Multiagent Reinforcement Learning for Linear Quadratic Regulators by Zero Order Policy Optimization

Na (Lina) Li
Associate Professor in Electrical Engineering and Applied Mathematics

Yingying Li
Yujie Tang
Guannan Qu
(Now at Caltech)

NSF CPS PI meeting
Workshop: Learning for Control
November 7, 2019
Overarching goal in distributed network systems

Local Rules  ➔  Global Behavior

Transportation
(LA traffic, YouTube)

Power Grids
(Earth at night, YouTube)

Robotic Swarms
(KiloBot, Nagpal’s lab)
This Talk: Multi-Agent Reinforcement Learning of LQR

\[ x(t + 1) = Ax(t) + Bu(t) + w(t) \]

Random Disturbance

\[ x = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}, \quad u = \begin{bmatrix} u_1 \\ \vdots \\ u_N \end{bmatrix} \]

Local control policy parameterized by \( K_i \)

### LTI dynamics

\[ c_i(t) = x(t)^T Q_i x(t) + u(t)^T R_i u(t) \]

\[ c(t) = \sum_{i=1}^N c_i(t) \]

### quadratic cost

### control policy

\[ u_i(t) = f \left( x_{-i}(t), K_i \right) \]

\[ \text{e.g., } u_i(t) = K_i x_{-i}(t) \]

\[ \min_{K_1, \ldots, K_N} J(K) := \lim_{T \to \infty} \mathbb{E} \left[ \frac{1}{T} \sum_{t=1}^{T} c(t) \right] \]
Existing Literature (Incomplete)

Decentralized Control
- Team decision theory [Ho, Chu, Basar, 1972, 1971, 1980]
- Many more

(Centralized) RL
- Many more

Decentralized/Multiagent RL
- Schneider, Wong, Morre, Riedmiller, 1990,
- Lauer, Riedmiller, 2000
- Littman, 1994, 2002
- Busoniu, Babuska, Schutter, 2008
- Kar, Moura, Poor, 2013
- Macua, Chen, Zazo, Sayed, 2014
- Vamvoudakis, hespanha, 2017
- Mathkar, Borkar, 2017
- Lee, Yoon, Hovakimyan, 2018
- Wai, Yang, Wang, Hong, 2018
- Zhang, Yang, Liu, Zhang, Basar, 2018
- Zhang Zavlanos, 2019
- Many more

Our work
Multi-Agent Reinforcement Learning of LQR

Interaction Network

**LTI dynamics**

\[ x(t + 1) = Ax(t) + Bu(t) + w(t) \]

**quadratic cost**

\[ c_i(t) = x(t)^T Q_i x(t) + u(t)^T R_i u(t) \]
\[ c(t) = \sum_{i=1}^{N} c_i(t) \]

**control policy**

\[ u_i(t) = K_i x(t) \]

Agent \( i \)'s observation at time \( t \):

\[ c_i(t), x(t) \]

Local policy:

\[ u_i(t) = K_i x(t) \] **generalizable**

During learning:

Comm. \( c_i(t) \) with neighbors

Communication matrix:

\[ W = [W_{ij}] \]

\[
\begin{align*}
\min_{K:= (K_1, \ldots, K_N)} & \quad J(K) := \lim_{T \to \infty} \mathbb{E} \left[ \frac{1}{T} \sum_{t=1}^{T} c(t) \right] \\
\end{align*}
\]
Policy Gradient

If we know $\nabla J(K)$, run policy gradient

$$K(s + 1) = K(s) - \eta \nabla J(K(s))$$

starting from some stabilizing controller known a priori

What each agent can actually do:
1. Apply a policy $K_i$
2. Observe (and communicate) local state $x_i(t)$ and cost $c_i(t)$ for an episode of finite length
3. Update the policy and iterate

How to bridge this gap?

Zero-order optimization

Gradient estimation: Using Zero-order Information of $J(K)$ to Estimate

[Fazel, Ge, Kakade, Mesbahi, 2018], [Malik, Pananjady, et al, 2019] [Venkataraman, Seiler, 2019][Nocedal, 2019 ]
\textbf{Zero-Order Optimization: Gradient estimation}

\[ f : \mathbb{R}^d \to \mathbb{R} \text{ differentiable.} \]

- Finite-difference estimator:
  \[ G_f^{(2d)}(x; r) := \sum_{k=1}^{d} \frac{f(x + re_k) - f(x - re_k)}{2r} e_k \]
  where \( e_k \) are the orthogonal bases.

- Does not scale well when \( d \) is large

- Stochastic case, \( \frac{f(x+re_i, \xi) - f(x-re_i, \xi)}{2r} ?? \)

\[ F'(x) := \mathbb{E}_{\xi} f(x, \xi) \text{ randomness} \]

- Single-point estimator [Flaxman 2005]:
  \[ \hat{g}(x, D, \xi) := d \frac{f(x + rD, \xi)}{r} z \] where \( D \sim \text{Uni}(\mathbb{S}^{d-1}) \)

- Prop:
  \[ \mathbb{E}_{D, \xi} [\hat{g}(x, D, \xi)] = \nabla F_r(x) \]
  where \( F_r(x) := \mathbb{E}_{D \sim \text{Uni}(\mathbb{B})} [F(x + rz)] \)

- Single-point estimator has large variance
  (inverse proportional to \( r^2 \))

- Therefore, average multiple

\[ \hat{g}(x) \approx \frac{1}{T_B} \sum_{b=1}^{T_B} d \frac{f(x + rD_b, \xi_b)}{r} D_b \]
  where \( D_b \sim \text{Uni}(\mathbb{S}^{d-1}) \).
Algorithm Framework

\[ K_i: \text{ Local control gain of agent } i, \quad u_i = K_i x_i \]
\[ n_K := n_{K_1} + \ldots + n_{K_N}: \text{ Dimension of unknown control gains} \]

1. for \( s = 1, 2, \ldots, T_G \) do

8. Agent \( i \) updates

\[ K_i(s + 1) = K_i(s) - \eta \hat{g}_i(s) \]

Stochastic gradient descent

9. end

Estimate of Gradient of \( J(K) \)
Algorithm Framework

1. for $s = 1, 2, \ldots, T_G$ do

7. Agent $i$ estimates the gradient by

8. Agent $i$ updates

9. end

$K_i$: Local control gain of agent $i$, $u_i = K_i x_i$

$n_K := n_{K_1} + \ldots + n_{K_N}$: Dimension of unknown control gains

\[ \hat{g}_i(s) = \frac{1}{T_B} \sum_{b=1}^{T_B} \frac{n_K}{r} J_i(s, b)D_i(s, b) \]

Averaging multiple single-point gradient estimator

Stochastic gradient descent

$i$'s Estimate of Global Cost $J(K+rD)$
Algorithm Framework

$K_i$: Local control gain of agent $i$, $u_i = K_i x_i$

$n_K := n_{K_1} + \ldots + n_{K_N}$: Dimension of unknown control gains

1. for $s = 1, 2, \ldots, T_G$ do
   2. for $b = 1, 2, \ldots, T_B$ do
      Generate $D(s, b) \sim \text{Uni}(S^{n_K})$
      Agent $i$ implements $K_i(s) + r D_i(s, b)$
      Agent $i$ produces an estimate of the global cost $\hat{J}_i(s, b)$ through observation of the trajectory and communication with neighbors

6. end

7. Agent $i$ estimates the gradient by

$$\hat{g}_i(s) = \frac{1}{T_B} \sum_{b=1}^{T_B} \frac{n_K}{r} \hat{J}_i(s, b) D_i(s, b)$$

Averaging multiple single-point gradient estimator

8. Agent $i$ updates

$$K_i(s + 1) = K_i(s) - \eta \hat{g}_i(s)$$

Stochastic gradient descent

9. end

Can we get some performance guarantee?
Estimating Global Cost

- What agent $i$ observes: $c_i(t), t = 1, 2, \ldots, T_J$

- If there’s only one agent:

$$\mu(T_J) := \frac{1}{T_J} \sum_{t=1}^{T_J} c(t) \approx J(K) \quad \iff \quad \begin{cases} \mu(0) = 0 \\ \mu(t) = \frac{t-1}{t} \mu(t-1) + \frac{1}{t} c(t) \end{cases}$$

- Multi-agent:

$$\mu_i(0) = 0$$
$$\mu_i(t) = \frac{t-1}{t} \sum_{j=1}^{N} W_{ij} \mu_j(t-1) + \frac{1}{t} c_i(t)$$

$W = [W_{ij}]$: communication matrix, doubly stochastic
Uniform Distribution on Unit Sphere: \( D := (D_1, D_2, ..., D_N) \)

- **Lemma:** Suppose \( V \sim \mathcal{N}(0, I_d) \), then \( V/||V|| \sim \text{Uni}(S^{d-1}) \)

- \( V = (V_1, \ldots, V_N) \sim \mathcal{N}(0, I_{n_K}) \) can be generated decentralized where \( V_i \sim \mathcal{N}(0, I_{n_{k_i}}) \)

- How to compute \( ||V|| \)?
  - Through consensus
  - Can be carried out *simultaneously* with global cost estimate \( T_J \text{ steps} \)
Algorithm Framework

1 for $s = 1, 2, \ldots, T_G$ do
2    for $b = 1, 2, \ldots, T_B$ do
3        $(D_i(s, b))_{i=1}^{N} \leftarrow \text{SampleUnitSphere}(T_J)$
4        $(\hat{J}_i(s, b))_{i=1}^{N} \leftarrow \text{GlobalCostConsensus}(K(s) + r D(s, b), T_J)$
5    end
6    Agent $i$ estimates the gradient by
7        $\hat{g}_i(s) = \frac{1}{T_B} \sum_{b=1}^{T_B} \frac{n_K}{r} \hat{J}_i(s, b) D_i(s, b)$
8 end

Agent $i$ updates

$K_i(s + 1) = K_i(s) - \eta \hat{g}_i(s)$

$J_i$ is "sufficiently bounded" if $\eta, r$ are chosen properly $\implies K(s) + r D(s, b)$ is stabilizing w.h.p
Algorithm Framework

1. $K(1) \leftarrow$ known stabilizing controller
2. for $s = 1, 2, \ldots, T_G$ do
3.   for $b = 1, 2, \ldots, T_B$ do
4.     $(D_i(s, b))_{i=1}^N \leftarrow \text{SampleUnitSphere}(T_J)$
5.     $(\tilde{J}_i(s, b))_{i=1}^N \leftarrow \text{GlobalCostConsensus}(K(s) + rD(s, b), T_J)$
6.     $\hat{J}_i(s, b) = \min\left\{\tilde{J}_i(s, b), \bar{J}\right\}$
7.   end
8. Agent $i$ estimates the gradient by
9.   $\hat{g}_i(s) = \frac{1}{T_B} \sum_{b=1}^{T_B} \frac{n_K}{r} \tilde{J}_i(s, b)D_i(s, b)$
10. Agent $i$ updates
11.   $K_i(s + 1) = K_i(s) - \eta \hat{g}_i(s)$
12. end
Performance guarantee and sample complexity

**Theorem** (informal, current)
Given a stabilizing initial controller $K(1)$. For any $0 < \varepsilon < 1$, when

$$0 < r < O(\sqrt{\varepsilon}), \quad T_B \geq \Omega \left( \frac{n_K^2 \eta}{r^2 \varepsilon} \right), \quad T_J \geq \Omega \left( \frac{1}{1 - \rho(W - \frac{1}{N}11^\top)} \right) \frac{n_K N}{r \sqrt{\varepsilon}}$$

and under stepsize $0 < \eta < O\left( \frac{r}{n_K} \right)$, after $T_G = \Theta \left( \frac{n_K}{\eta \varepsilon} \right)$ iterations of policy gradient, with probability at least 0.7,

- $K(1), \ldots, K(T_G)$ are all stabilizing controllers, and

- $\frac{1}{T_G} \sum_{s=1}^{T_G} \| \nabla J(K(s)) \|_F^2 \leq \varepsilon$

**Sample complexity:** $T_G T_B T_J = \Theta \left( \frac{n_K^3 N}{(1 - \rho)\varepsilon^4} \right)$

Li, Tang, Zhang, Li, “Multiagent Reinforcement Learning Based on Zero-order Policy Gradient,” coming soon
Numerical Simulation

- 4-zone Building
- Outdoor temperature: $x^o = 30^\circ C$
- Target temperature: $x_{set} = 22^\circ C$
- Thermal Dynamic model:

$$C_i \dot{x}_i = \frac{x^o - x_i}{R_i} + \sum_{j \in N(i)} \frac{x_j - x_i}{R_{ij}} + u_i + Q_i + w_i$$

- Objective:

$$\min \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \|x_t - x_{set}\|^2 + \|u_t\|^2$$

- Controller is of the form

$$u_i = K_i x_i + b_i$$

Fig. 1: Schematic of a typical AHU&VAV system.

Numerical Simulation

Initial $u_i = 0 x_i + 0$

Cost Curve with Different Initial Controllers
Thermal Dynamics
Stage Cost Dynamics
Open Questions

• Global convergence property for *special structure* systems?
• Control policy: beyond the linear, static controller structure?
• Comparison to indirect learning methods?
  ➢ Learn dynamical model from partial observations then design the controller, in particular LQG?
• Robustness?
• Fundamental performance limit and tradeoffs?
• Experimental Test?
Other Multiagent Learning in Our Group

Distributed Zero-order Opt
(Extreme-Seeking Control)

• Minimize $\sum_i f_i(x_1, x_2, \ldots, x_n)$
• Nonconvex objectives
• Only inquires local objective values

Qu, Wierman, Li, “Exploiting Fast Decaying and Locality in Networked Multi-Agent MDP Learning”, Coming soon

Multiagent RL for Networked MDPs

State: $s_i \in S_i$ Finite Set
Action: $a_i \in A_i$ Finite Set
Transition Prob.: $P(s^+|s, a) = \prod_{i=1}^n P_i(s_i^+|s_{N_i}, a_i)$
Stage Reward: $r(s) = \sum_{i=1}^n r_i(s_i, a_i)$

Tang, Ren, Li, “Distributed Zero-order Multi-agent Nonconvex Optimization”, Coming soon
Numerical Example for Distributed Zero-order Opt

Thank you!