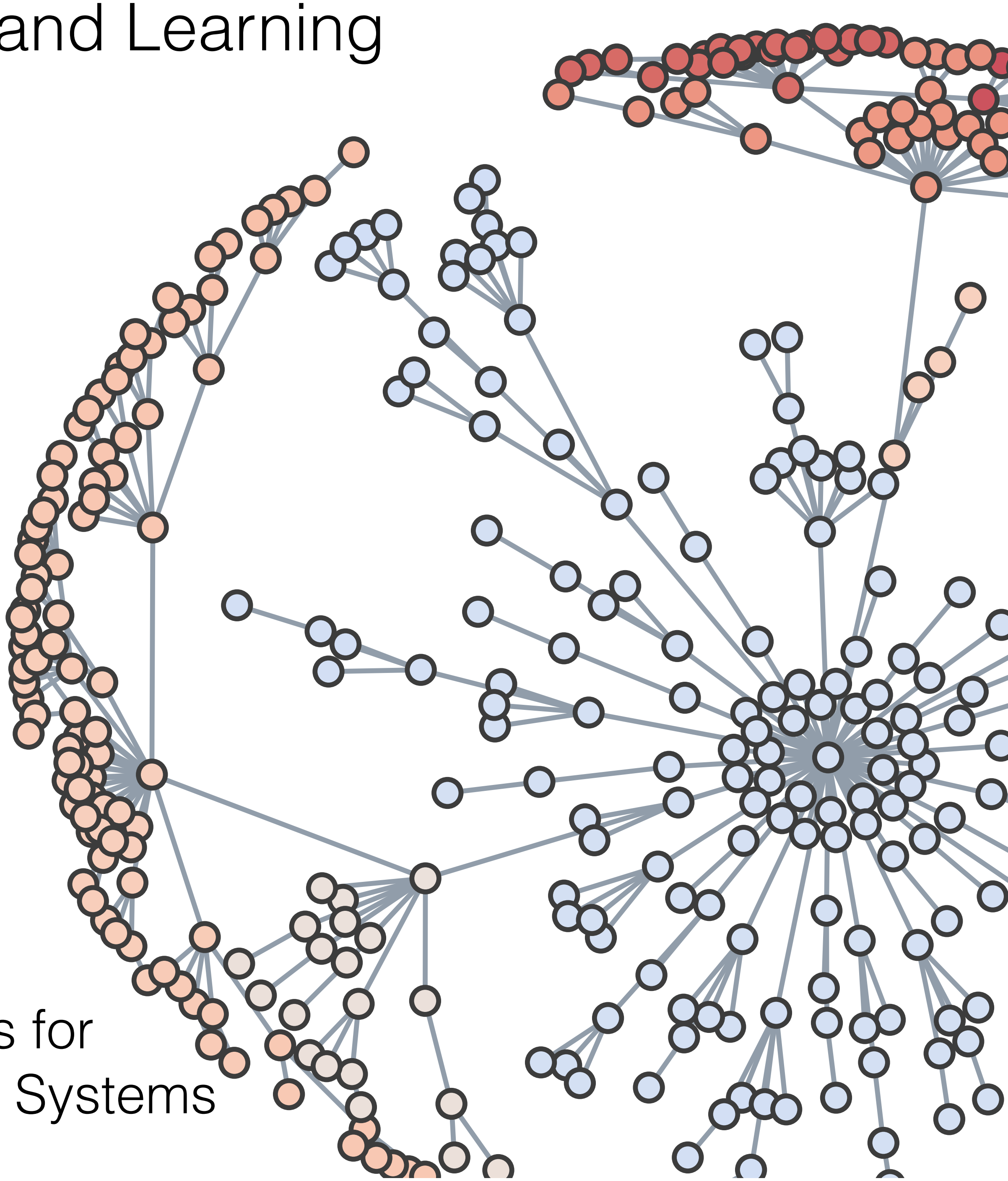
Dynamical Systems with Multiplicative Noise:

Control, Sparse Design, and Learning



Tyler Summers

THE UNIVERSITY OF TEXAS AT DALLAS

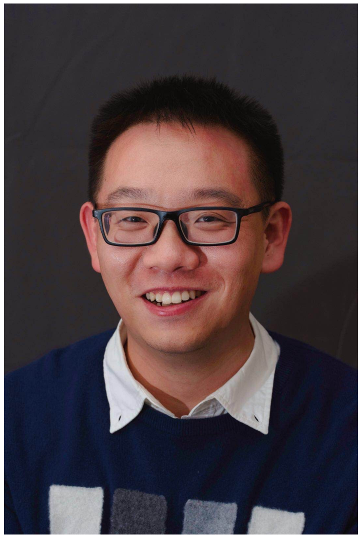
NREL
NATIONAL RENEWABLE ENERGY LABORATORY

April 12, 2019

Workshop on Innovative Optimization & Control Methods for Highly Distributed Autonomous Systems

collaborators

Ben Gravell



Yi Guo

funding

Peyman M. Esfahani

Wouter Jongeneel

dynamical systems with multiplicative noise

$$x_{t+1} = \left(A + \sum_{i=1}^{p} {}_{ti} A_i \right) x_t + \left(B + \sum_{i=1}^{q} {}_{ti} B_i \right) u_t + w_t$$

$$y_t = \left(C + \sum_{i=1}^{r} {}_{ti} C_i \right) x_t + v_t$$

- state $x_t \in \mathbf{R}^n$, input $u_t \in \mathbf{R}^m$, output $y_t \in \mathbf{R}^l$

- nominal system matrices A, B, C

- additive noises w_t, v_t with covariances W, V

- multiplicative noises ${}_{ti},\ {}_{ti},\ {}_{ti}$ with variances $\alpha_i,\ {}_i,\ {}_i$

dynamical systems with multiplicative noise

$$x_{t+1} = \quad A + \sum_{i=1}^{p} \Bigg({}_{ti} A_i \Bigg) \Bigg(x_t + \quad B + \sum_{i=1}^{q} \Bigg({}_{ti} B_i \Bigg) \Bigg(u_t + w_t$$

$$y_t = \quad C + \sum_{i=1}^{r} \Bigg({}_{ti} C_i \Bigg) \Bigg(x_t + v_t$$

optimal control

minimize

$$\mathbf{E} \sum_{t=0}^{\infty} (x_t^T Q x_t + u_t^T R u_t)$$

optimal estimation

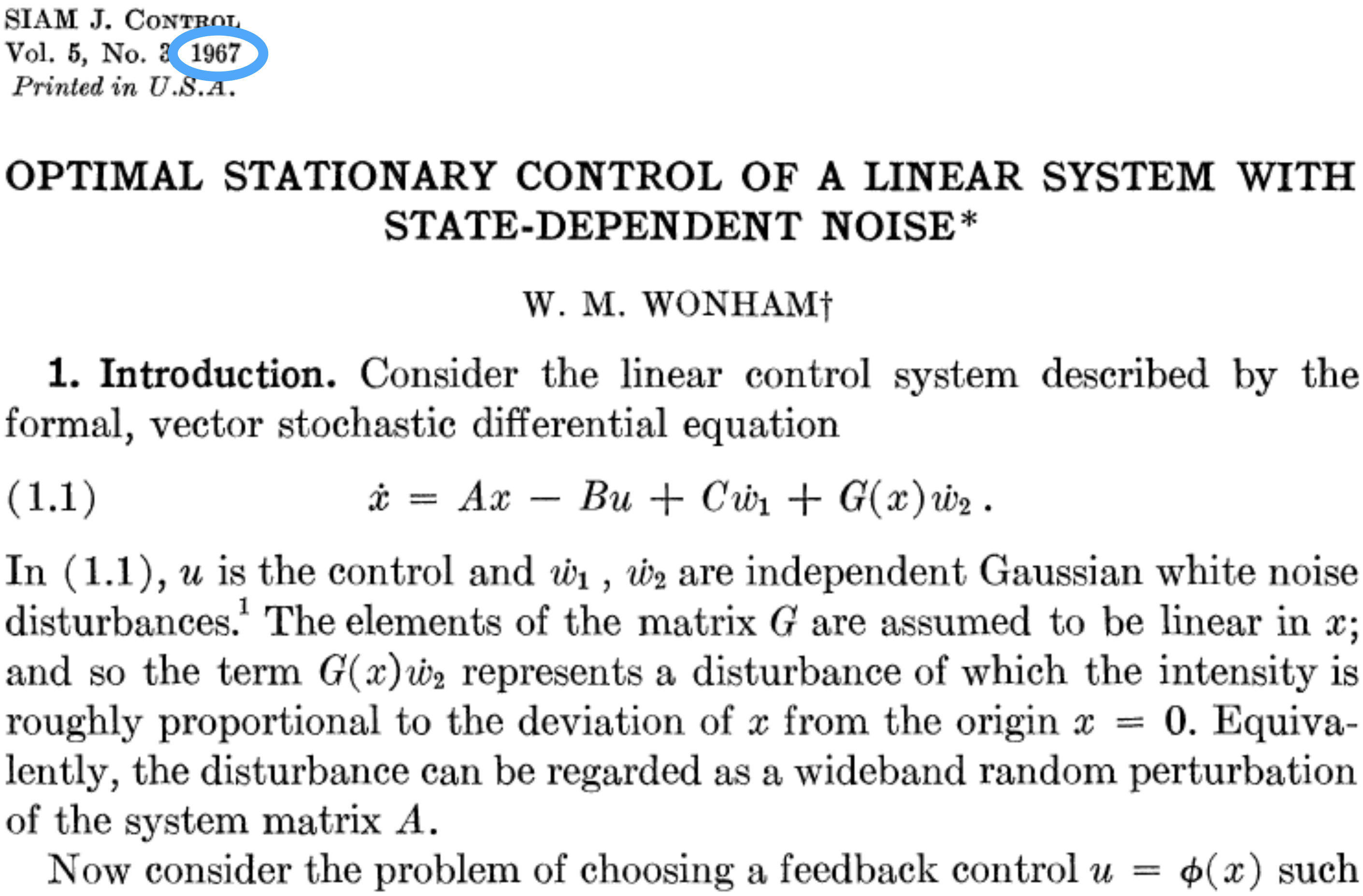
minimize

$$\lim_{T \to \infty} \frac{1}{T} \mathbf{E} \sum_{t=0}^{T} e_t^T e_t$$

$$e_t = x_t - \hat{x}_t$$

dynamical systems with multiplicative noise

not a new topic…



SIAM J. Control
Vol. 5, No. 3 1967
Printed in U.S.A.

OPTIMAL STATIONARY CONTROL OF A LINEAR SYSTEM WITH STATE-DEPENDENT NOISE*

W. M. WONHAM†

1. Introduction. Consider the linear control system described by the formal, vector stochastic differential equation

$$\dot{x} = Ax - Bu + C\dot{w}_1 + G(x)\dot{w}_2 . \tag{1.1}$$

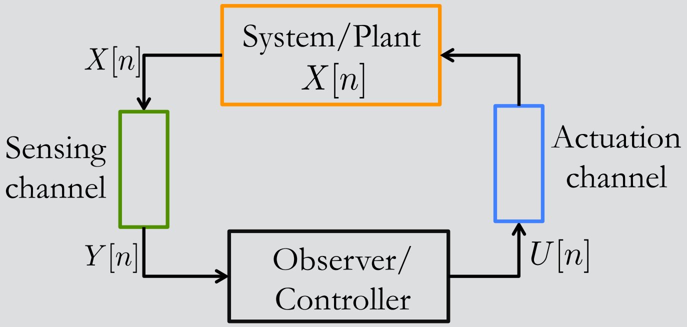
In (1.1), u is the control and $\dot{w}_1$, $\dot{w}_2$ are independent Gaussian white noise disturbances.[1] The elements of the matrix G are assumed to be linear in x; and so the term $G(x)\dot{w}_2$ represents a disturbance of which the intensity is roughly proportional to the deviation of x from the origin $x = 0$. Equivalently, the disturbance can be regarded as a wideband random perturbation of the system matrix A.

Now consider the problem of choosing a feedback control $u = \phi(x)$ such

dynamical systems with multiplicative noise

… but increasingly relevant in emerging highly distributed autonomous systems

networked control systems



System/Plant $X[n]$

$X[n]$

Sensing channel

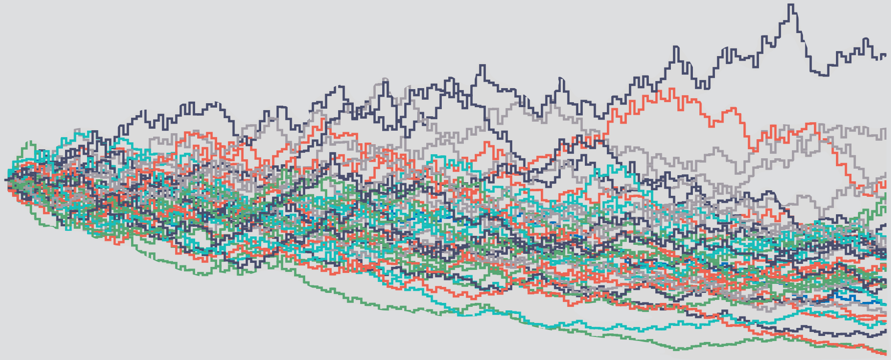
$Y[n]$

Observer/ Controller

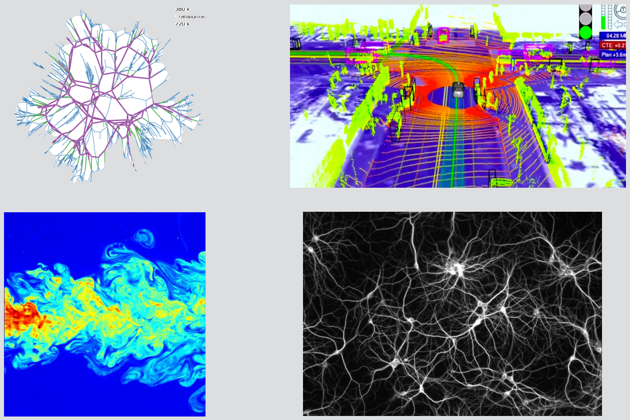
$U[n]$

Actuation channel

data-driven/adaptive/learning



complex processes + sensors



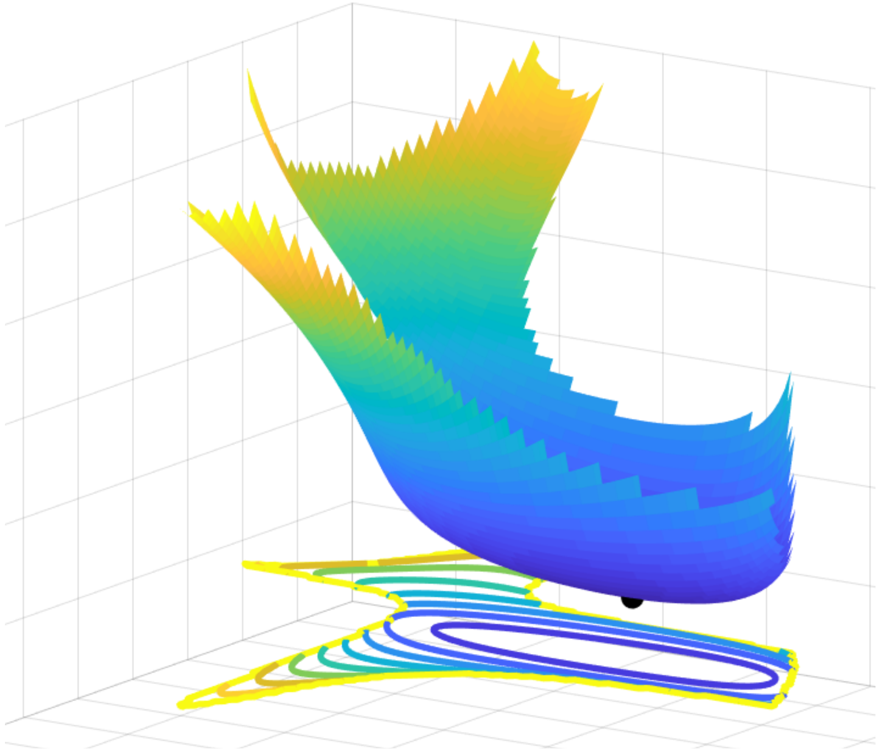
uncertainty and robustness

talk outline

- exact global convergence of policy gradient
- sparse control architecture design
- data-driven control

- extensions and variations
- conclusions

exact global convergence of policy gradient



dynamical systems with multiplicative noise

$$x_{t+1} = \quad A + \sum_{i=1}^{p} \Bigg({}_{ti} A_i \Bigg) \Bigg(x_t + \quad B + \sum_{i=1}^{q} \Bigg({}_{ti} B_i \Bigg) \Bigg(u_t$$

optimal control

minimize

$$\mathbf{E} \sum_{t=0}^{\infty} (x_t^T Q x_t + u_t^T R u_t)$$

assumption:

mean square stabilizability

optimal control of dynamical systems with multiplicative noise

- exact solution via dynamic programming yields *generalized* Riccati equation

$$P = Q + A^T P A + \sum_{i=1}^{p} \alpha_i A_i^T P A_i - A^T P B(R + B^T P B + \sum_{j=1}^{q} \quad {}_j B_j^T P B_j)^{-1} B^T P A$$

$$K^* = - \quad R + B^T P B + \sum_{i=1}^{q} \beta_i B_i^T P B_i \Bigg)\Bigg(^{1} B^T P A$$

- solved via recursion with $P_0 = Q$ (value iteration)

$$P_{t+1} = Q + A^T P_t A + \sum_{i=1}^{p} \alpha_i A_i^T P_t A_i - A^T P_t B(R + B^T P_t B + \sum_{j=1}^{q} \quad {}_j B_j^T P_t B_j)^{-1} B^T P_t A$$

optimal control of dynamical systems with multiplicative noise

- exact solution via dynamic programming yields *generalized* Riccati equation

$$P = Q + A^T P A + \sum_{i=1}^{p} \alpha_i A_i^T P A_i - A^T P B(R + B^T P B + \sum_{j=1}^{q} \;{}_j B_j^T P B_j)^{-1} B^T P A$$

$$K^* = - \quad R + B^T P B + \sum_{i=1}^{q} \beta_i B_i^T P B_i \Bigg)\Bigg(^{1} \; B^T P A$$

- solved via recursion with $P_0 = Q$ (value iteration)

$$P_{t+1} = Q + A^T P_t A + \sum_{i=1}^{p} \alpha_i A_i^T P_t A_i - A^T P_t B(R + B^T P_t B + \sum_{j=1}^{q} \;{}_j B_j^T P_t B_j)^{-1} B^T P_t A$$

optimal control of dynamical systems with multiplicative noise

- alternative: fix linear policy $u_t = Kx_t$, define cost

$$C(K) = \mathop{\mathbf{E}}_{x_0,\ \ _i,\ \ _i} \sum_{t=0}^{\infty} x_t^T (Q + K^T RK) x_t$$

$$= \mathbf{trace}(P_K X_0)$$

where P_K solves *generalized* Lyapunov equation

$$P_K = Q + K^T RK + (A + BK)^T P_K (A + BK)$$

$$+ \sum_{i=1}^{p} \alpha_i A_i^T P_K A_i + \sum_{j=1}^{q} \ \ _j K^T B_j^T P_K B_j K$$

policy gradient algorithm

Lemma (policy gradient expression)

$$\nabla C(K) = 2\left[\left(R + B^T P_K B + \sum_{j=1}^{q} \beta_j B_j^T P_K B_j\right) K + B^T P_K A\right] \Sigma_K$$

where $\Sigma_K = \mathbf{E}\sum_{t=0}^{\infty} x_t x_t^T$ solves *generalized* Lyapunov equation

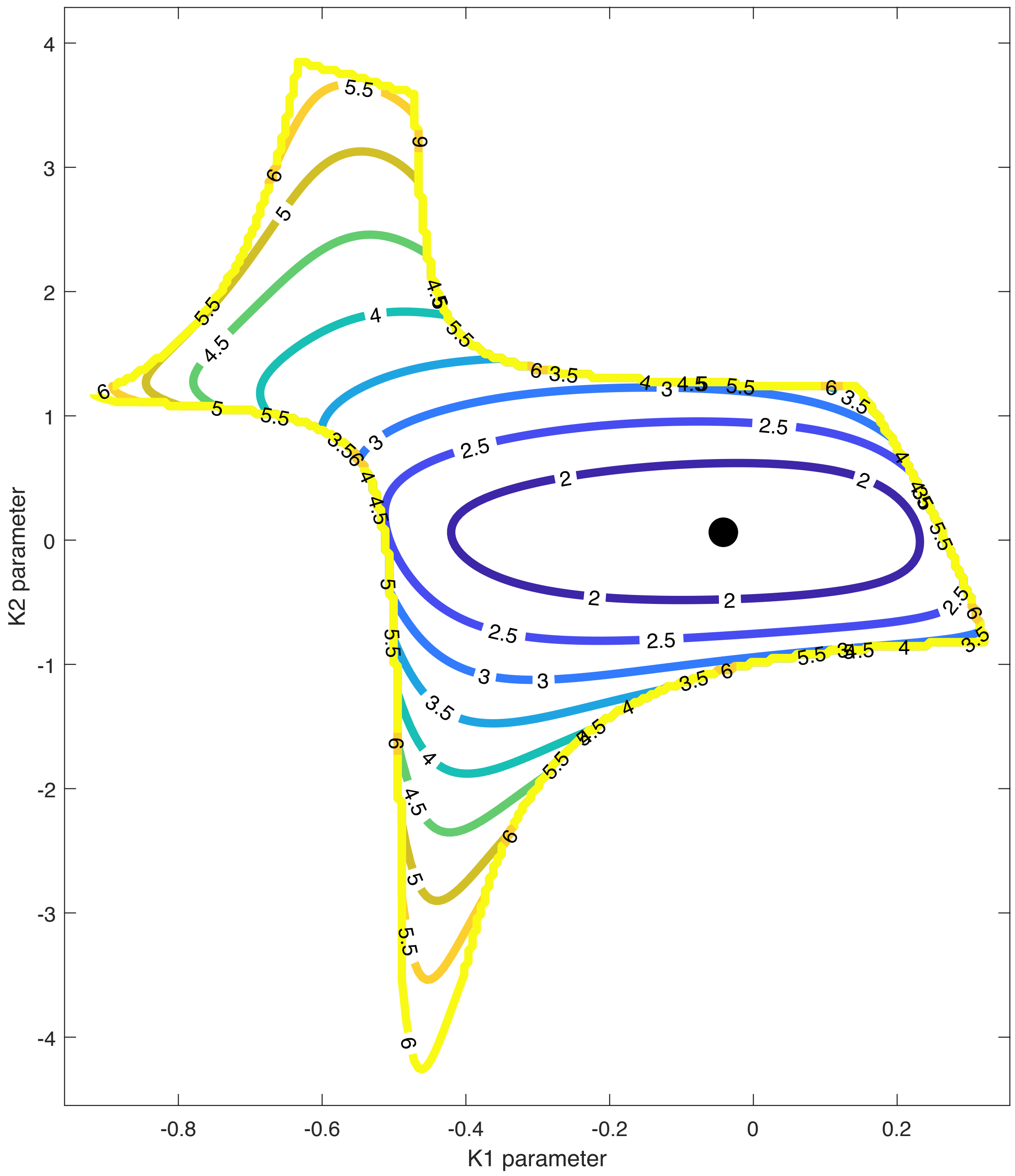
$$\Sigma_K = X_0 + (A+BK)\Sigma_K(A+BK)^T + \sum_{i=1}^{p} \alpha_i A_i \Sigma_K A_i^T + \sum_{j=1}^{q} \quad {}_j B_j K \Sigma_K K^T B_j^T$$

Algorithm (policy gradient)

$$K \qquad K \quad \eta\nabla C(K)$$

one small problem…

non-convexity of $C(K)$



gradient domination

Definition (gradient domination)

A function $f : \mathbf{R}^n \to \mathbf{R}$ is called *gradient dominated* if

$$\exists\lambda > 0 : \qquad \|\nabla f(x)\|^2 \qquad \lambda(f(x) \qquad f(x^*)) \quad \forall x \in \mathbf{R}^n$$

- gradient grows faster than quadratic away from optimal value
- every stationary point is a global minimum
- includes many non-convex functions
- *easy to prove gradient descent converges globally*

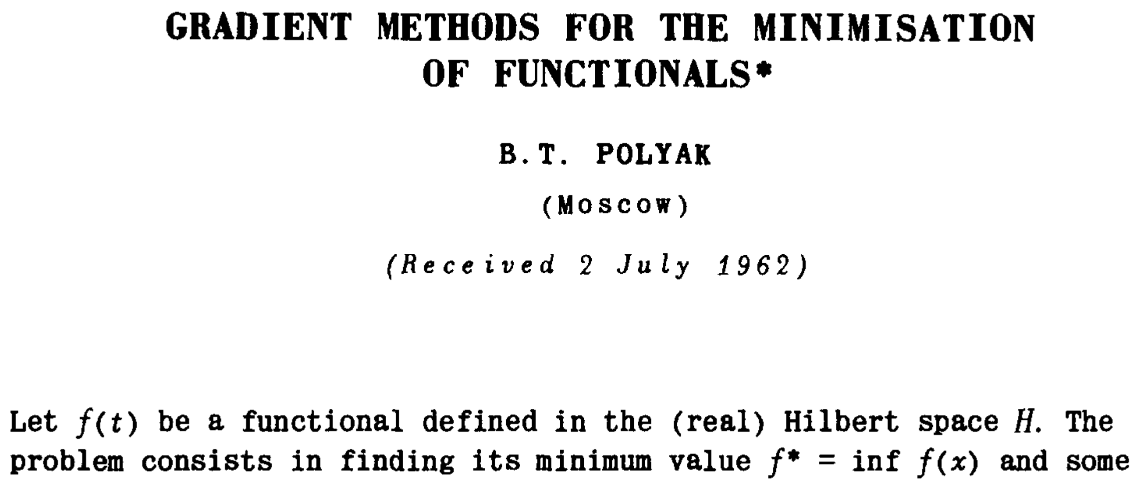
gradient domination

Definition (gradient domination)

A function $f : \mathbf{R}^n \to \mathbf{R}$ is called *gradient dominated* if

$$\exists \lambda > 0 : \quad \|\nabla f(x)\|^2 \quad \lambda(f(x) \quad f(x^*)) \quad \forall x \in \mathbf{R}^n$$

- also an old topic!



GRADIENT METHODS FOR THE MINIMISATION OF FUNCTIONALS*

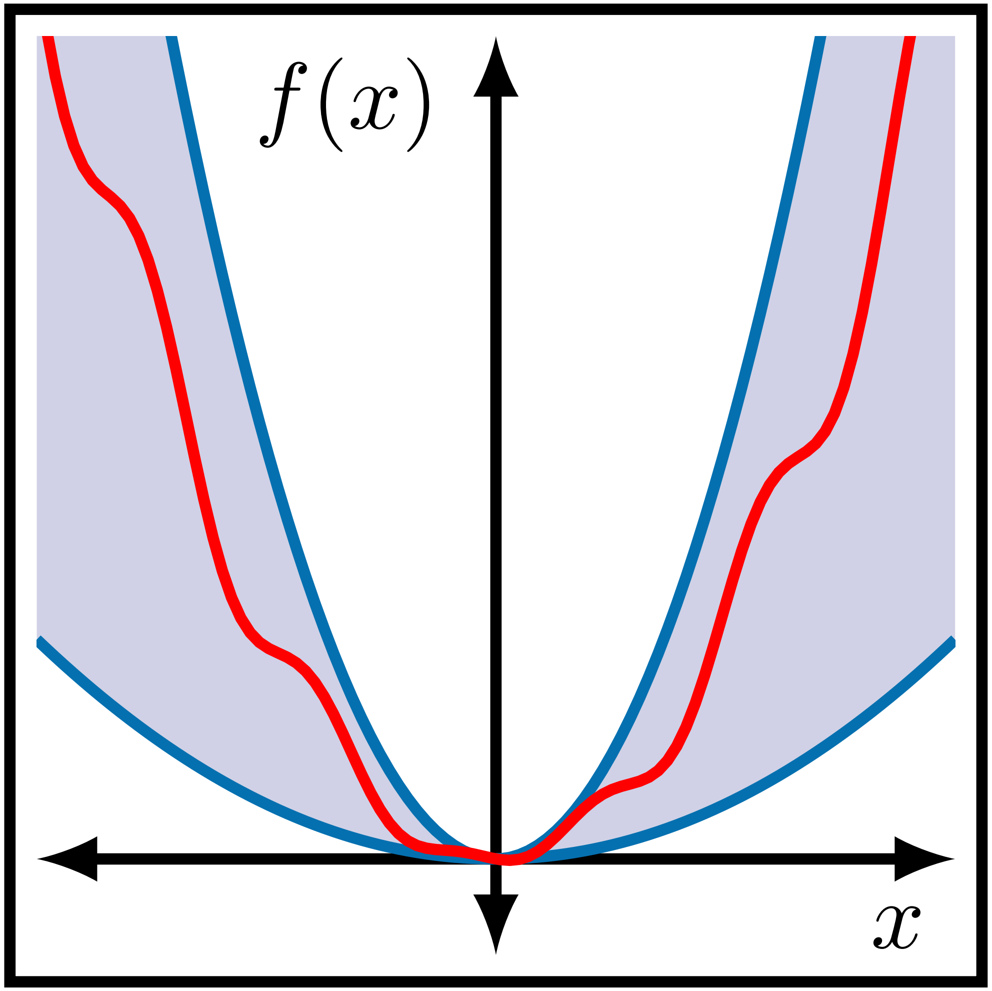
B.T. POLYAK

(Moscow)

(Received 2 July 1962)

Let $f(t)$ be a functional defined in the (real) Hilbert space H. The problem consists in finding its minimum value $f^* = \inf f(x)$ and some

gradient domination

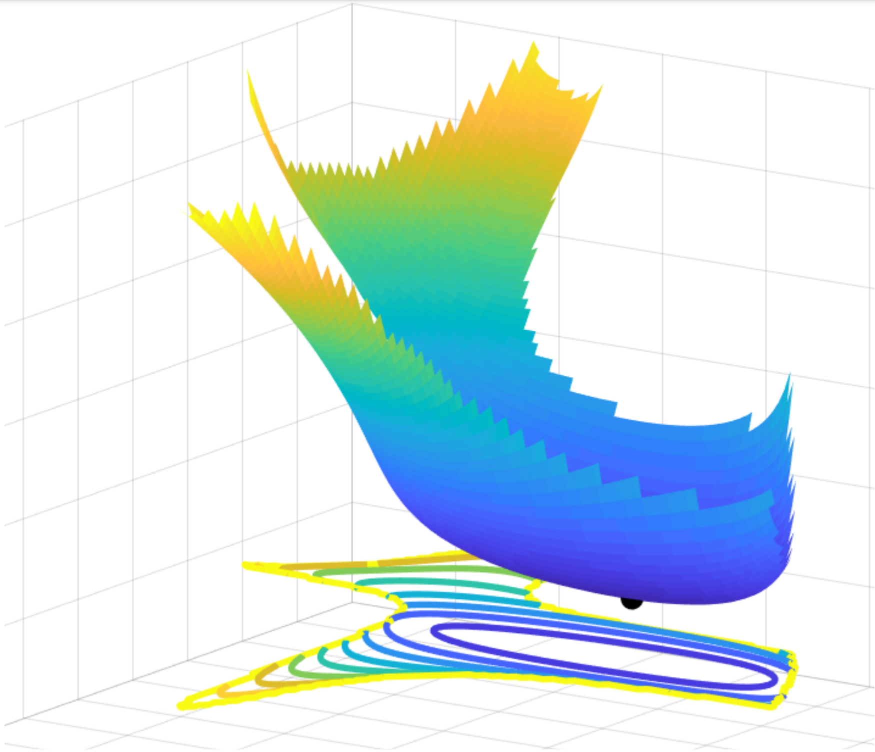


multiplicative LQR cost is gradient dominated

Theorem (Gravell + Summers 2019)

There exists $\lambda > 0$ such that

$$C(K) \quad C(K^*) \leq \lambda \|\nabla C(K)\|_F^2 \quad \forall K \in \mathrm{dom} C$$



- generalizes recent deterministic result (Fazel et al. 2018)

policy gradient converges globally

Algorithm (policy gradient)

$$K_{s+1} = K_s \quad \eta \nabla C(K_s)$$

Theorem (Gravell + Summers 2019)

For any initial gain $K_0 \in \mathrm{dom} C$ there is a constant step size $\eta(K_0, \text{problem data})$ and linear rate $\rho < 1$ such that

$$C(K_{s+1}) \quad C(K^*) \leq \rho(C(K_s) \quad C(K^*))$$

- generalizes recent deterministic result (Fazel et al. 2018)

policy iteration (a.k.a. Gauss-Newton)

Algorithm (policy iteration)

$$K_{s+1} = K_s \quad \eta R_{K_s}^{-1} \nabla C(K_s) \Sigma_{K_s}^{-1}$$

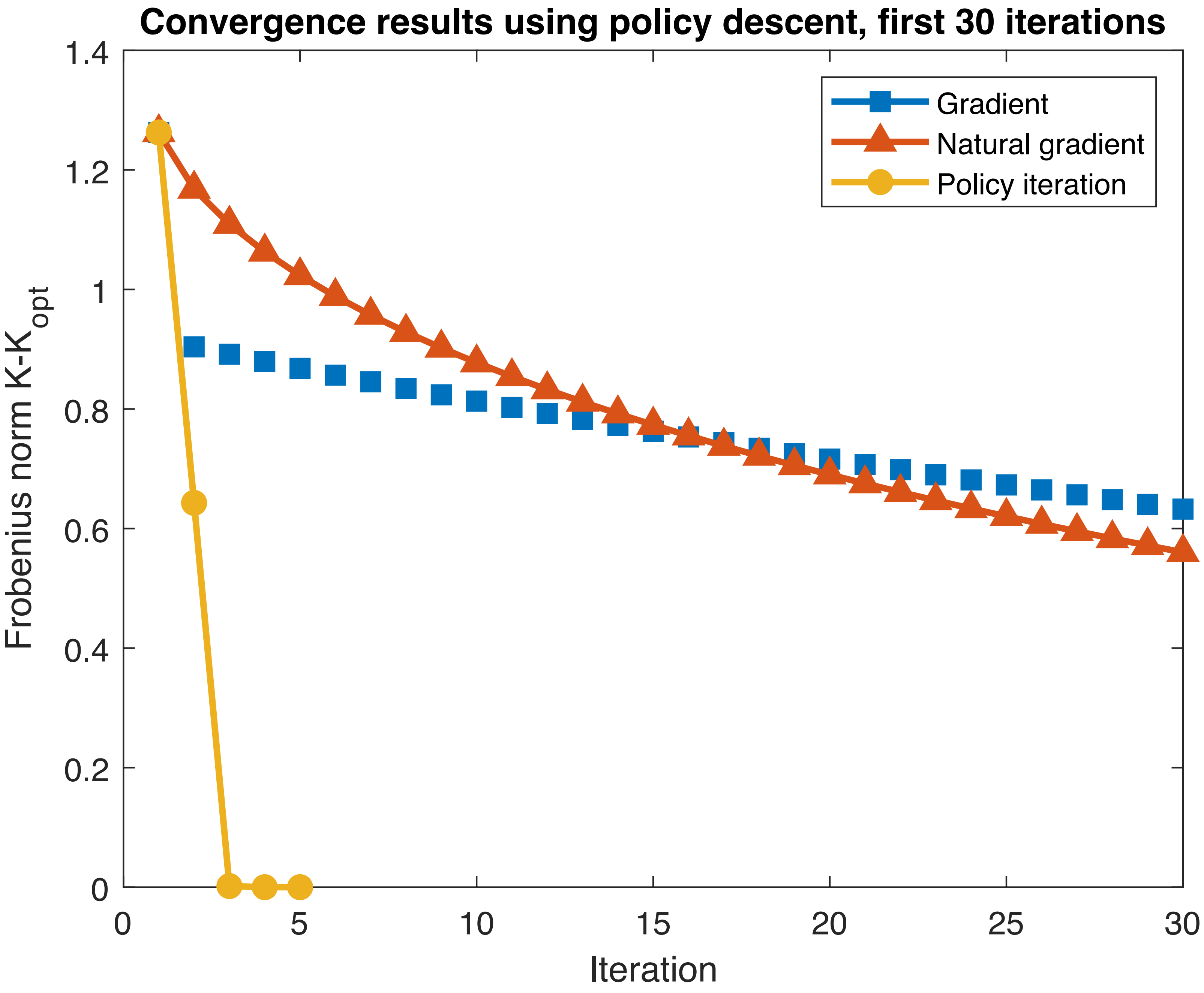
Theorem (Gravell + Summers 2019)

For any initial gain $K_0 \in \mathrm{dom} C$ with a constant step size $\eta = 1/2$ there is a (faster) linear rate $\rho < 1$ such that

$$C(K_{s+1}) \quad C(K^*) \leq \rho(C(K_s) \quad C(K^*))$$

- essentially a Newton algorithm (in function space)

policy iteration far faster than policy gradient



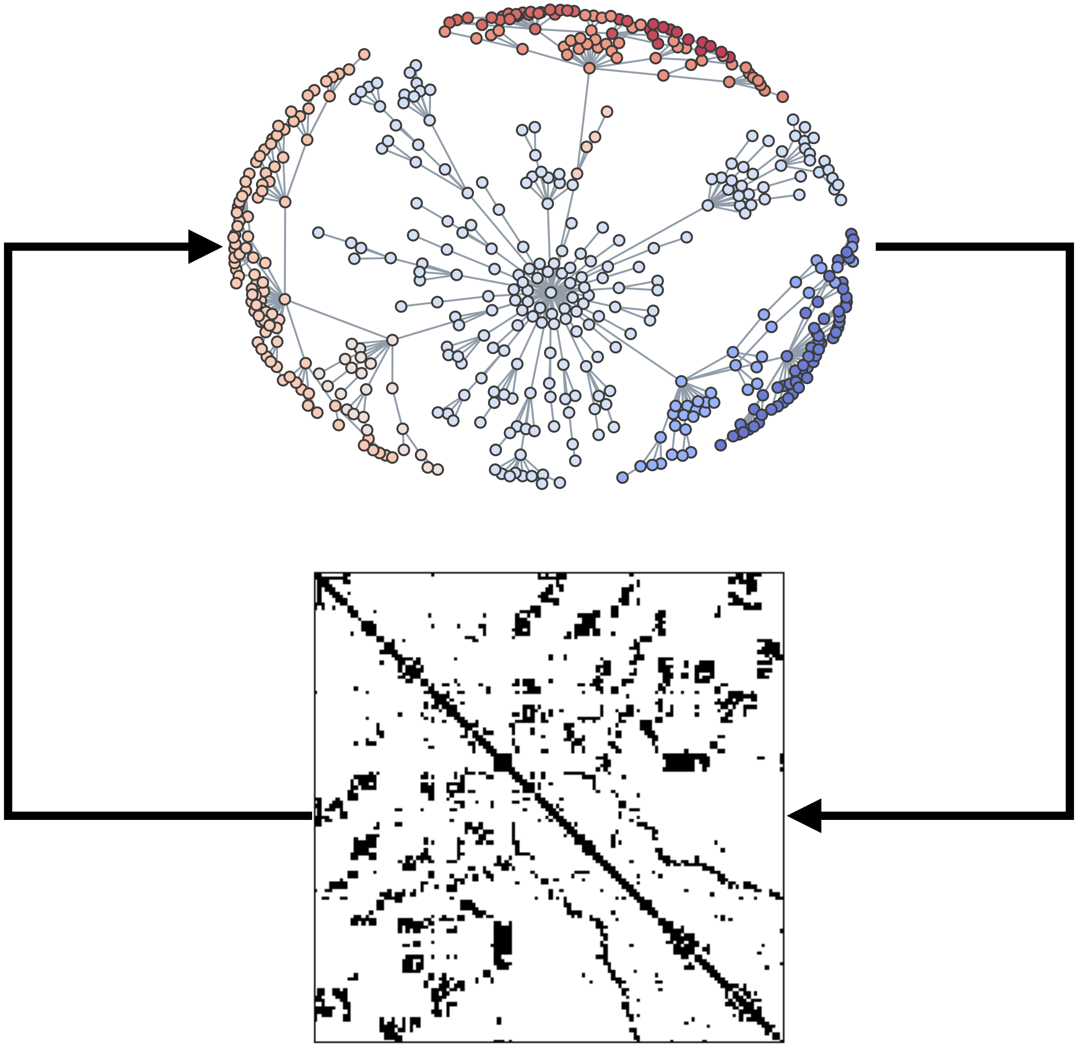
multiplicative noise awareness can be crucial for good control performance

	noise-aware control	noise-ignorant control
noisy system	484	∞
noise-free system	27.5	26.3
gain matrix	$\left[\begin{matrix} 2.26 & -0.04 \\ -0.04 & 0.004 \end{matrix} \right.$	$\left[\begin{matrix} 2.62 & 0.06 \\ -0.001 & 0.00 \end{matrix} \right.$

LQR cost

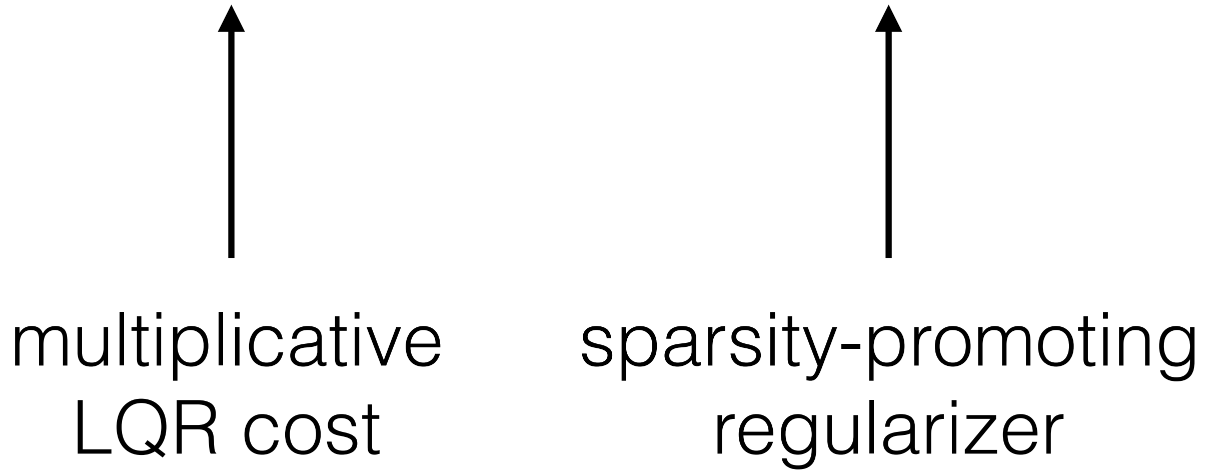
- failing to account for multiplicative noise can destabilize an otherwise (mean square) stable system!

sparse control architecture design



sparse gain design

$$\text{minimize} \quad C(K) \quad + \quad \gamma g(K)$$



multiplicative LQR cost

sparsity-promoting regularizer

$\gamma > 0$ trades off performance and controller sparsity

cf. Lin, Fardad, Jovanovic, Dörfler, et al. for additive case

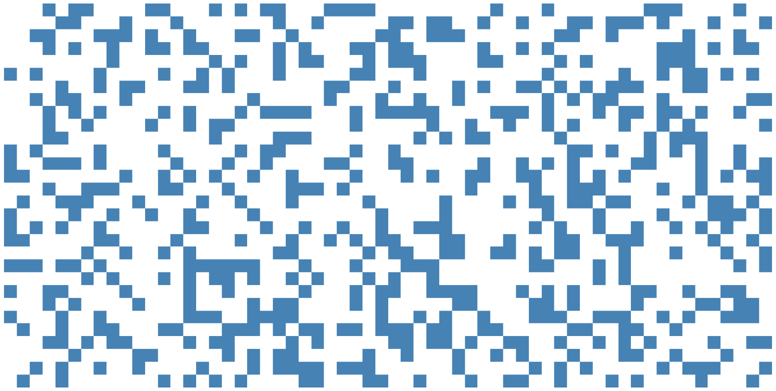
regularizer flavors

$g(K)$

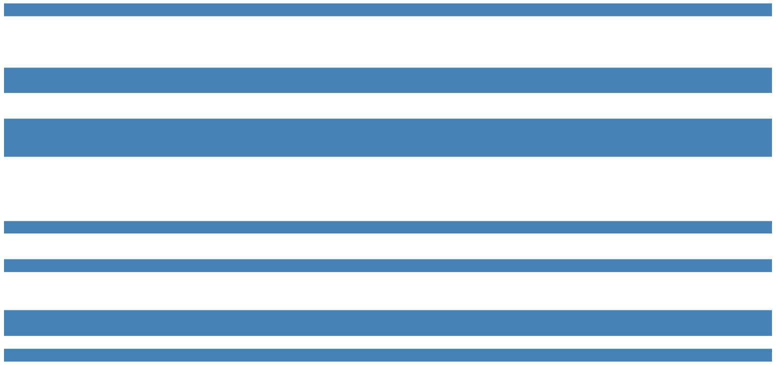
$$\|\mathrm{vec}(K)\|_1$$

$$\sum_{i=1}^{G} \|[K]_i\|_2$$

K



communication
architecture design

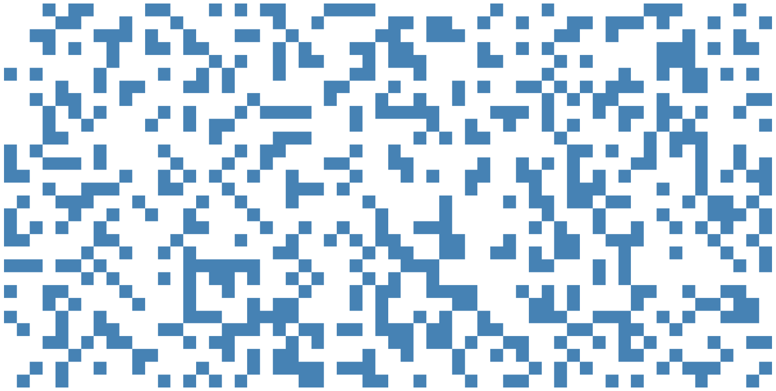


sensor & actuator
selection

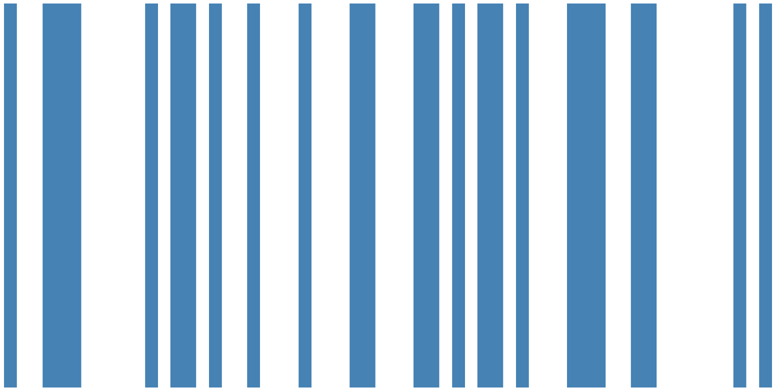
regularizer flavors

$$\|\text{vec}(K)\|_1$$

$$\sum_{i=1}^{G} \|[K]_i\|_2$$



communication architecture design

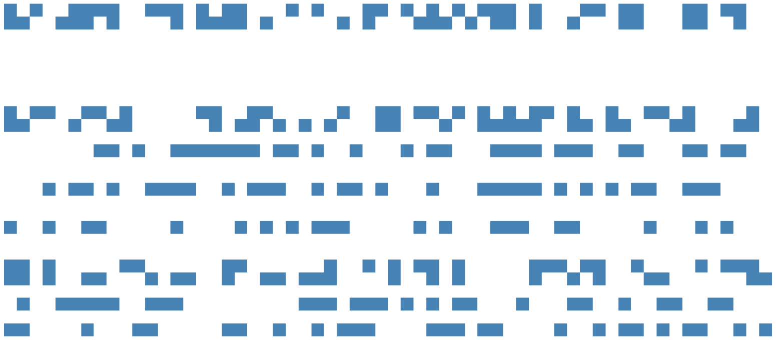


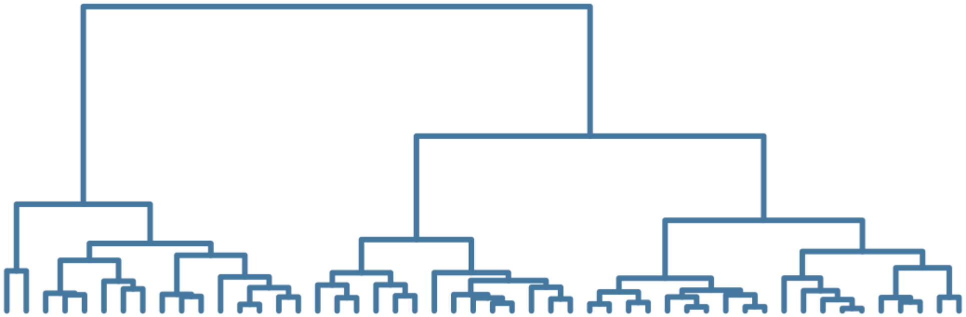
regularizer flavors

$$\mu\|\mathrm{vec}(K)\|_1 + (1 \quad \mu)\sum_{i=1}^{G}\|[K]_i\|_2$$

$$\sum_{i=1}^{G}\|[K]_i\|_2$$

sparse group LASSO
with group overlap





+

architecture design
with logical constraints

proximal algorithms

$$\text{minimize} \qquad \underset{\substack{\uparrow \\ \text{smooth}}}{C(K)} \qquad + \qquad \underset{\substack{\uparrow \\ \text{non-smooth}}}{\gamma g(K)}$$

- proximal/operator splitting methods
- nice review by Parikh + Boyd 2014, nice recent work by Jovanovic et al.

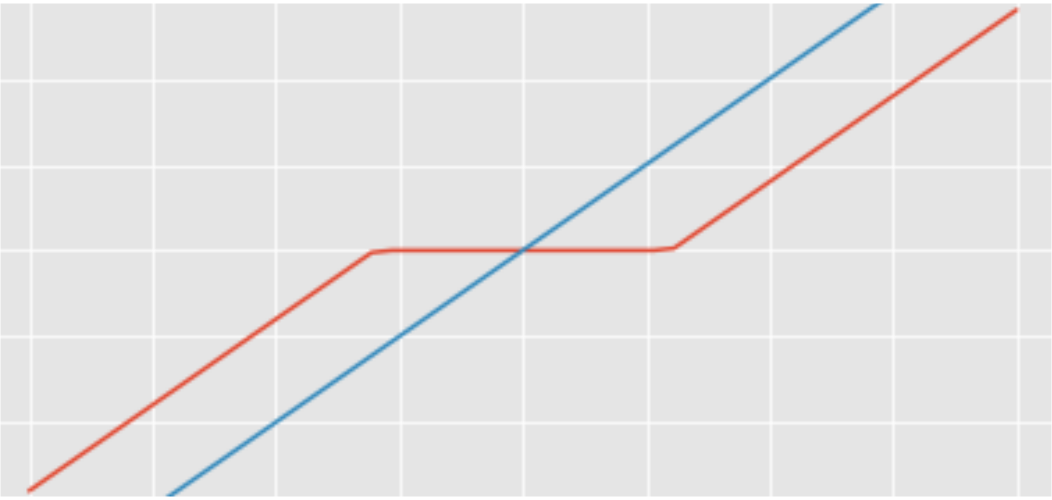
Algorithm (proximal gradient)

$$K \qquad \mathbf{prox}_{\eta\gamma g}(K \qquad \eta\nabla C(K))$$

proximal operators

- some proximal operators doable in closed form: e.g., for $\|\mathrm{vec}(K)\|_1$ (elementwise) *soft thresholding*

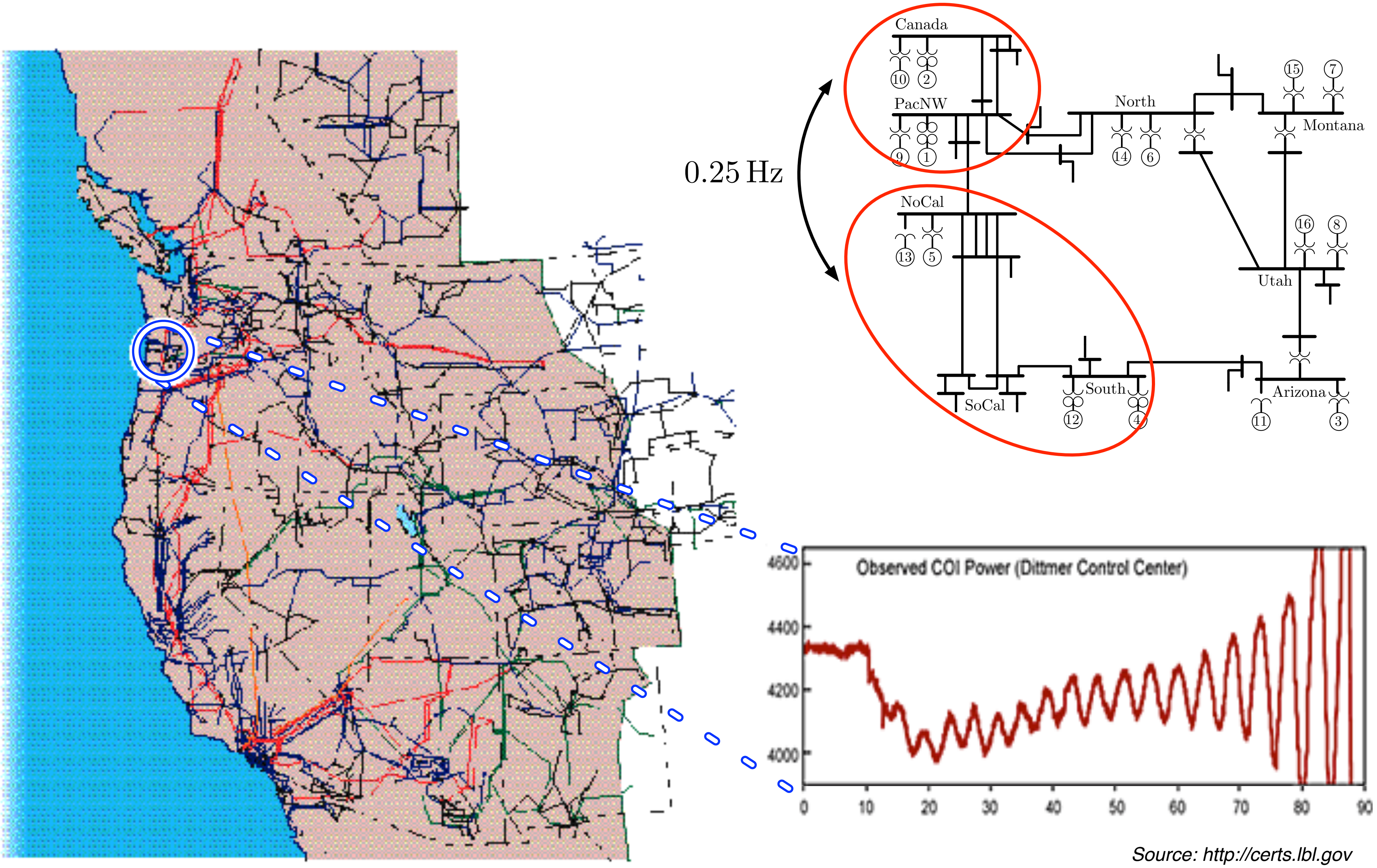
$$\mathbf{prox}_{\eta\gamma g}(K) = (K - \eta\gamma)_+ - (-K + \eta\gamma)_+$$



- others can be evaluated efficiently

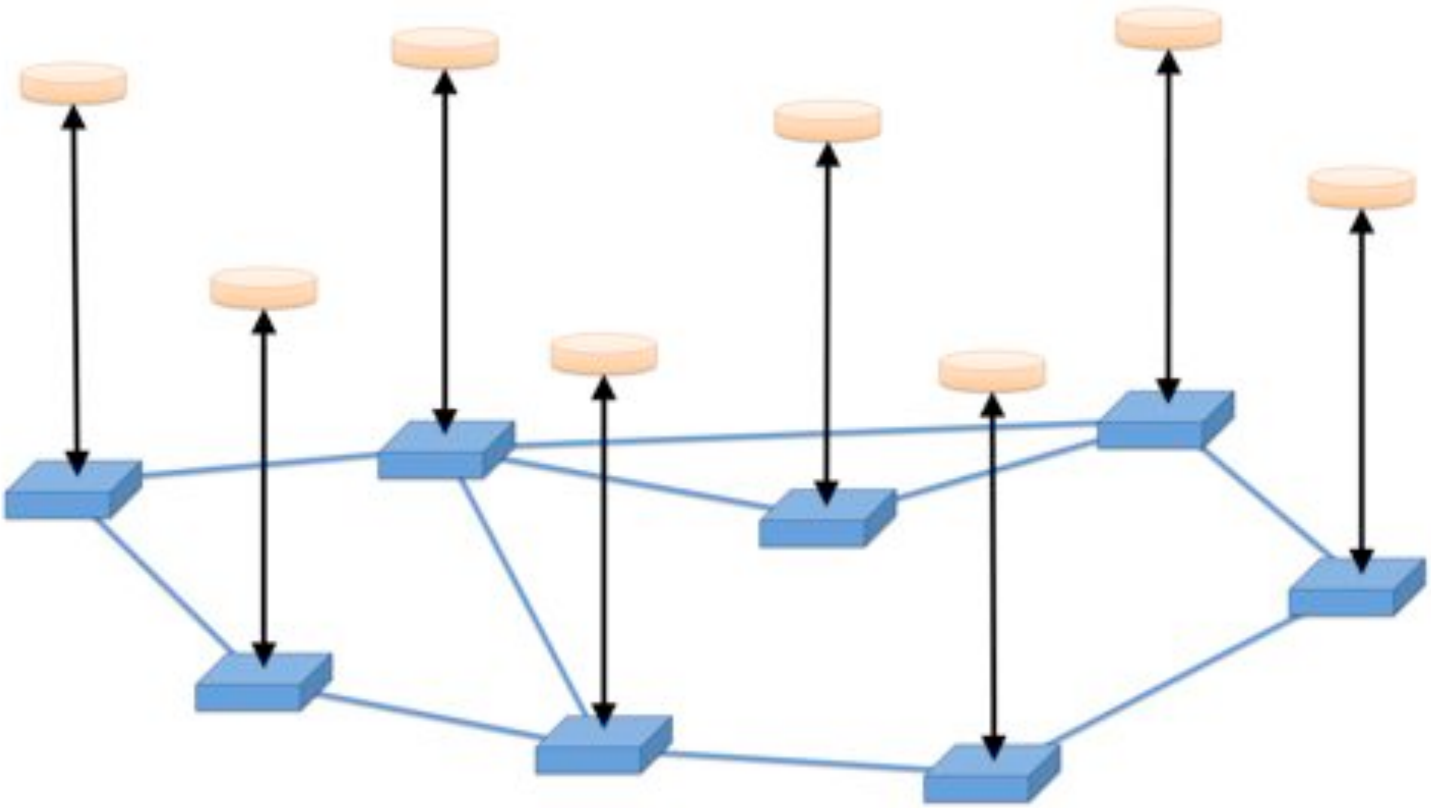
wide area control of power networks

8/10/96 blackout due to instability of 0.25Hz inter-area mode



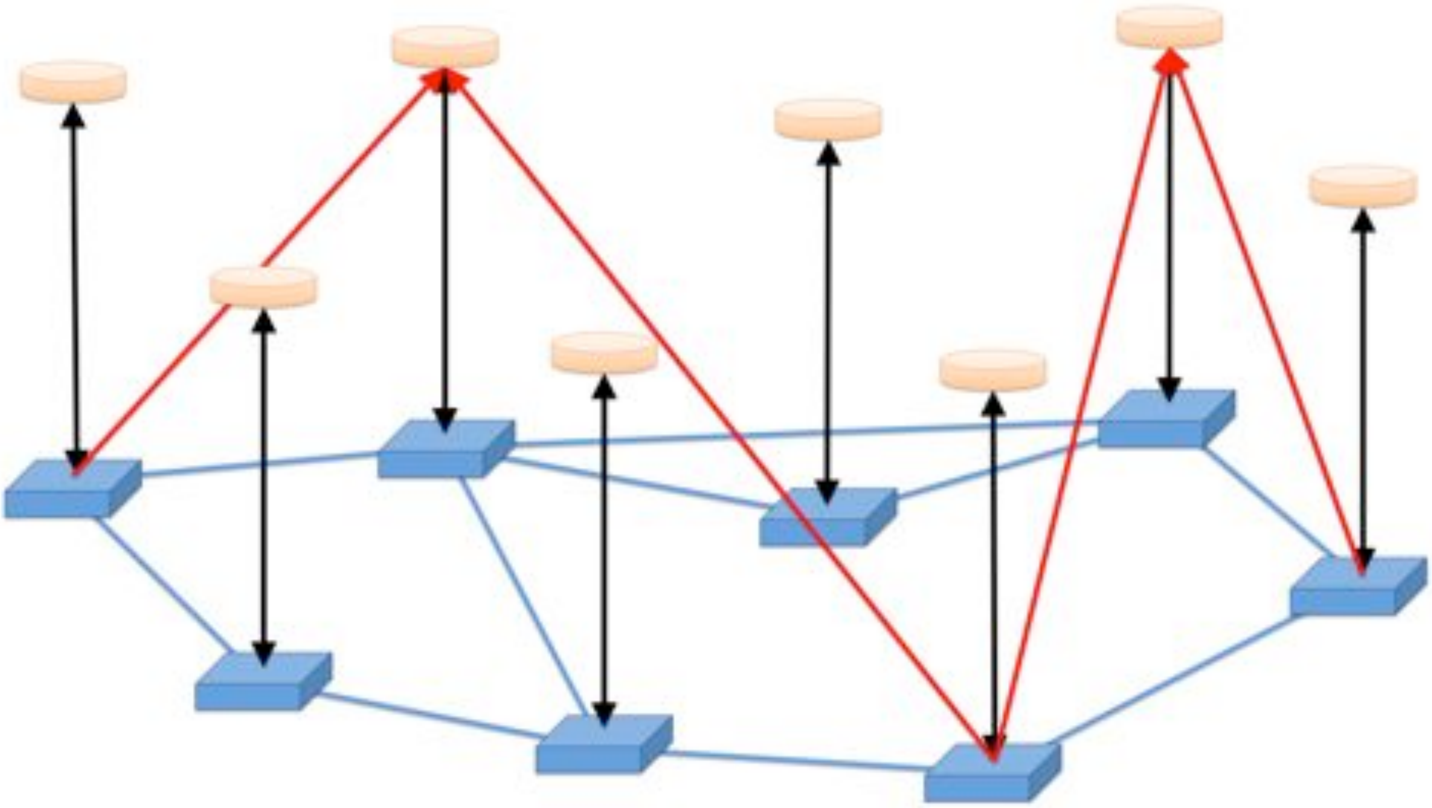
Source: http://certs.lbl.gov

wide area control of power networks



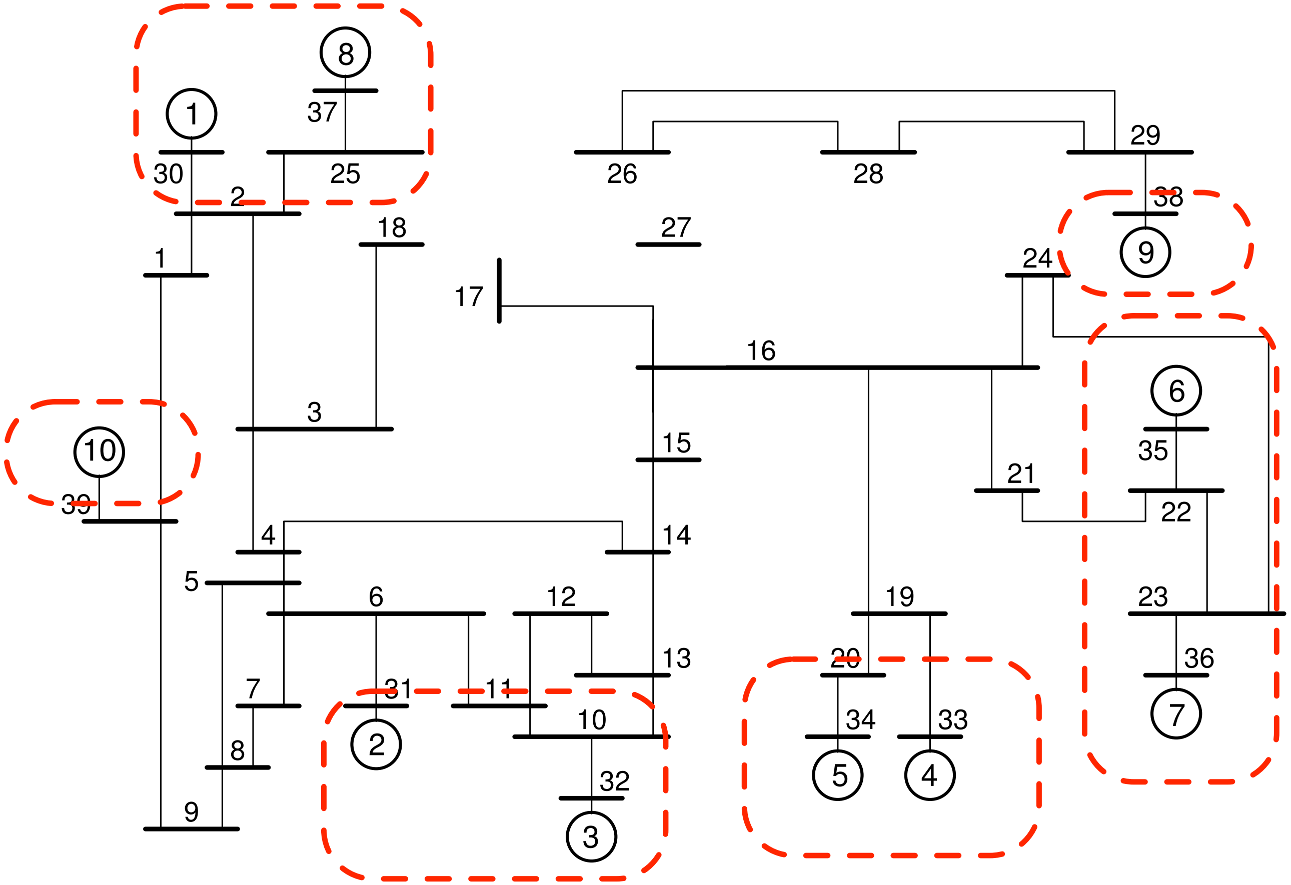
conventional primary frequency vs. distributed wide area control

wide area control of power networks



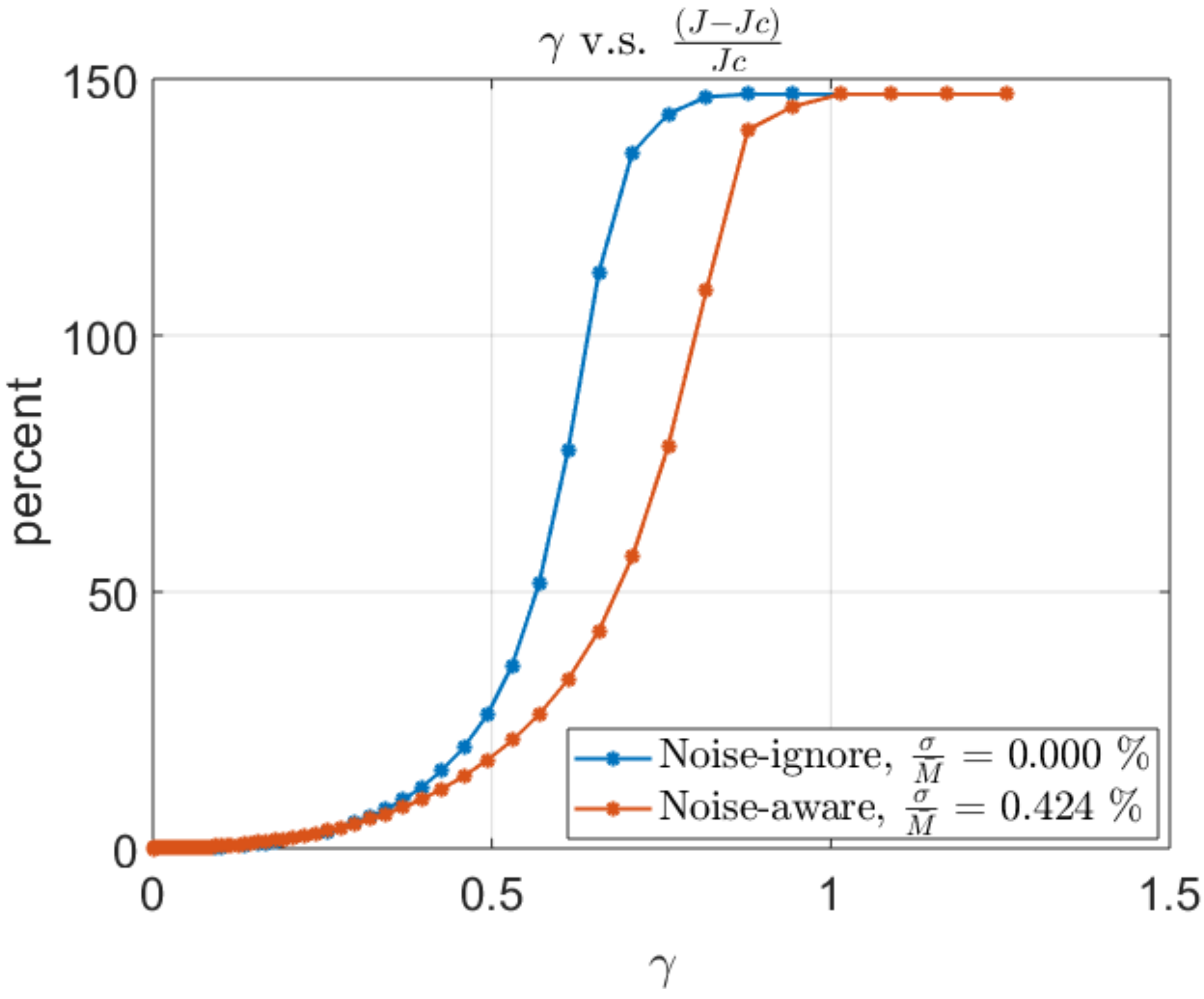
conventional primary frequency vs. distributed wide area control

case study: IEEE 39 bus test network



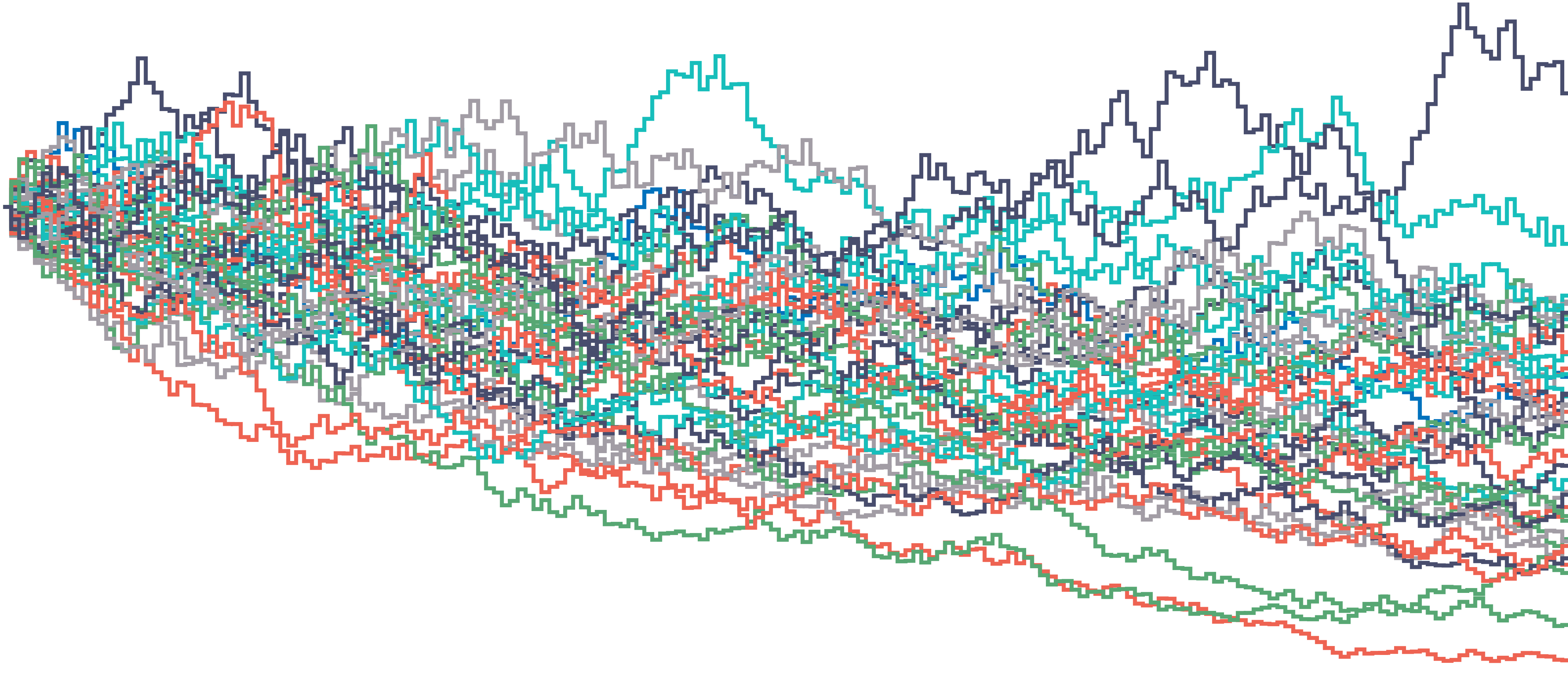
- linearized swing dynamics, with multiplicative inertia noise
- quadratic network coherency performance metric to penalize relative angle differences and frequency deviations
- proximal policy iteration for **noise-aware** architecture design

noise-aware architecture design



significantly lower cost with noise-aware architecture

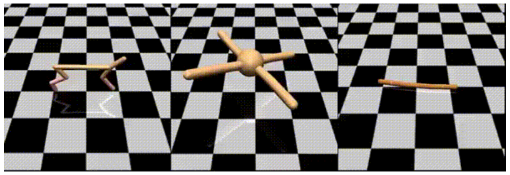
data-driven control



data-driven, learning-based control

- recent undeniable successes, along with massively (over?)hyped efforts in ML/RL/AI + control

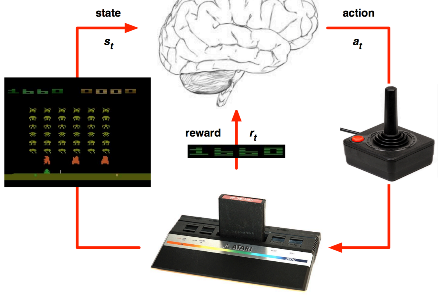
How research funders profit from hidden investments
New books for budding scientists
Drug leads for malaria
Science
AAAS
A DIGITAL PRODIGY
AlphaZero teaches itself chess, shogi, and Go

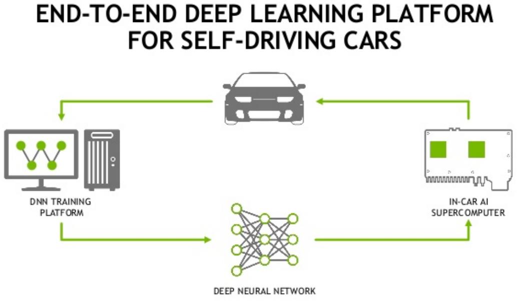


Deep Drone Racing: Learning Agile Flight in Dynamic Environments

Elia Kaufmann* University of Zurich · Antonio Loquercio* University of Zurich · Rene Ranftl Intel Labs

Alexey Dosovitskiy Intel Labs · Vladlen Koltun Intel Labs · Davide Scaramuzza University of Zurich

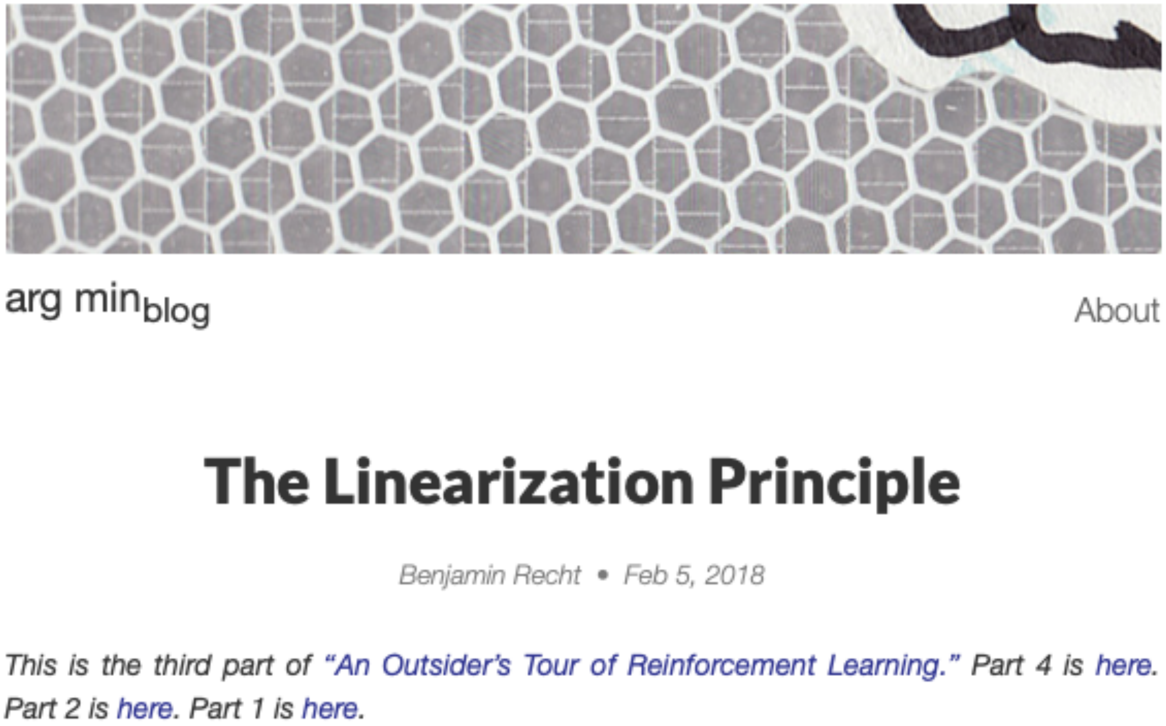




END-TO-END DEEP LEARNING PLATFORM FOR SELF-DRIVING CARS

- unclear if/how these can be transferred to large, complex, safety critical systems
- false dichotomy: model-free vs. model-based
  - how best to *combine?*
- despite long history, still many fundamental open problems, opportunities to exploit spectacular modern computation and data resources

linearization principle



arg min$_{\text{blog}}$

About

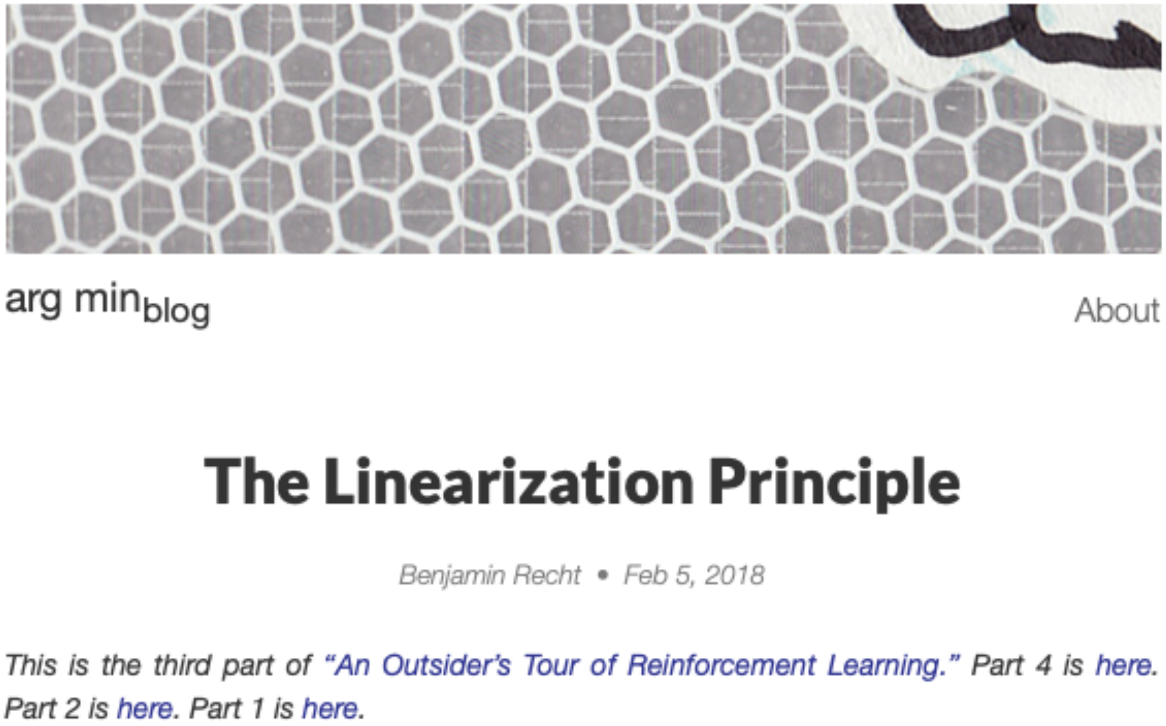
The Linearization Principle

Benjamin Recht • Feb 5, 2018

This is the third part of "An Outsider's Tour of Reinforcement Learning." Part 4 is here. Part 2 is here. Part 1 is here.

"If a machine learning algorithm does crazy things when restricted to linear models, it's going to do crazy things on complex nonlinear models too"

linearization principle



arg min$_{\text{blog}}$

About

The Linearization Principle

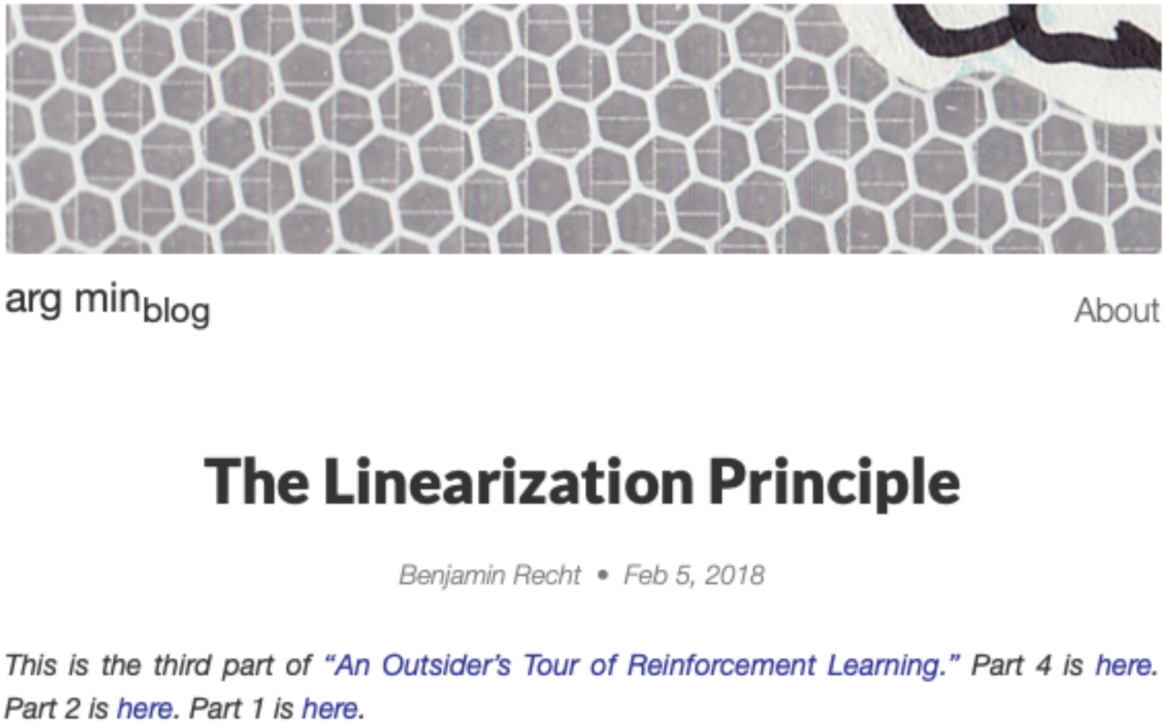
Benjamin Recht • Feb 5, 2018

This is the third part of "An Outsider's Tour of Reinforcement Learning." Part 4 is here. Part 2 is here. Part 1 is here.

The New York Times

"What is frustrating about machine learning is that the algorithms can't articulate what they're thinking. We don't know why they work, so we don't know if they can be trusted… As human beings, we want more than answers. We want **insight**. This is going to be a source of tension in our interactions with computers from now on."

linearization principle



arg min$_{\text{blog}}$

About

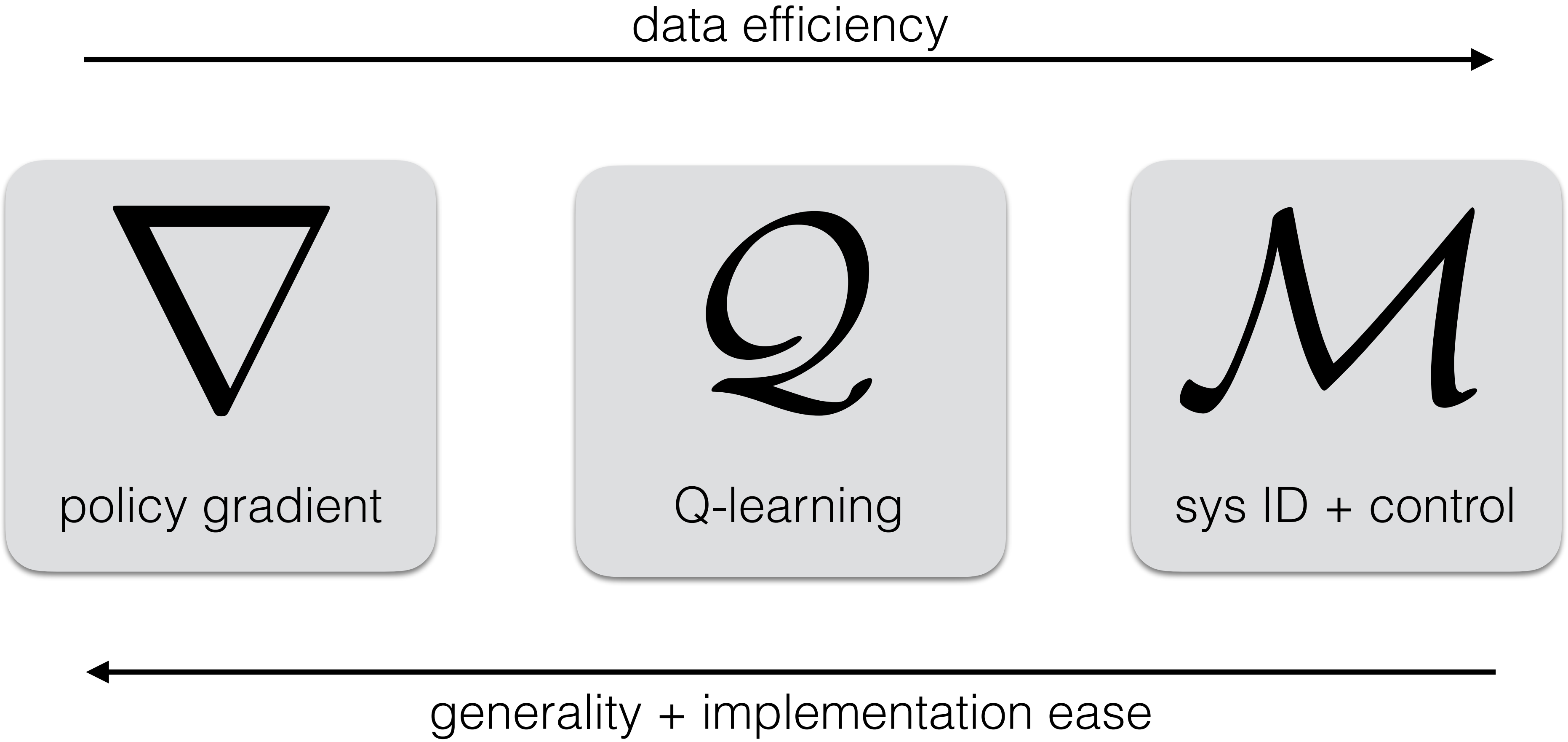
The Linearization Principle

Benjamin Recht • Feb 5, 2018

This is the third part of "An Outsider's Tour of Reinforcement Learning." Part 4 is here. Part 2 is here. Part 1 is here.

- LQR as an important theoretical benchmark
- multiplicative noise LQR as an interesting elaboration that explicitly incorporates model uncertainty, robustness issues

data efficiency trade offs



data efficiency

∇

policy gradient

$\mathcal{Q}$

Q-learning

$\mathcal{M}$

sys ID + control

generality + implementation ease

“model-free” policy gradient

```
**Algorithm** (“model-free” policy gradient)
    $K \qquad K \qquad \eta\widehat{\nabla C(K)}$
```

- can estimate policy gradient via zeroth order optimization

```
**Algorithm** (“model-free” policy gradient estimate)
for $i = 1, ..., m$    ($m$ trajectory rollouts, length $l$)
    sample policy: $\hat{K}_i = K + U_i, \quad U_i \sim U_{\|\cdot\|_F, r}$
    simulate $l$ steps, record empirical costs $\hat{C}_i = \sum_{t=1}^{l} c_t$
end for
return estimate $\widehat{\nabla C(K)} = \frac{1}{m} \sum_{i=1}^{m} \frac{n}{r^2} \hat{C}_i U_i$
```

“model-free” policy gradient converges globally for multiplicative noise LQR

Algorithm (“model-free” policy gradient)

$$K \qquad K \qquad \eta\widehat{\nabla C(K)}$$

Theorem (Gravell + Summers 2019)

For any initial stabilizing gain, there is an exploration radius, a number and length of trajectory rollouts of polynomial size in problem data, and constant step size such that gradient descent converges globally to the optimal gain matrix.

- multiplicative noise generalization of Fazel et al.
- i.e., policy gradient provably “works”
- however, extremely data inefficient

policy iteration via Q-learning

Algorithm (“model-free” policy iteration via Q-learning)

$$K_{s+1} = K_s \quad \eta R_{K_s}^{-1} \nabla C(K_s) \Sigma_{K_s}^{-1}$$

- update equivalent to $K_{s+1} = -\mathcal{Q}_{uu}^{-1} \mathcal{Q}_{xu}^T$ where

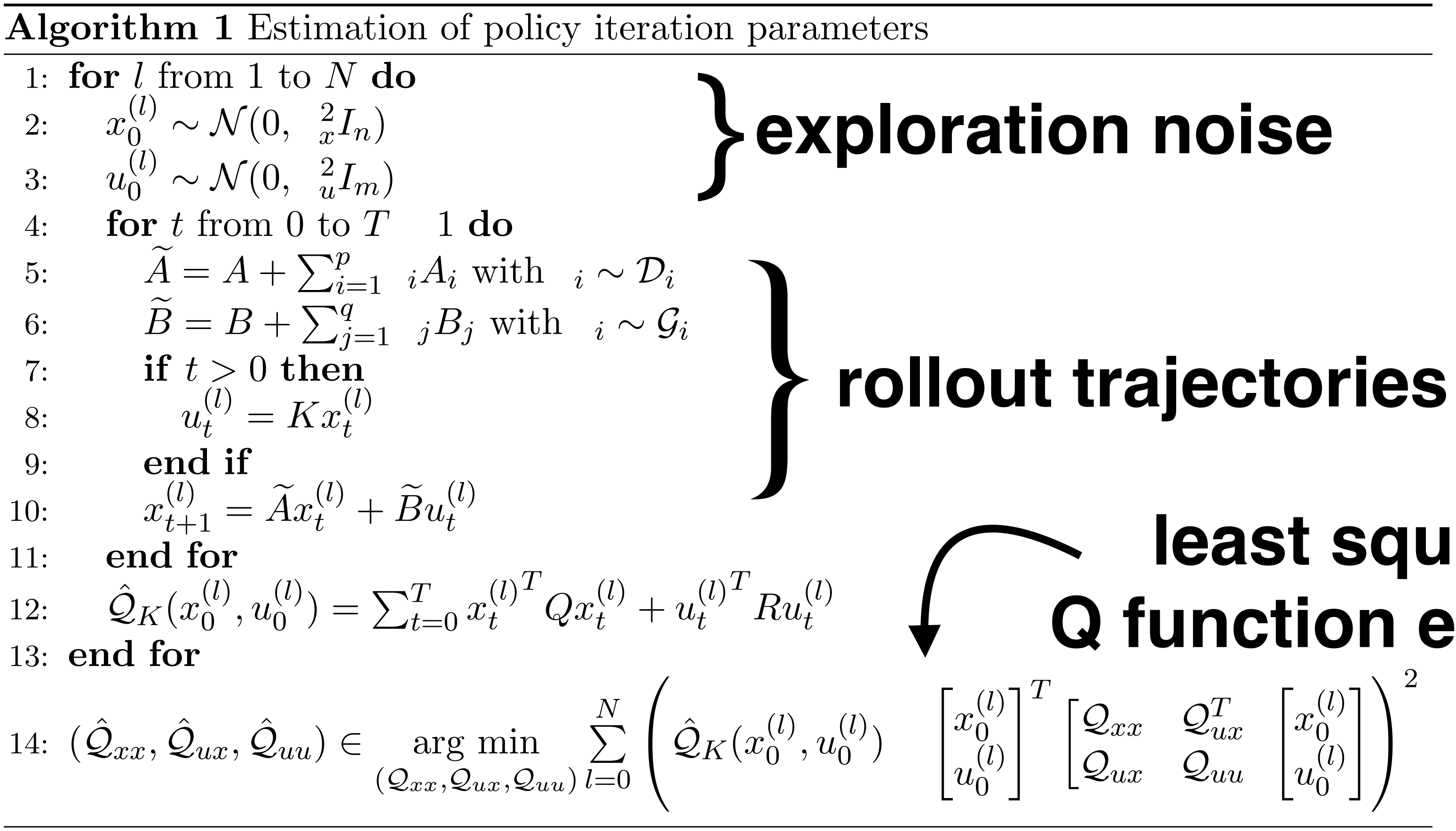
$$\mathcal{Q}_{K_s}(x,u) = \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} Q + A^T P_{K_s} A + \sum_{i=1}^p \alpha_i A_i^T P_{K_s} A_i & A^T P_{K_s} B \\ B^T P_{K_s} A & R + B^T P_K B + \sum_{j=1}^q \; {}_j B_j^T P_{K_s} B_j \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}$$

- just need Q-function estimate to implement policy iteration!

policy iteration via Q-learning

Algorithm (“model-free” policy iteration via Q-learning)

$$K_{s+1} = -\hat{\mathcal{Q}}_{uu}^{-1}\hat{\mathcal{Q}}_{xu}^T$$



Algorithm 1 Estimation of policy iteration parameters

```
for $l$ from 1 to $N$ do
    $x_0^{(l)} \sim \mathcal{N}(0, \ {}_x^2 I_n)$
    $u_0^{(l)} \sim \mathcal{N}(0, \ {}_u^2 I_m)$
    for $t$ from 0 to $T \ \ 1$ do
        $\widetilde{A} = A + \sum_{i=1}^{p} \ {}_i A_i$ with $\ {}_i \sim \mathcal{D}_i$
        $\widetilde{B} = B + \sum_{j=1}^{q} \ {}_j B_j$ with $\ {}_i \sim \mathcal{G}_i$
        if $t > 0$ then
            $u_t^{(l)} = K x_t^{(l)}$
        end if
        $x_{t+1}^{(l)} = \widetilde{A} x_t^{(l)} + \widetilde{B} u_t^{(l)}$
    end for
    $\hat{\mathcal{Q}}_K(x_0^{(l)}, u_0^{(l)}) = \sum_{t=0}^{T} x_t^{(l)^T} Q x_t^{(l)} + u_t^{(l)^T} R u_t^{(l)}$
end for
$(\hat{\mathcal{Q}}_{xx}, \hat{\mathcal{Q}}_{ux}, \hat{\mathcal{Q}}_{uu}) \in \underset{(\mathcal{Q}_{xx}, \mathcal{Q}_{ux}, \mathcal{Q}_{uu})}{\arg\min} \sum_{l=0}^{N} \left( \hat{\mathcal{Q}}_K(x_0^{(l)}, u_0^{(l)}) \quad \begin{bmatrix} x_0^{(l)} \\ u_0^{(l)} \end{bmatrix}^T \begin{bmatrix} \mathcal{Q}_{xx} & \mathcal{Q}_{ux}^T \\ \mathcal{Q}_{ux} & \mathcal{Q}_{uu} \end{bmatrix} \begin{bmatrix} x_0^{(l)} \\ u_0^{(l)} \end{bmatrix} \right)^2$
```

exploration noise

rollout trajectories

least squares Q function estimate

policy iteration via Q-learning converges globally for multiplicative noise LQR

Algorithm (“model-free” policy iteration via Q-learning)

$$K_{s+1} = -\hat{\mathcal{Q}}_{uu}^{-1} \hat{\mathcal{Q}}_{xu}^{T}$$

- multiplicative noise generalization of BradtkeYdstieBarto 1994
- i.e., policy iteration via Q-learning provably “works”
- however, somewhat data inefficient

sys ID + model-based control

- traditional alternative: system ID + (robust) model-based control
  - cf. recent sample complexity results from “coarse ID” framework of Dean, Mania, Matni, Recht, Tu

- how to estimate *multiplicative noise model* from data?

$$x_{t+1} = \quad A + \sum_{i=1}^{p} \Bigg({}_{ti} A_i \Bigg) \Bigg(x_t + \quad B + \sum_{i=1}^{q} \Bigg({}_{ti} B_i \Bigg) \Bigg(u_t$$

  - assume A_i, B_i given, want to estimate $A, B, \alpha_i, \quad {}_i$

sys ID + model-based control

- traditional alternative: system ID + (robust) model-based control

- how to estimate *multiplicative noise model* from data?
    - first moment (mean) dynamics $\mu_t = \mathbf{E} x_t$

$$\mu_{t+1} = A\mu_t + Bu_t$$

-
    - second moment dynamics $X_t = \mathbf{E} x_t x_t^T \quad U_t = \mathbf{E} u_t u_t^T$

$$X_{t+1} = AX_tA^T + BU_tB^T + \sum_{i=1}^{p} \alpha_i A_i X_t A_i^T + \sum_{j=1}^{q} \quad {}_j B_j U_t B_j^T$$

sys ID + model-based control

- two-stage least squares estimation algorithm

Stage 1

$$\{\hat{A}, \hat{B}\} = \underset{A,B}{\operatorname{argmin}} \left\{ \sum_{t=0}^{l} \quad \mu_{t+1} - (A\mu_t + Bu_t) \quad {}_2^2 \right\}$$

Stage 2

$$\{\hat{\alpha}, \hat{\ }\} = \underset{\alpha\geq 0,\ \ \geq 0}{\operatorname{argmin}} \left\{ \sum_{t=0}^{l} \quad X_{t+1} - (\hat{A}X_t\hat{A}^T + \hat{B}U_t\hat{B}^T + \sum_{i=1}^{p} \alpha_i A_i X_t A_i^T - \sum_{j=1}^{q} \ {}_j B_j U_t B_j^T \quad {}_2^2 \right\}$$

sys ID + model-based control

- two-stage least squares estimation algorithm
- using rollout trajectory data

Stage 1

$$\{\hat{A}, \hat{B}\} = \underset{A,B}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n_r} \sum_{t=0}^{l-1} \quad \bar{x}_{i_{t+1}}^{(l)} - (A\bar{x}_{i_t}^{(l)} + Bu_{i_t}^{(l)}) \quad {}_2^2 \right\}$$

Stage 2

$$\{\hat{\alpha}, \hat{\ }\} = \underset{\alpha \quad 0, \quad 0}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n_r}\sum_{j=1}^{n_p}\sum_{t=0}^{l \quad 1} \quad x_{i,j_{t+1}}^{(l)} x_{i,j_{t+1}}^{(l)}{}^T - (\hat{A}x_{i,j_t}^{(l)} + \hat{B}u_{i_t}^{(l)})(\hat{A}x_{i,j_t}^{(l)} + \hat{B}u_{i_t}^{(l)})^T \right.$$

$$\left. - \sum_{i=1}^{p} \alpha_i A_i x_{i,j_t}^{(l)} x_{i,j_t}^{(l)}{}^T A_i^T - \sum_{j=1}^{q} \quad {}_j B_j u_{i_t}^{(l)} u_{i_t}^{(l)}{}^T B_j^T \quad {}_2^2 \right\}$$

sys ID + model-based control: data efficiency

- with model estimates, simply use model-based design procedure to compute (approximate) optimal controller
  - or via robust variants

- preliminary numerical experiments indicate that sys ID + model-based control is far more data efficient than policy gradient/iteration for multiplicative noise models
  - at least when # of multiplicative noise terms is small

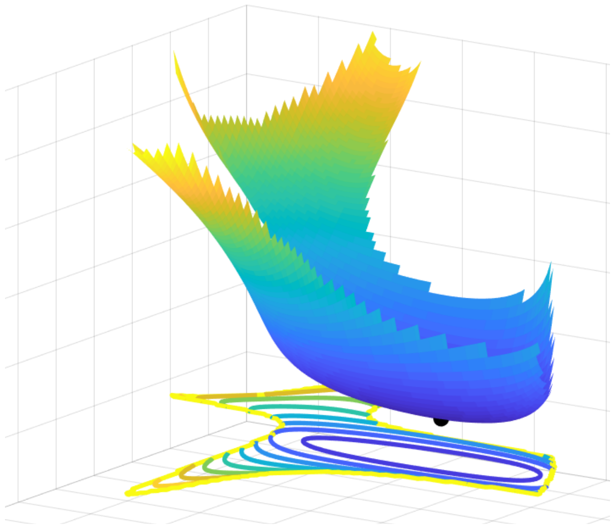
- working on theoretical non-asymptotic sample complexity results that generalize Recht et al. to multiplicative noise
  - however, data efficiency trade offs will take a different shape and may depend on # of multiplicative noise terms

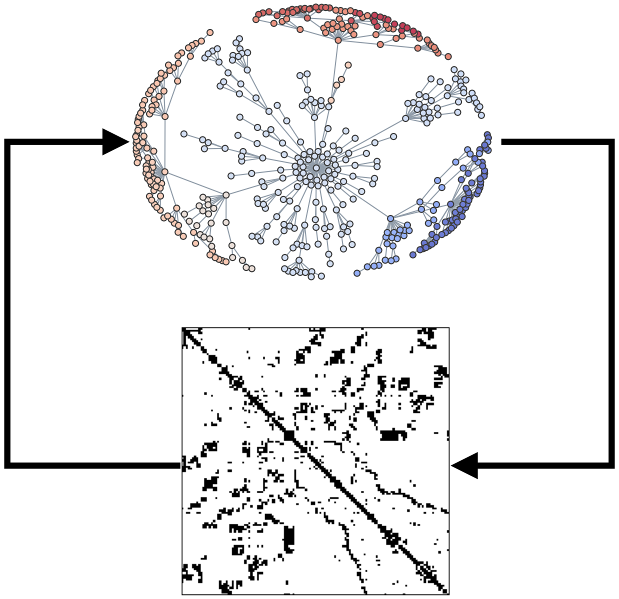
summary
&
conclusions

summary & conclusions

- multiplicative noise systems deserve more attention in context of highly distributed autonomous systems

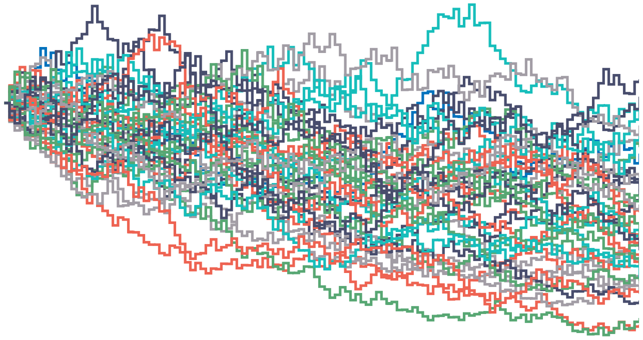
exact global convergence





control architecture design

data-driven control



- preliminary results here; a lot of exciting work ahead!

extensions and variations

ongoing and future work

- data-drive control architecture design
- (multiplicative noise) dynamic games
- continuous time (stochastic differential equations)
- state estimation
- accelerated and robust gradient methods
- static output feedback
- non-asymptotic sample complexity for sys ID + control
- regret analysis for adaptive control
- learning in nonlinear systems via local linear approximation
- applications, e.g.:
    - low- and variable-inertia power networks
    - networked autonomous robot teams
    - turbulent fluid flow
    - soft robots
    - neuronal brain networks

thank you!

questions?