mathbb{R}^n : f_k(x) = 0, g_r(x) \geq 0, k = 1, \dots, K, r = 1, \dots, R\}$$

- ▶ Let  $\phi: \mathbb{R}^n \rightarrow \mathbb{R}$  be a smooth function.

**The challenge:** find a smooth vector field  $V \equiv V(\phi)$  on  $M$  such that the dynamical system  $\dot{x} = -V(x)$  converges to the minimum points of  $\phi$ .



## Nonlinear Learning — Non-convex Optimization

General optimization problem formulation:

- ▶ Let  $M$  be a smooth manifold with boundary embedded in  $\mathbb{R}^n$ :

$$M = \{x \in \mathbb{R}^n : f_k(x) = 0, g_r(x) \geq 0, k = 1, \dots, K, r = 1, \dots, R\}$$

- ▶ Let  $\phi: \mathbb{R}^n \rightarrow \mathbb{R}$  be a smooth function.

**The challenge:** find a smooth vector field  $V \equiv V(\phi)$  on  $M$  such that the dynamical system  $\dot{x} = -V(x)$  converges to the minimum points of  $\phi$ .

Easy if  $M$  is boundaryless ( $V = \text{grad}(\phi|_M)$  works!), but what happens if  $M$  **does** have a boundary (viz. power/rate constraints of previous examples)?



## The General Idea

Guidelines and desiderata:

- ▶ Project  $\text{grad}(\phi)$  on  $M$ .
- ▶ "Temper"  $\text{grad}(\phi)$  to avoid collisions with the boundary.
- ▶ Recover standard dynamics (think replicator) in simple cases.



## A First Implementation

A promising approach is the following:

1. Let  $w_k = \text{grad}(f_k)$  be the "outward normals" of  $M$ .



## A First Implementation

A promising approach is the following:

1. Let  $w_k = \text{grad}(f_k)$  be the "outward normals" of  $M$ .
2. Take the "normal projection" of the objective  $\phi$ :

$$\text{grad}_{\perp}(\phi) \equiv \text{proj}_W \text{grad}(\phi) = \sum_k \frac{\langle w_k, \text{grad}(\phi) \rangle}{\|w_k\|^2} w_k$$



## A First Implementation

A promising approach is the following:

1. Let  $w_k = \text{grad}(f_k)$  be the "outward normals" of  $M$ .
2. Take the "normal projection" of the objective  $\phi$ :

$$\text{grad}_{\perp}(\phi) \equiv \text{proj}_W \text{grad}(\phi) = \sum_k \frac{\langle w_k, \text{grad}(\phi) \rangle}{\|w_k\|^2} w_k$$

3. Let  $\text{grad}_M(\phi) = \text{grad}(\phi) - \text{grad}_{\perp}(\phi)$  be the projection of  $\text{grad}(\phi)$  on  $M$ .





## A First Implementation

A promising approach is the following:

1. Let  $w_k = \text{grad}(f_k)$  be the "outward normals" of  $M$ .
2. Take the "normal projection" of the objective  $\phi$ :

$$\text{grad}_{\perp}(\phi) \equiv \text{proj}_W \text{grad}(\phi) = \sum_k \frac{\langle w_k, \text{grad}(\phi) \rangle}{\|w_k\|^2} w_k$$

3. Let  $\text{grad}_M(\phi) = \text{grad}(\phi) - \text{grad}_{\perp}(\phi)$  be the projection of  $\text{grad}(\phi)$  on  $M$ .
4. Temper  $\tilde{V} \equiv \text{grad}_M(\phi)$  by setting  $V_{\alpha} = g_{\alpha} \tilde{V}_{\alpha}$  where

$$g_{\alpha}(x) = \prod_{r \in r} g_r(x)$$

and " $\alpha \in r$ " here means that  $\frac{\partial g_r}{\partial x_{\alpha}} \neq 0$ .



## A First Implementation

A promising approach is the following:

1. Let  $w_k = \text{grad}(f_k)$  be the "outward normals" of  $M$ .
2. Take the "normal projection" of the objective  $\phi$ :

$$\text{grad}_{\perp}(\phi) \equiv \text{proj}_W \text{grad}(\phi) = \sum_k \frac{\langle w_k, \text{grad}(\phi) \rangle}{\|w_k\|^2} w_k$$

3. Let  $\text{grad}_M(\phi) = \text{grad}(\phi) - \text{grad}_{\perp}(\phi)$  be the projection of  $\text{grad}(\phi)$  on  $M$ .
4. Temper  $\tilde{V} \equiv \text{grad}_M(\phi)$  by setting  $V_{\alpha} = g_{\alpha} \tilde{V}_{\alpha}$  where

$$g_{\alpha}(x) = \prod_{r \in r} g_r(x)$$

and " $\alpha \in r$ " here means that  $\frac{\partial g_r}{\partial x_{\alpha}} \neq 0$ .

5. Learn by following the dynamics  $\dot{x}_{\alpha} = -V_{\alpha}(x) = -g_{\alpha}(x) \text{grad}_{\alpha}(\phi)$ .



## A Constructive Failure

This approach fails (but constructively so) even in the simplex case.

- ▶  $\phi: \mathbb{R}^n \rightarrow \mathbb{R}$  is a function to be minimized over the unit simplex.
- ▶  $f(x) \equiv \sum_{\alpha} x_{\alpha} - 1 = 0$  defines the boundaryless manifold.
- ▶  $g_{\alpha}(x) = x_{\alpha} \geq 0, \alpha = 1, \dots, n$ , defines the boundary.



## A Constructive Failure

This approach fails (but constructively so) even in the simplex case.

- ▶  $\phi: \mathbb{R}^n \rightarrow \mathbb{R}$  is a function to be minimized over the unit simplex.
- ▶  $f(x) \equiv \sum_{\alpha} x_{\alpha} - 1 = 0$  defines the boundaryless manifold.
- ▶  $g_{\alpha}(x) = x_{\alpha} \geq 0, \alpha = 1, \dots, n$ , defines the boundary.

The outlined approach then gives:

$$\dot{x}_{\alpha} = -x_{\alpha} \left( \frac{\partial \phi}{\partial x_{\alpha}} - \frac{1}{n} \sum_{\beta} \frac{\partial \phi}{\partial x_{\beta}} \right)$$

which fails to remain in the unit simplex ( $\sum_{\alpha} \dot{x}_{\alpha} \neq 0$ ).

**Reason: the tempering meddles with the projection phase and vice versa.**



## Overcoming the Collision vs. Projection Conflict

The problem is subtle (but profound!): what does "project" mean?



## Overcoming the Collision vs. Projection Conflict

The problem is subtle (but profound!): what does "project" mean?

- ▶ The Euclidean metric is not natural!



## Overcoming the Collision vs. Projection Conflict

The problem is subtle (but profound!): what does "project" mean?

- ▶ The Euclidean metric is not natural!
- ▶ Instead, consider the "complementary slackness" metric  $g$ :

$$\langle u, v \rangle = \sum_{\alpha} u_{\alpha} g_{\alpha} v_{\alpha}$$

where  $g_{\alpha} = \prod_{\alpha \in r} g_r$  are the complementary slackness scale factors.



## Overcoming the Collision vs. Projection Conflict

The problem is subtle (but profound!): what does "project" mean?

- ▶ The Euclidean metric is not natural!
- ▶ Instead, consider the "complementary slackness" metric  $g$ :

$$\langle u, v \rangle = \sum_{\alpha} u_{\alpha} g_{\alpha} v_{\alpha}$$

where  $g_{\alpha} = \prod_{\alpha \in r} g_r$  are the complementary slackness scale factors.

- ▶ We then obtain the **generalized replicator dynamics**:

$$\dot{x}_{\alpha} = -\text{proj}_{\alpha}(\text{grad}_M(\phi))$$

where the projection on the  $\alpha$ -th coordinate is taken w.r.t.  $g$  and  $\text{grad}_M(\phi) = \text{grad}(\phi) - \text{grad}_{\perp}(\phi)$  is the gradient of  $\phi$  **with respect to  $g$** .





## Overcoming the Collision vs. Projection Conflict

The problem is subtle (but profound!): what does "project" mean?

- ▶ The Euclidean metric is not natural!
- ▶ Instead, consider the "complementary slackness" metric  $g$ :

$$\langle u, v \rangle = \sum_{\alpha} u_{\alpha} g_{\alpha} v_{\alpha}$$

where  $g_{\alpha} = \prod_{r \in R} g_r$  are the complementary slackness scale factors.

- ▶ We then obtain the **generalized replicator dynamics**:

$$\dot{x}_{\alpha} = -\text{proj}_{\alpha}(\text{grad}_M(\phi))$$

where the projection on the  $\alpha$ -th coordinate is taken w.r.t.  $g$  and  $\text{grad}_M(\phi) = \text{grad}(\phi) - \text{grad}_{\perp}(\phi)$  is the gradient of  $\phi$  **with respect to  $g$** .

### Theorem (M 2011)

If the embedding of  $M$  has no singular points, then:

- ▶ The generalized replicator dynamics are always tangent to  $M$ .
- ▶ The objective function  $\phi$  is (weakly) decreasing along solution orbits.



## Typical Results for the Generalized Replicator Dynamics

Hard part is over! A simple product allows us to **learn in nonlinear games**:

$$\dot{x}_k = -\text{proj}_{M_k} \left( \text{grad}_{M_k} (u_k) \right),$$

where  $M_k$  is the strategy manifold of agent  $k$  and  $u_k: \prod_k M_k \rightarrow \mathbb{R}$  his utility.



## Typical Results for the Generalized Replicator Dynamics

Hard part is over! A simple product allows us to **learn in nonlinear games**:

$$\dot{x}_k = -\text{proj}_{M_k} \left( \text{grad}_{M_k} (u_k) \right),$$

where  $M_k$  is the strategy manifold of agent  $k$  and  $u_k: \prod_k M_k \rightarrow \mathbb{R}$  his utility.

### Theorem

Let  $\mathfrak{G} = (\mathcal{X}, M_k, u_k)$  be a nonlinear game. Then:

- ▶ *The stationary points of the generalized replicator dynamics are restricted equilibria (when they exist).*



## Typical Results for the Generalized Replicator Dynamics

Hard part is over! A simple product allows us to **learn in nonlinear games**:

$$\dot{x}_k = -\text{proj}_{M_k} \left( \text{grad}_{M_k} (u_k) \right),$$

where  $M_k$  is the strategy manifold of agent  $k$  and  $u_k: \prod_k M_k \rightarrow \mathbb{R}$  his utility.

### Theorem

Let  $\mathfrak{G} = (\mathcal{K}, M_k, u_k)$  be a nonlinear game. Then:

- ▶ The stationary points of the generalized replicator dynamics are restricted equilibria (when they exist).
- ▶ If  $x(t) \rightarrow q$ , then  $x^*$  is a restricted equilibrium.



## Typical Results for the Generalized Replicator Dynamics

Hard part is over! A simple product allows us to **learn in nonlinear games**:

$$\dot{x}_k = -\text{proj}_{M_k} \left( \text{grad}_{M_k} (u_k) \right),$$

where  $M_k$  is the strategy manifold of agent  $k$  and  $u_k: \prod_k M_k \rightarrow \mathbb{R}$  his utility.

### Theorem

Let  $\mathfrak{G} = (\mathcal{X}, M_k, u_k)$  be a nonlinear game. Then:

- ▶ *The stationary points of the generalized replicator dynamics are restricted equilibria (when they exist).*
- ▶ *If  $x(t) \rightarrow q$ , then  $x^*$  is a restricted equilibrium.*
- ▶ *If, in addition, the game admits a potential function  $\phi$ , then all generic minima of  $\phi$  (those that satisfy the strict KKT conditions) are asymptotically stable.*



## Typical Results for the Generalized Replicator Dynamics

Hard part is over! A simple product allows us to **learn in nonlinear games**:

$$\dot{x}_k = -\text{proj}_{M_k} \left( \text{grad}_{M_k} (u_k) \right),$$

where  $M_k$  is the strategy manifold of agent  $k$  and  $u_k: \prod_k M_k \rightarrow \mathbb{R}$  his utility.

### Theorem

Let  $\mathfrak{G} = (\mathcal{X}, M_k, u_k)$  be a nonlinear game. Then:

- ▶ *The stationary points of the generalized replicator dynamics are restricted equilibria (when they exist).*
- ▶ *If  $x(t) \rightarrow q$ , then  $x^*$  is a restricted equilibrium.*
- ▶ *If, in addition, the game admits a potential function  $\phi$ , then all generic minima of  $\phi$  (those that satisfy the strict KKT conditions) are asymptotically stable.*
- ▶ ...



## Conclusions and Future Directions

Some conclusions:

- ▶ Many relevant problems (esp. in engineering and OR) call for thinking outside the box of finite/population games.
- ▶ Existing results: mostly customized ad-hoc solutions to specific problems.
- ▶ Generalized replicator dynamics: first step in developing



## Conclusions and Future Directions

Some conclusions:

- ▶ Many relevant problems (esp. in engineering and OR) call for thinking outside the box of finite/population games.
- ▶ Existing results: mostly customized ad-hoc solutions to specific problems.
- ▶ Generalized replicator dynamics: first step in developing

And some future directions:

- ▶ Stabilità in non-potential games
- ▶ Rates of convergence (usually exponential, but...)
- ▶ Generalize other important dynamics (BR, BNN, etc.)
- ▶ ...