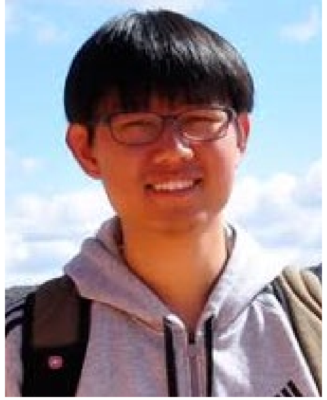
HARVARD
John A. Paulson
School of Engineering
and Applied Sciences

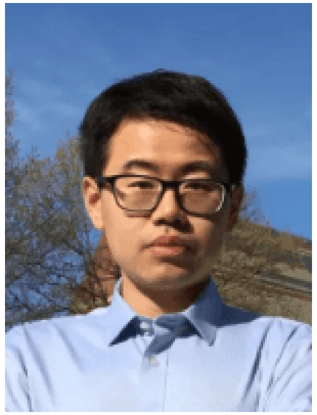
Multiagent Reinforcement Learning for Linear Quadratic Regulators by Zero Order Policy Optimization

Na (Lina) Li

Associate Professor in Electrical Engineering and Applied Mathematics


Yingying Li

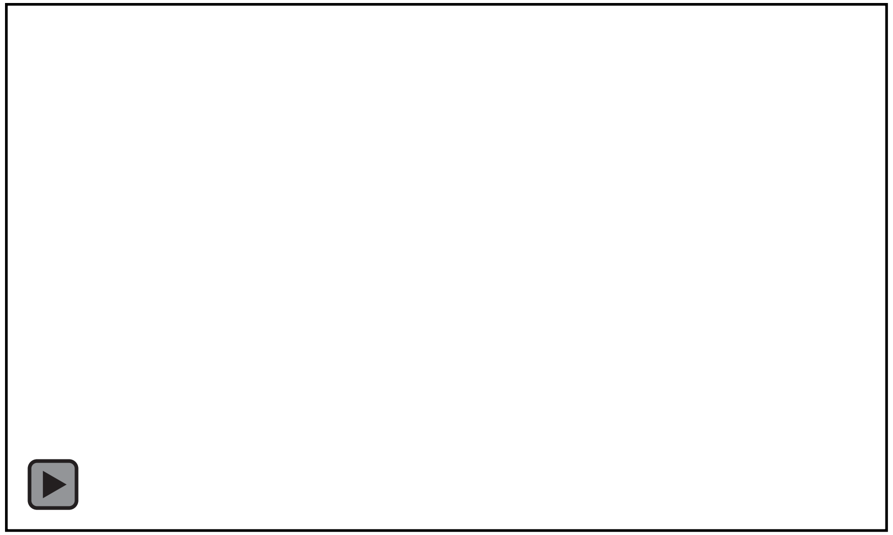

Yujie Tang


Guannan Qu
(Now at Caltech)

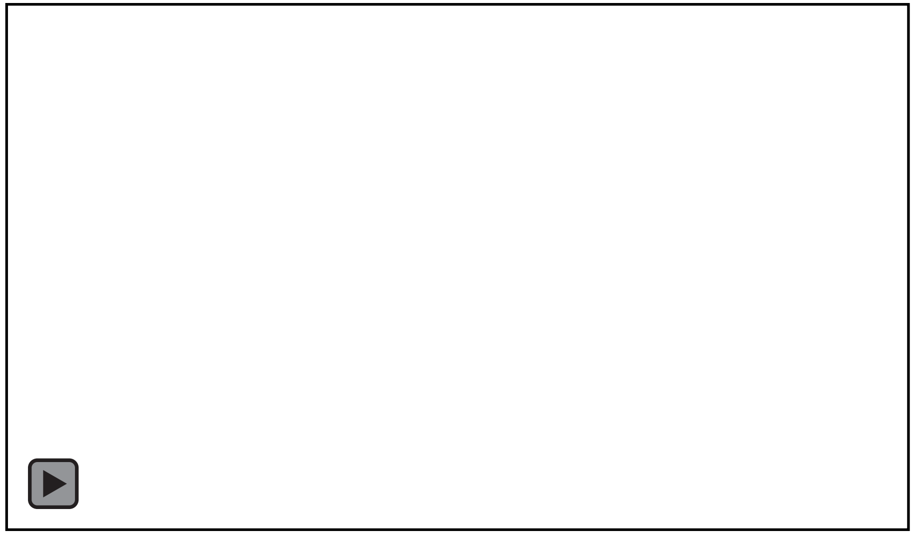
NSF CPS PI meeting
Workshop: Learning for Control
November 7, 2019

Overarching goal in distributed network systems

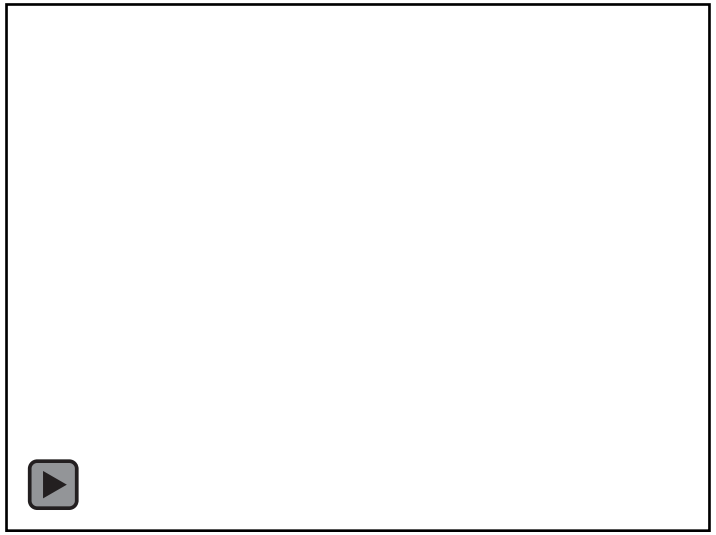
Local Rules* → *Global Behavior



Transportation
(LA traffic, YouTube)

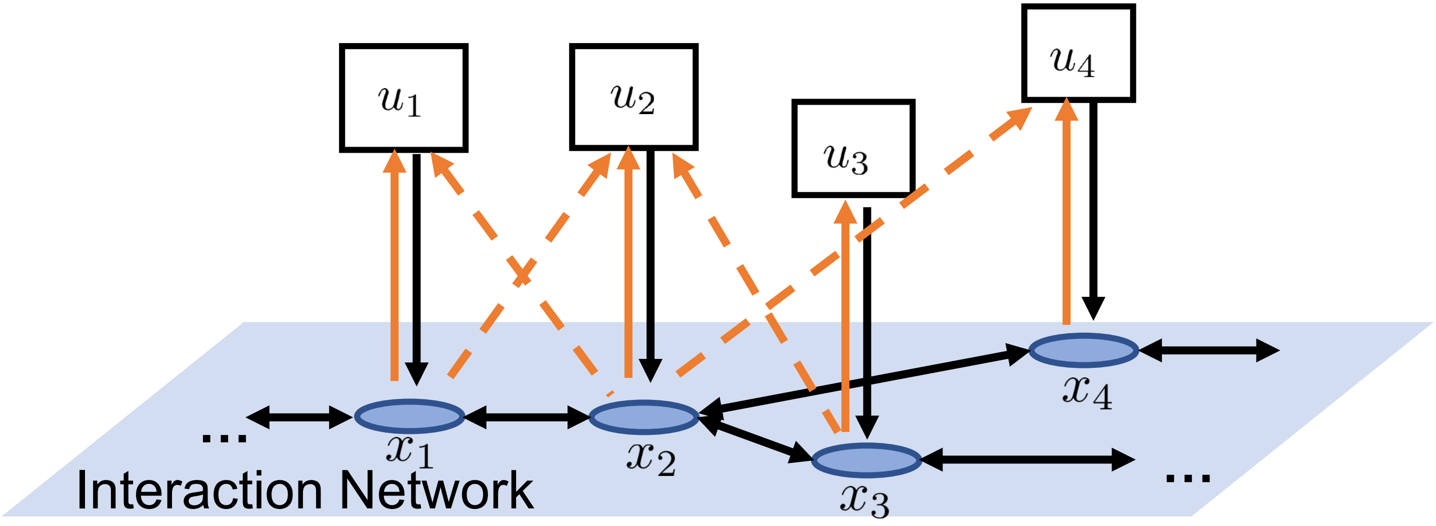


Power Grids
(Earth at night, YouTube)



Robotic Swarms
(KiloBot, Nagpal’s lab)

This Talk: Multi-Agent Reinforcement Learning of LQR



$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \quad u = \begin{bmatrix} u_1 \\ \vdots \\ u_N \end{bmatrix}$$

Local control policy parameterized by K_i

LTI dynamics

$$x(t+1) = Ax(t) + Bu(t) + w(t)$$

Random Disturbance

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

$$u_i(t) = f\left(x_{\mathcal{I}}(t), K_i\right)$$
$$\text{e.g., } u_i(t) = K_i x_{\mathcal{I}}(t)$$

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

Existing Literature (Incomplete)

Decentralized Control

- Decentralized control methods for large scale systems [Witsenhausen,1968, Athans, 1974, Siljak, 1976, 1988, Mayne, 1976, Morse, 1976, Speyer, 1979]
- Team decision theory [Ho, Chu, Basar, 1972, 1971, 1980]
- Convex opt approaches [Rotkowitz, Lall, Lessard, Rantzer 2005, 2010, 1999, 2009, 2013]
- Network Control, Formation Control [Olfati-saber, Murray, 2004, Bullo, 2004, 2009, Jadbabaie, 2006]
- Many more

(Centralized) RL

- Adaptive control: System ID, Self-tuning regulator, etc [Wittenmark, 1975, 1971, 1984, Astrom, 1983, 1987, 1995, Ljung, 1974, 1977, 2011, etc]
- Dynamic programming [Bertsekas, Tsitsiklis, 1996, Lewis, 2005, 2009, 2010]
- [Vamvoudakis, Lewis, 2010, 2012, Abbasi-Yadkori, Szepesvari, 2011, 2014, Matni, Rechtz, 2017, 2018, 2019, Ouyang, Jain, 2017]
- Zero-order Opt [Fazel, Ge, Kakade, Mesbahi, 2018, Malik, Pananjady, et al 2019, Venkataranman, Seiler, 2019, Nocedal, 2019]
- Many more

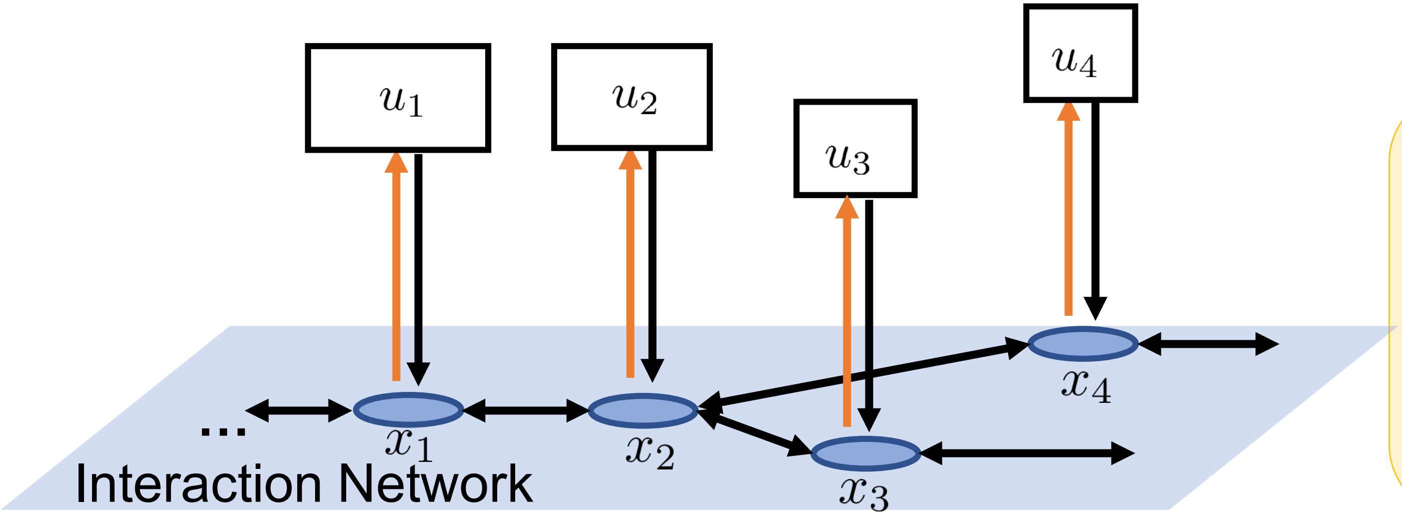
Decentralized/Multiagent RL

- 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

Decentralized Learning for Decentralized (Local) Policies based on Zero-order Opt.

Multi-Agent Reinforcement Learning of LQR



Agent i's observation at time t: $c_i(t)$,$x_{\mathcal{I}}(t)$

Local policy: $u_i(t) = K_i x_{\mathcal{I}}(t)$ generalizable

During learning: Comm. $c_i(t)$ with neighbors

Communication matrix: $W = [W_{ij}]$

LTI dynamics	quadratic cost	control policy
$x(t+1) = Ax(t) + Bu(t) + w(t)$	$c_i(t) = x(t)^T Q_i x(t) + u(t)^T R_i u(t)$ $c(t) = \sum_{i=1}^N c_i(t)$	$u_i(t) = K_i x_{\mathcal{I}}(t)$

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

Policy Gradient

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

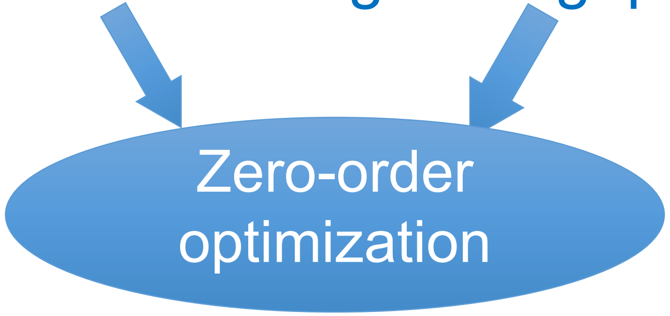
$$K(s+1) = K(s) - \eta \nabla J(K(s))$$

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?



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]

Zero-Order Optimization: Gradient estimation

$f : \mathbb{R}^d \to \mathbb{R}$ differentiable.

$F(x) := \mathbb{E}_\xi f(x, \xi)$ randomness

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

- Single-point estimator [Flaxman 2005]:

$$\hat{g}(x, D, \xi) := d\frac{f(x+rD,\xi)}{r} z \text{ where } D \sim \text{Uni}(\mathbb{S}^{d-1})$$

- 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: 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

---

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

Agent i updates

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

Stochastic gradient descent

Estimate of Gradient of J(K)

end

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

---

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

????

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

Random direction

Agent i estimates the gradient by

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

Number of single point estimators

Averaging multiple single-point gradient estimator

Agent i updates

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

Stochastic gradient descent

end

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

---

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

  for $b = 1, 2, \ldots, T_B$ **do**

    Generate $D(s,b) \sim \text{Uni}(\mathbb{S}^{n_K})$

    Agent i implements $K_i(s) + rD_i(s,b)$

    Agent i produces an estimate of the global cost $\hat{J}_i(s,b)$ through observation of the trajectory and communication with neighbors

  end

  Agent i estimates the gradient by

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

  Agent i updates

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

end

Estimation of Global cost J(K+D)

How?

How to ensure the stability?

How? Number of samples we need?

Averaging multiple single-point gradient estimator

Stochastic gradient descent

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:

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

Can be computed in real time

- Multi-agent:

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

Variation: mini-batch scheme

$W = [W_{ij}]$: 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 $V/||V|| \sim \text{Uni}(\mathbb{S}^{d-1})$

- $V = (V_1, \ldots, V_N) \sim \mathcal{N}(0, I_{n_K})$ can be generated decentralizedly 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 steps)

Algorithm Framework

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

  for $b = 1, 2, \ldots, T_B$ **do**

    $(D_i(s,b))_{i=1}^N \leftarrow \texttt{SampleUnitSphere}(T_J)$

    $(\hat{J}_i(s,b))_{i=1}^N \leftarrow \texttt{GlobalCostConsensus}(K(s) + rD(s,b), T_J)$

  end

  Agent i estimates the gradient by

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

  Agent i updates

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

end

How to ensure the stability?

$$\left.\begin{array}{r} J_i \text{ is "sufficiently bounded"} \\ \eta, r \text{ are chosen properly} \end{array}\right\} \Longrightarrow K(s) + rD(s,b) \text{ is stabilizing w.h.p}$$

Algorithm Framework

$K(1) \leftarrow$ known stabilizing controller
for $s = 1, 2, \ldots, T_G$ **do**
  for $b = 1, 2, \ldots, T_B$ **do**
    $(D_i(s,b))_{i=1}^N \leftarrow \texttt{SampleUnitSphere}(T_J)$
    $(\tilde{J}_i(s,b))_{i=1}^N \leftarrow \texttt{GlobalCostConsensus}(K(s) + rD(s,b), T_J)$
    $\hat{J}_i(s,b) = \min\left\{\tilde{J}_i(s,b), \bar{J}\right\}$
  end
  Agent i estimates the gradient by

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

  Agent i updates

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

end

parameters

- r : smoothing radius
- η : step size
- T_J : length of an episode
- T_B : batch size
- T_G : # of outer iterations

Performance guarantee and sample complexity

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

$$0 < r < O(\sqrt{\epsilon}),\ T_B \geq \Omega(\frac{n_K^2 \eta}{r^2 \epsilon}),\ T_J \geq \Omega(\underline{\frac{1}{1-\rho(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top)}} \frac{n_K N}{r\sqrt{\epsilon}})$$

and under stepsize $0 < \eta < O(\frac{r}{n_K})$, after $T_G = \Theta(\frac{\Delta_1}{\eta\epsilon})$ iterations of policy gradient, with probability at least 0.7,

- $K(1), \ldots, K(T_G)$ are all stabilizing controllers, and
- $\frac{1}{T_G} \sum_{s=1}^{T_G} \|\nabla J(K(s))\|_F^2 \leq \epsilon$

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

Li, Tang, Zhang, Li, "Multiagent Reinforcement Learning Based on Zero-order Policy Gradient," coming soon

Numerical Simulation

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

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

External heat gain

- Objective:

$$\min \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} ||x_t - x_{set}||_2^2 + ||u_t||_2^2$$

- Controller is of the form

$$u_i = K_i\, x_i + b_i$$

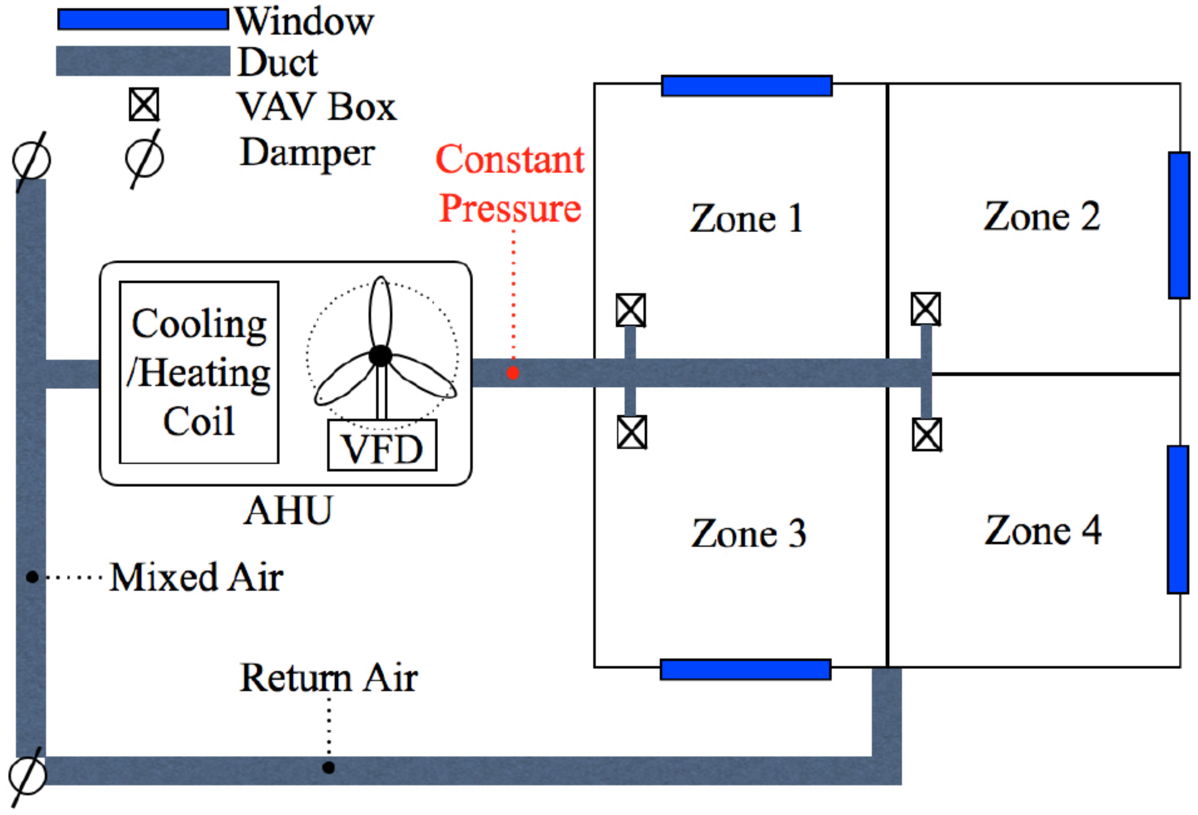
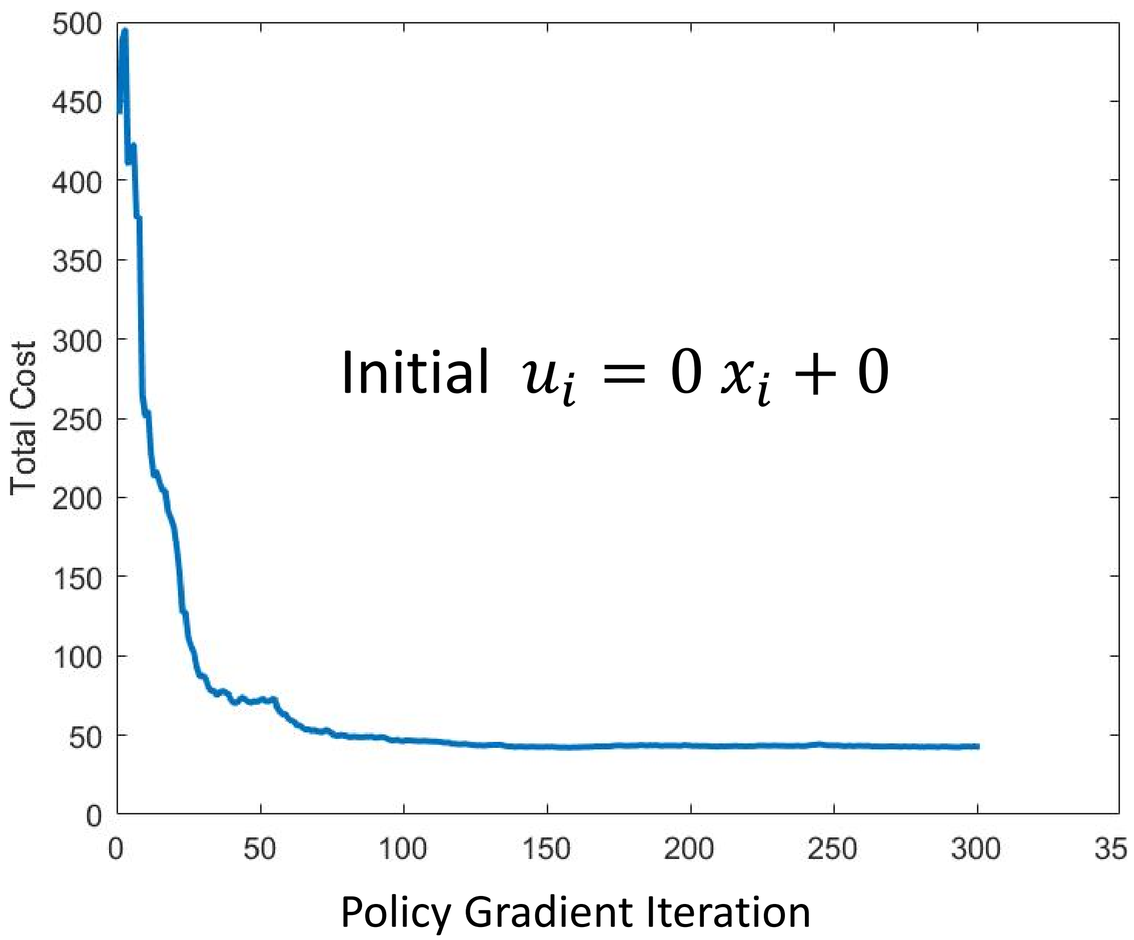
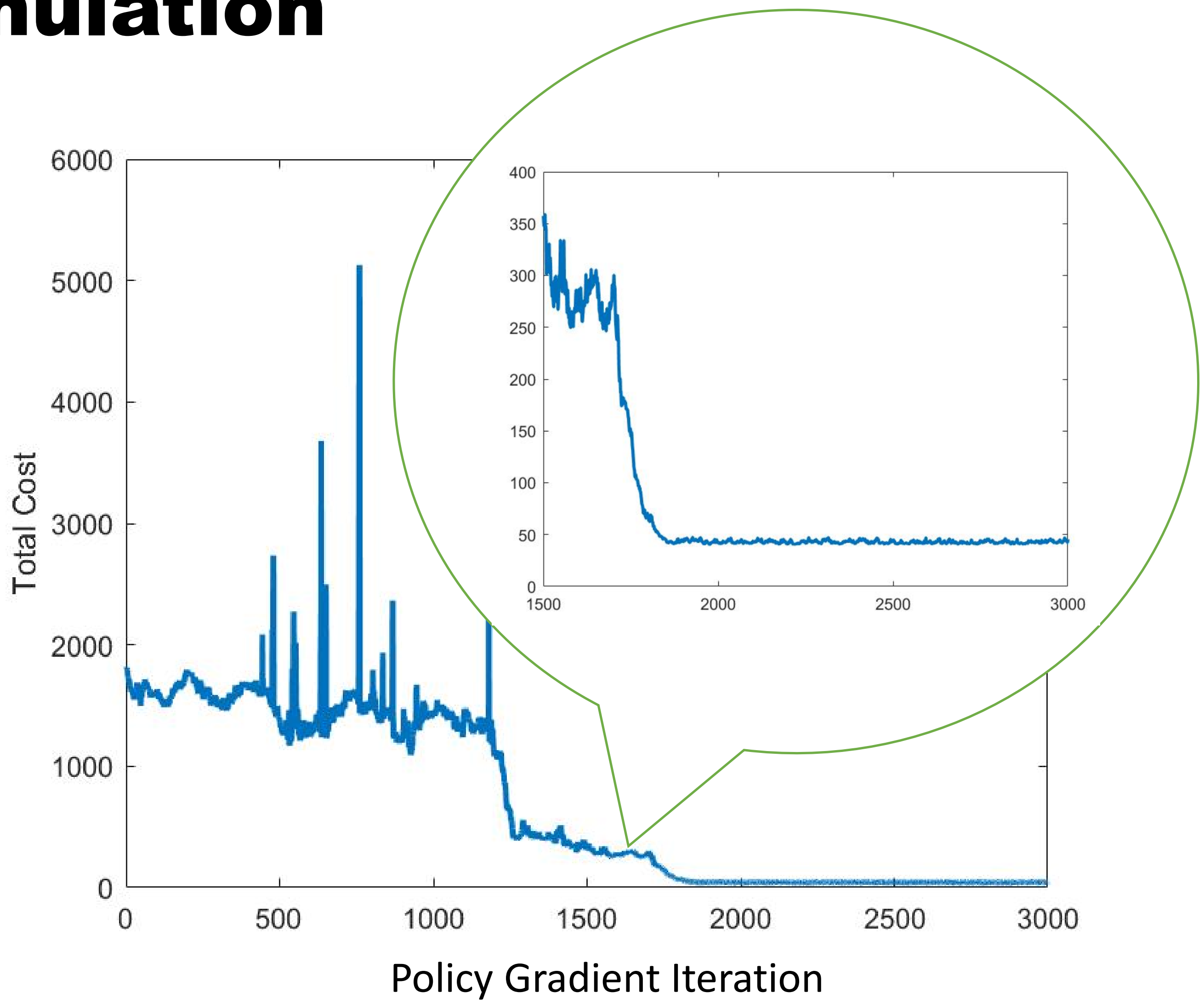


Fig. 1: Schematic of a typical AHU&VAV system.

X Zhang et. al 'Decentralized Temperature Control via HVAC Systems in Energy Efficient Buildings: An Approximate Solution Procedure'

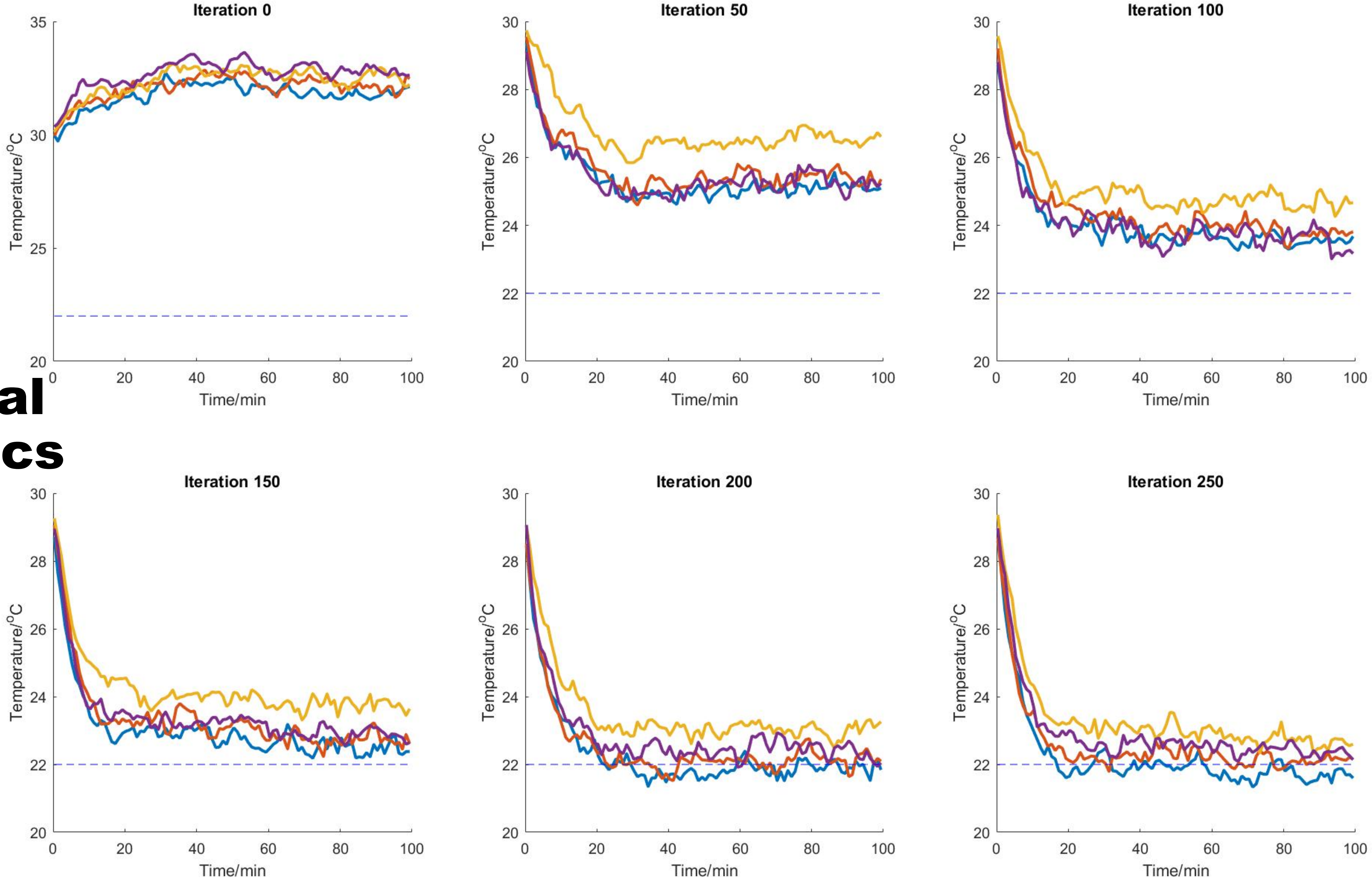
Numerical Simulation



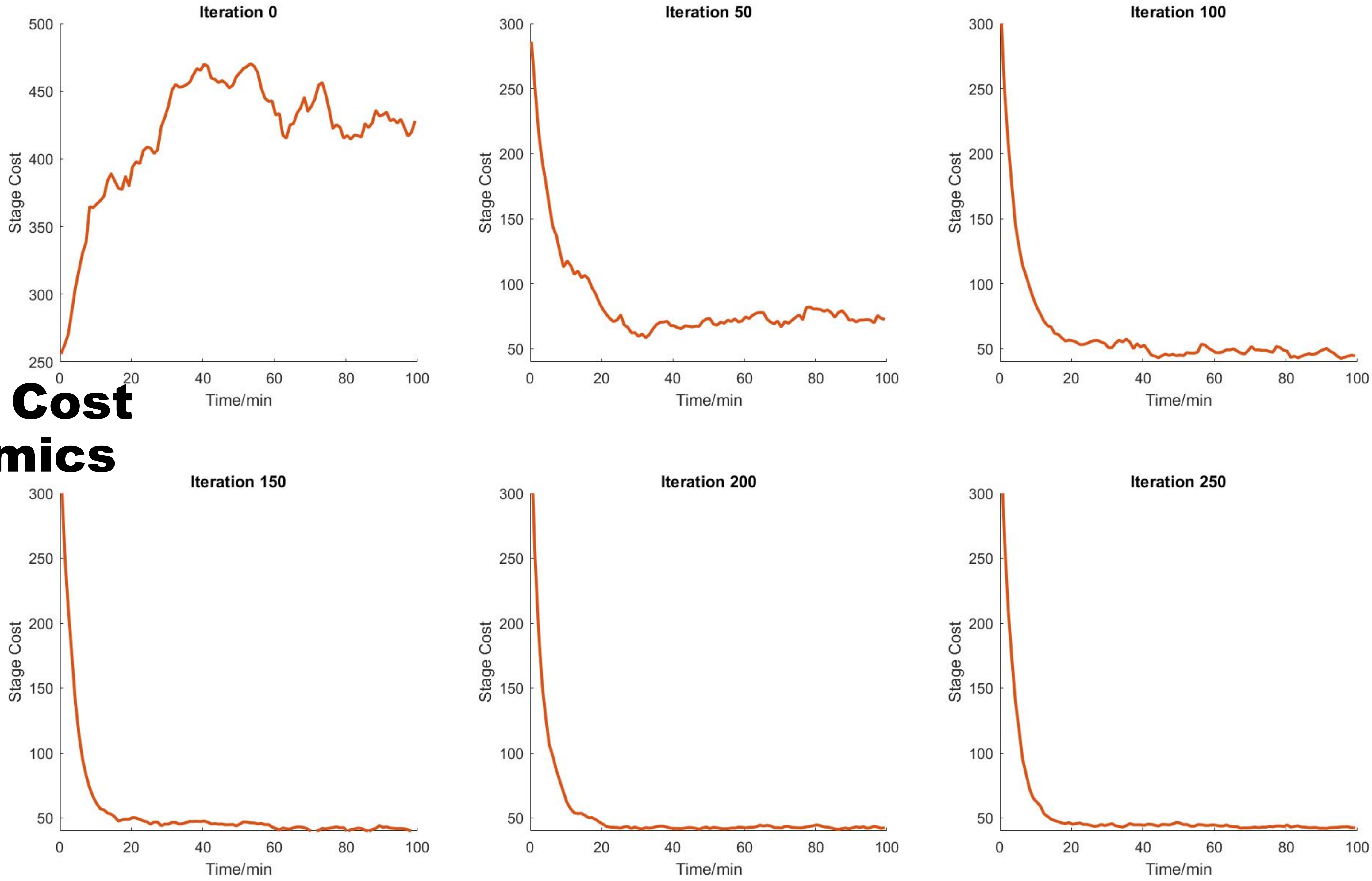


Cost Curve with Different Initial Controllers

Thermal Dynamics



Stage Cost Dynamics



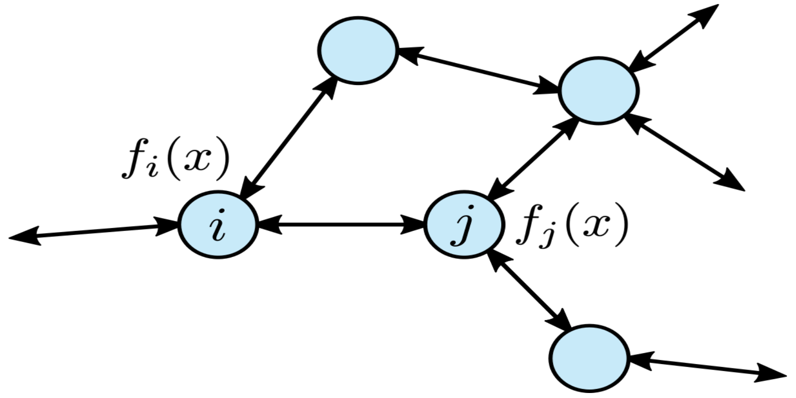
Open Questions

- 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**



Tang, Ren, Li, "Distributed Zero-order Multi-agent Nonconvex Optimization", Coming soon

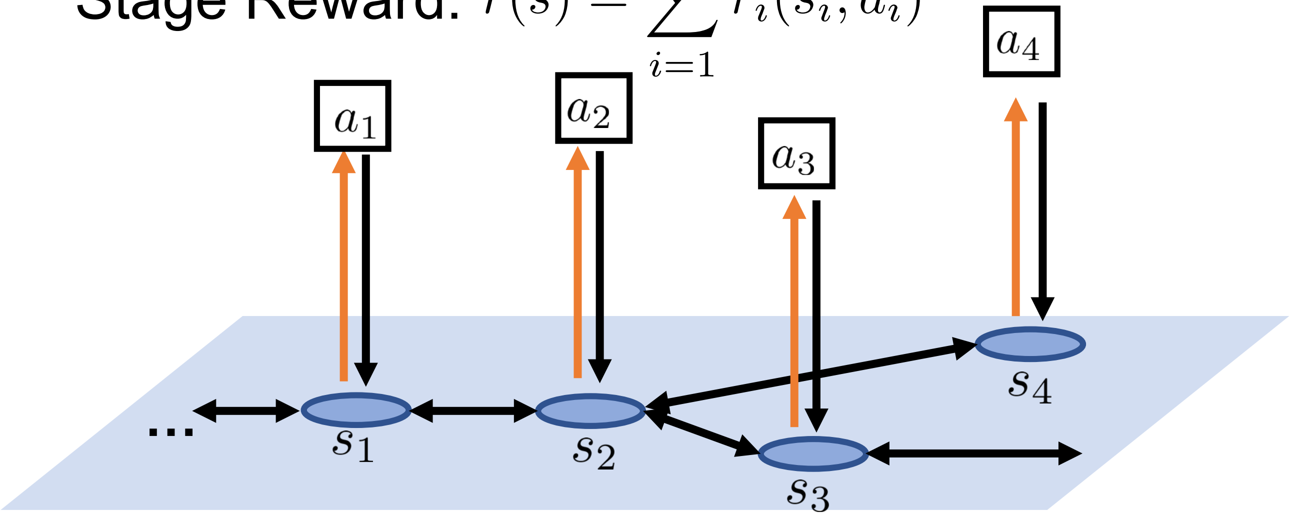
Multiagent RL for Networked MDPs

State: $s_i \in \mathcal{S}_i$ Finite Set

Action: $a_i \in \mathcal{A}_i$ Finite Set

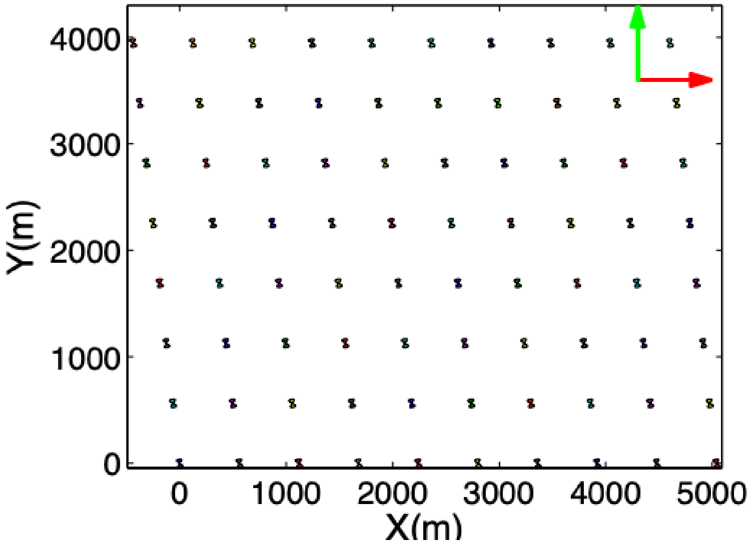
Transition Prob.: $P(s^+|s,a) = \prod_{i=1}^{n} P_i(s_i^+|s_{N_i}, a_i)$

Stage Reward: $r(s) = \sum_{i=1}^{n} r_i(s_i, a_i)$

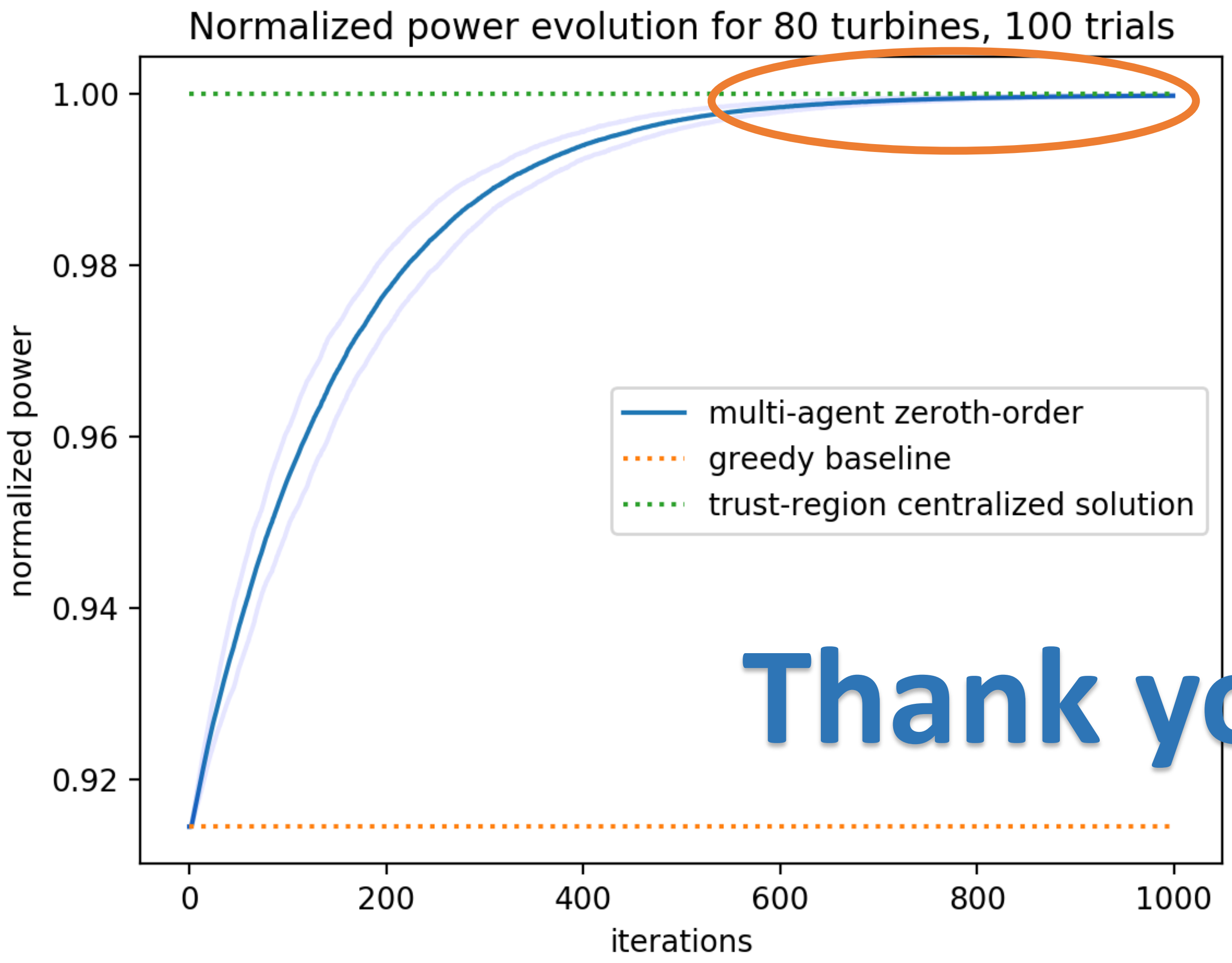


Qu, Wierman, Li, "Exploiting Fast Decaying and Locality in Networked Multi-Agent MDP Learning", Coming soon

Numerical Example for Distributed Zero-order Opt



(a) Wind farm layout



Thank you!