ICML Tutorial:
Online & Non-stochastic control

PRINCETON UNIVERSITY

Google AI Princeton

Microsoft Research

Elad Hazan

Karan Singh

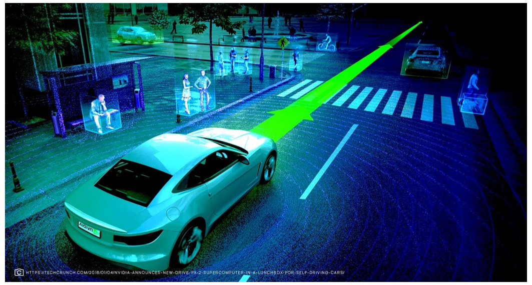
Tutorial materials

References (email us for more!)

+ slides, lecture notes, colab notebooks:
https://sites.google.com/view/nsc-tutorial/home

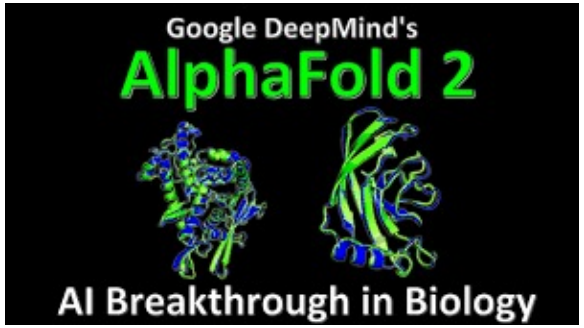
Deluca: more experiments, notebooks

Control vs. RL



ME/AE/EE

COS

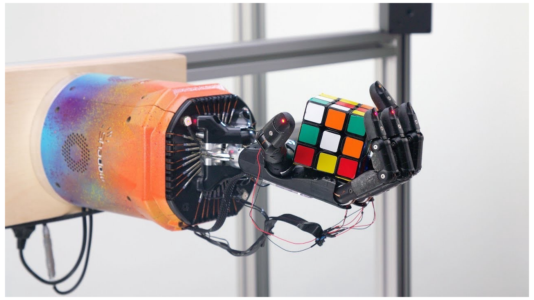


Control of dynamical systems

- Autonomous drones
- Robotics
- Data center cooling
- Medical ventilation

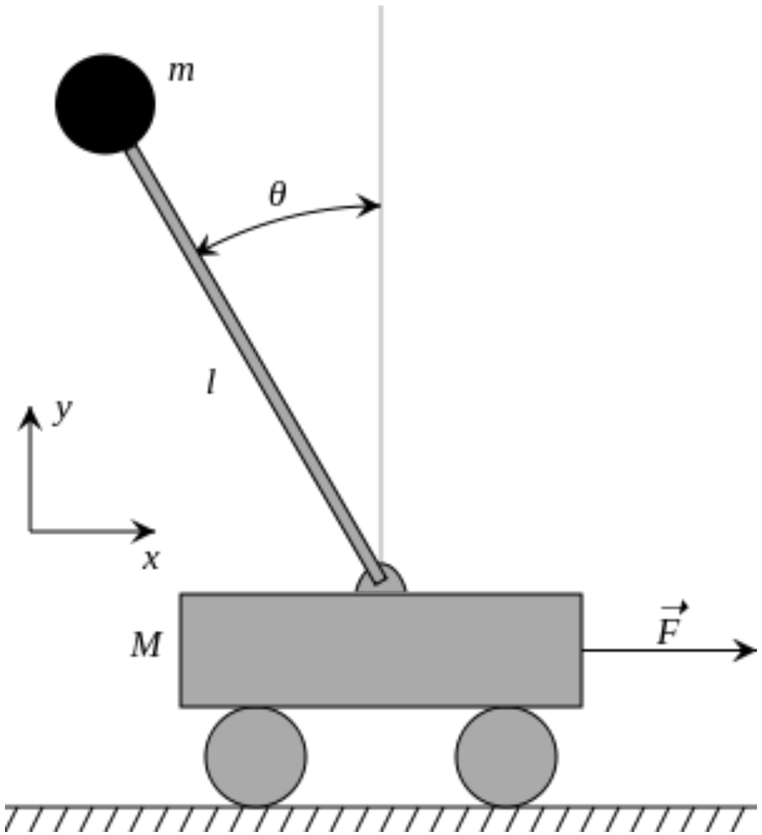
Reinforcement learning

- Atari games
- Go
- Protein folding

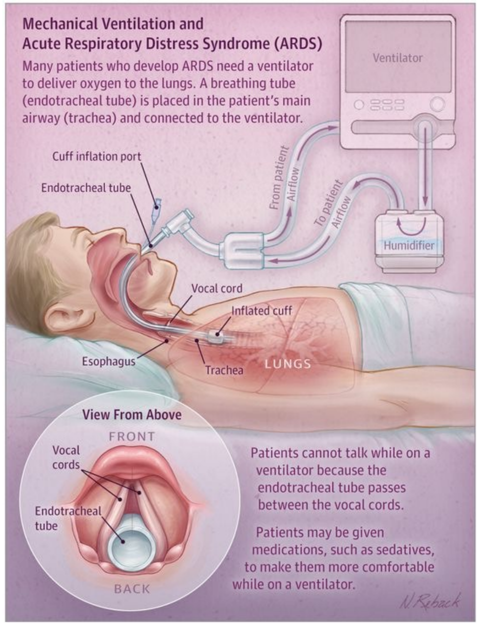


Differentiable Reinforcement Learning

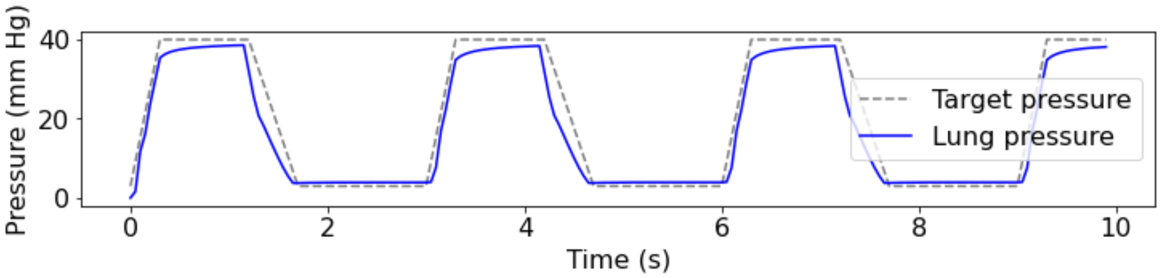
Examples



Input air+O2 flow



Observe lung/airway pressure

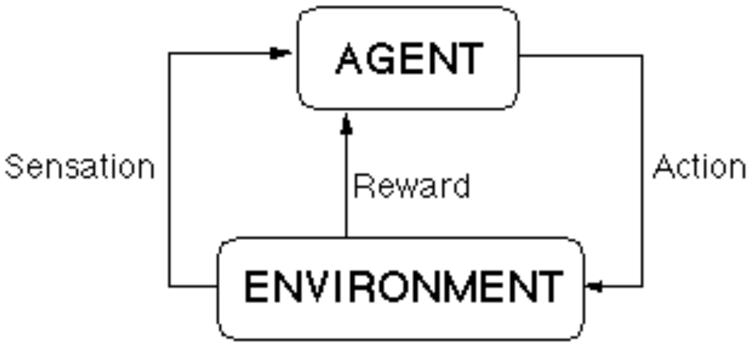


Upcoming NeurIPS/Kaggle ventilator competition!

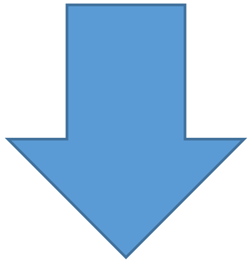
What is this tutorial about?

Reinforcement Learning / optimal control:
stochastic env., max long-term/discounted reward

Recht, ICML 2018 tutorial: "Control ≈ RL"



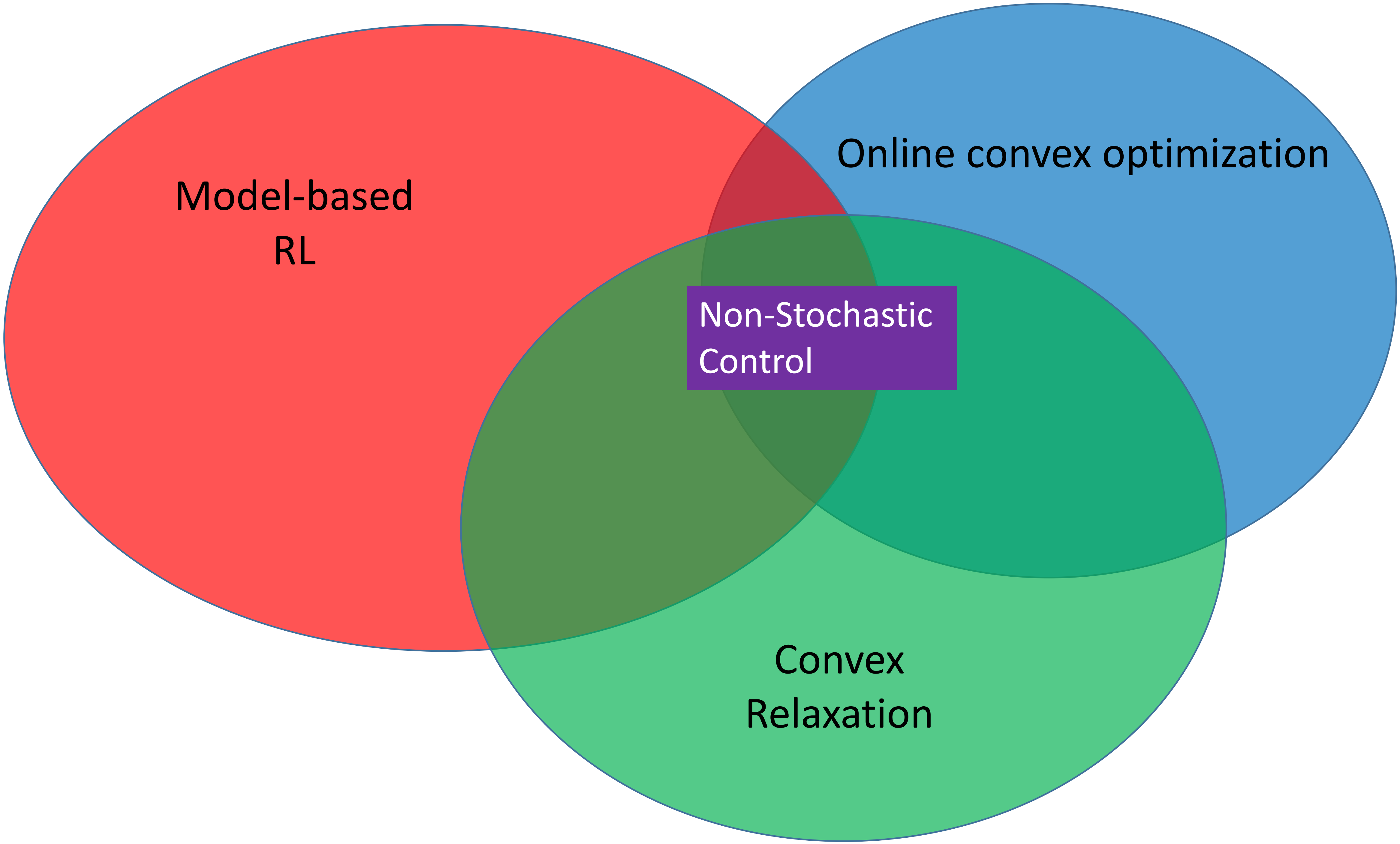
Today: "Control ≠ RL"



environment w. structure ➔ **Robust, Scalable, Gradient-based** methods**?**

→ using online convex optimization & convex relaxations → finite-time regret guarantees
→ extends to time-varying systems/planning/partial observation/bandit information/safety constraints/controller verification…

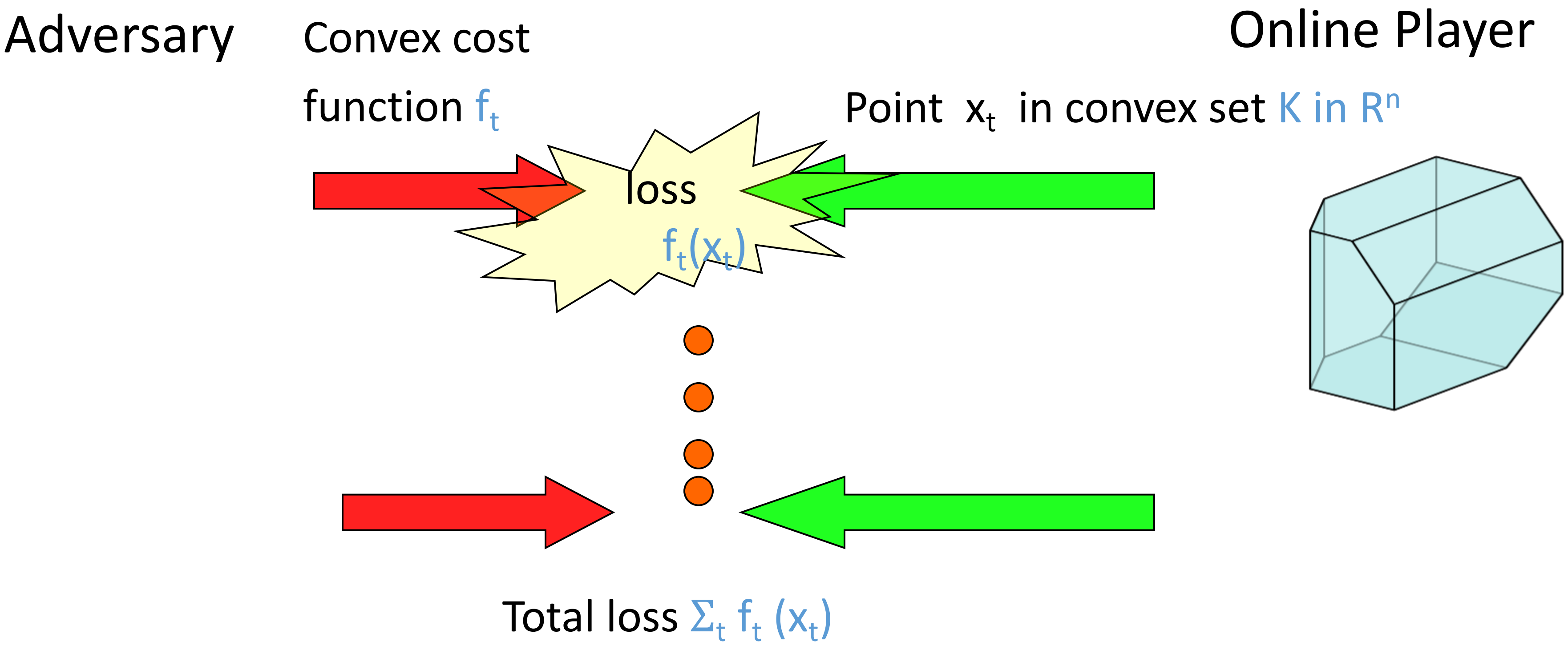
**Classic Sutton & Barto book*


Online convex optimization
Model-based
RL
Non-Stochastic
Control
Convex
Relaxation

A mini-tutorial: Online Convex Optimization (+ convex relaxation)

Non-stochastic control based on OCO + convex relaxations

Online Convex Optimization



$$\text{Regret} = \sum_t f_t(x_t) - \min_{x^* \in K} \sum_t f_t(x^*) = o(T) \text{ , or } \frac{Reg}{T} \mapsto_{T \mapsto \infty} 0$$

Examples

1. Online Linear Regression:
   - $K = \{x \mid \|x\| \le \omega\}$
   - Loss function $f_t(x) = (a_t^T x - b_t)^2$

2. Online shortest paths:
   - K = flow polytope
   - Loss function $f_t(x) = \sum_e \ell_e^t x_e$

3. Online Matrix Completion:
   - $\mathrm{K} = \{X \in R^{n \times n}, \|X\|_* \le k\}$ matrices with bounded nuclear norm
   - At time t, if $a_t = (i_t, j_t)$, then loss function $f_t(x) = (x(i_t, j_t) - b_t)^2$

Online Portfolio selection, online ranking, online ad placement / revenue maximization,….

Later today: decision set = policy class !

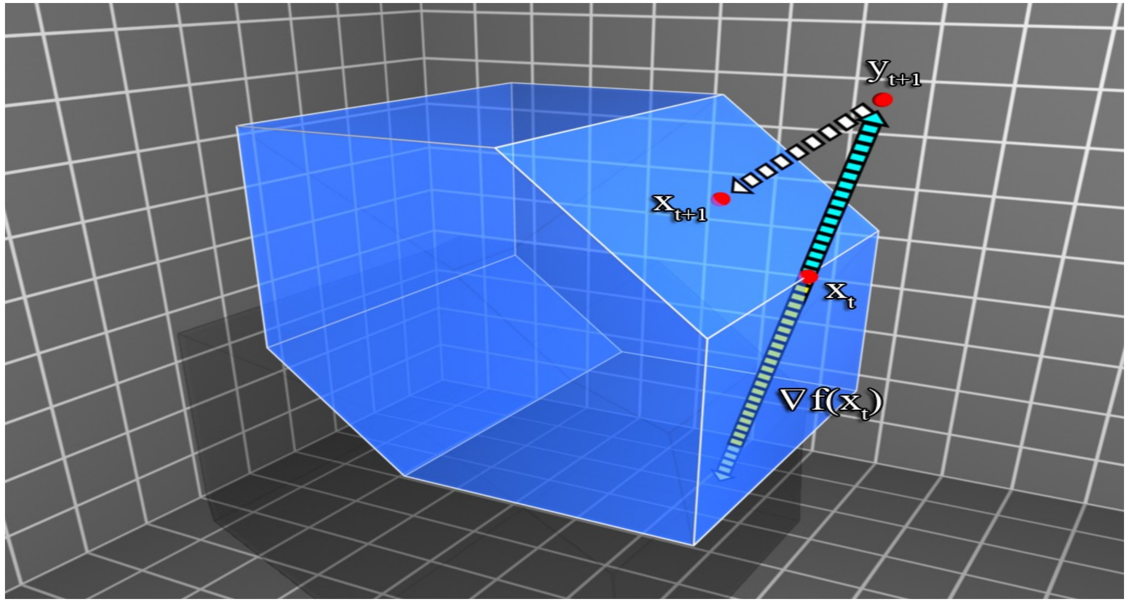
Why is OCO important?

- vs. statistical learning: more general, deterministic guarantees
- Derivation of (offline) optimization algorithms (sublinear convex optimization, adaptive regularization / AdaGrad, saddle-point optimization….)
- Learning multi-party-games, convergence to equilibria
- Allows efficient algorithms for large, structured hypothesis classes
  paths in graphs = flow polytope
  low-trace matrices for matrix completion
  …
- Bandit convex optimization,…
- By now, host of techniques/methods developed!

Online gradient descent

$$y_{t+1} = x_t - \eta \nabla f_t(x_t)$$

$$x_{t+1} = \arg\min_{x \in K} |y_{t+1} - x|$$



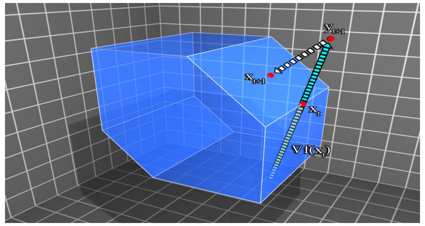
Theorem: Regret $\leq 2GD\sqrt{T}$, G = Lipschitz const, D=diameter

Analysis

$$\nabla_t := \nabla f_t(x_t)$$

$$y_{t+1} = x_t - \eta \nabla f_t(x_t)$$

$$x_{t+1} = \arg\min_{x \in K} |y_{t+1} - x|$$



Observation 1:

$$|y_{t+1} - x^*|^2 = |x_t - x^*|^2 - 2\eta \nabla_t^T (x_t - x^*) + \eta^2 |\nabla_t|^2$$

Observation 2: (Pythagoras). $|x_{t+1} - x^*|^2 \leq |y_{t+1} - x^*|^2$

Thus:

$$|x_{t+1} - x^*|^2 \leq |x_t - x^*|^2 - 2\eta \nabla_t^T (x_t - x^*) + \eta^2 |\nabla_t|^2$$

Convexity:

$$\sum_t [f_t(x_t) - f_t(x^*)] \leq \sum_t \nabla_t^T (x_t - x^*)$$

$$\leq \frac{1}{\eta} \sum_t (|x_t - x^*|^2 - |x_{t+1} - x^*|^2) + \eta \sum_t |\nabla_t|^2$$

$$\leq \frac{1}{\eta} |x_1 - x^*|^2 + \eta T G^2 \leq 2DG\sqrt{T}$$

OGD++ methods for OCO

- Fast rates with 1/t learning rate
- Online Newton Step
- Follow the perturbed leader
- Online Frank Wolfe
- Online Mirror Descent , RFTL
- Deterministic regret → SGD
- Many many extensions…

Introduction to online convex optimization

Agenda

1. The basic paradigm of non-stochastic control:
   - Pre-tutorial on OCO
   - Setting
   - Performance metric
   - Methods
2. Extensions:
   partial observation, unknown systems, bandit feedback, black-box control, time-varying systems and non-linearity
3. Advanced settings:
   adversarial noise design and controller verification, planning

Part 1: the basics of non-stochastic control

Online control with adversarial perturbations
Agarwal, Bullins, Hazan, Kakade, Singh, ICML '19

Control: basic formalization

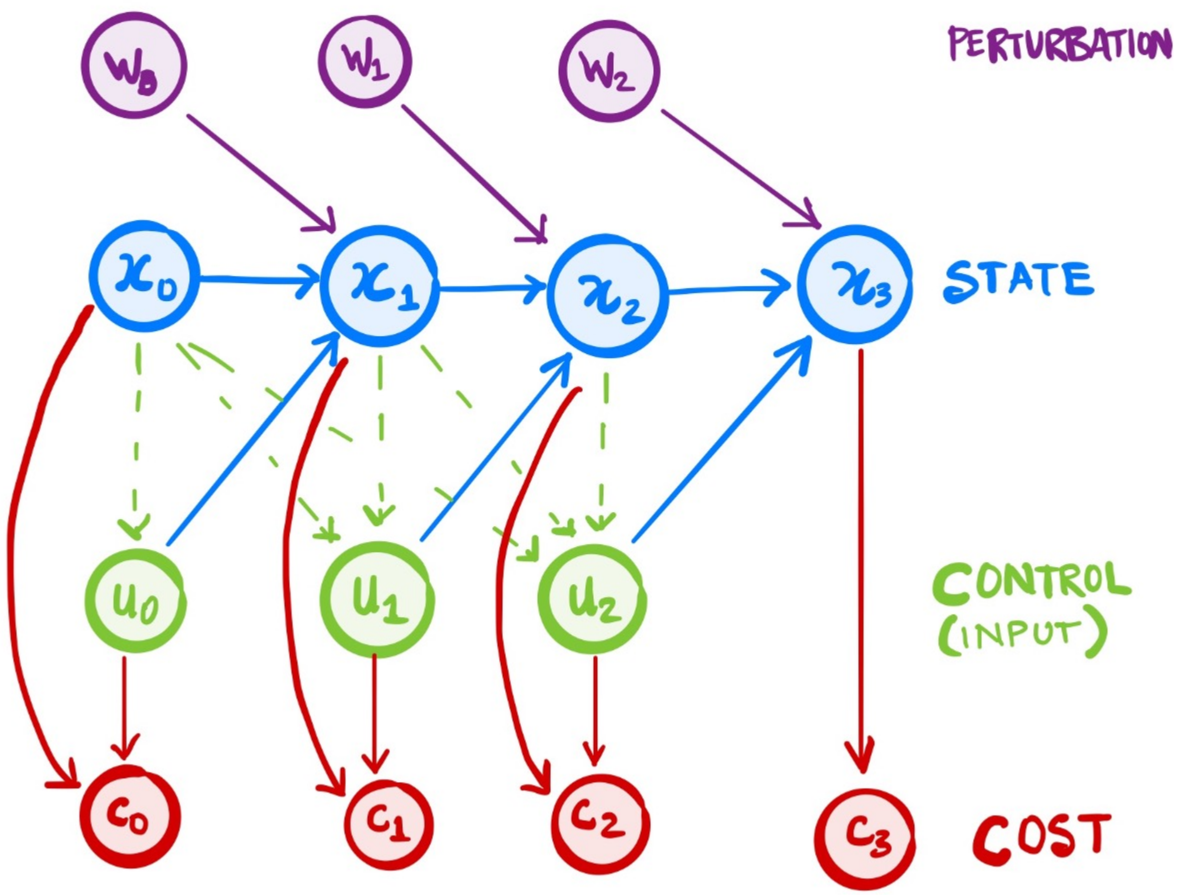
$$\min_{u(x)} \sum_{t=1}^{T} c_t(x_t, u_t)$$

$$\text{s.t.} \quad x_{t+1} = f(x_t, u_t) + w_t$$

x_t = state.

u_t = control input.

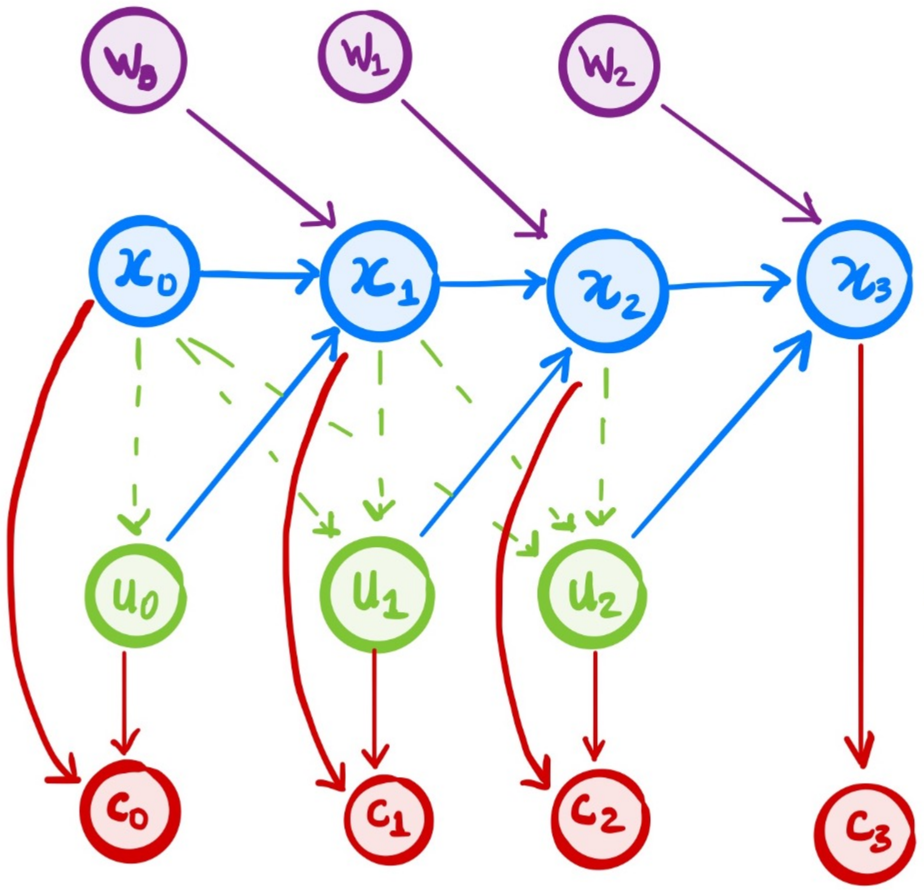
w_t = perturbation.



Control: basic formalization

$$\min_{u(x)} \sum_{t=1}^{T} c_t(x_t, u_t)$$

$$\text{s.t.} \quad x_{t+1} = A_t x_t + B_t u_t + w_t$$



x_t = state.

u_t = control input.

w_t = perturbation.

If we know the system, can we find the optimal control?

Optimal control: in principle, yes!

For stochastic perturbation,

$$\min_{\mathrm{u(x)}} \sum_{t=1}^{T} c_t(x_t, u_t)$$

$$\text{s.t.} \quad x_{t+1} = A_t x_t + B_t u_t + w_t$$

Mathematical (stochastic) optimization problem

Example: LQR

$$\min_{u(x)} \sum_{t=1}^{T} c_t(x_t, u_t)$$

$$\text{s.t.} \quad x_{t+1} = A_t x_t + B_t u_t + w_t$$

- LQR – Gaussian noise & quadratic costs **only**
  Solution (K_t depends on A_t,B_t)

$$u_t = K_t x_t$$

→ (algebraic Ricatti equation)

The Bellman optimality equation for the system:

$$v_{t-1}(x) \ = \min_u \{x^T Q x + u^T R\, u \ + v_t(A_t x + B_t u)\}$$

Backward induction: assume it's a quadratic, then opt control is linear in x…
Essentially known from the 60's, see Rechts's ICML 2018 tutorial for more information!

Example: LQR

$$\min_{u(x)} \sum_{t=1}^{T} c_t(x_t, u_t)$$
$$\text{s.t. } x_{t+1} = A_t x_t + B_t u_t + w_t$$

- LQR – Gaussian noise & quadratic costs **only**
  Solution (K_t depends on A_t,B_t)

$$u_t = K_t x_t$$

- H_∞-control:

$$\min_{K_{1:T}} \max_{|w_{1:T}|_2 \leq C} \sum_t c_t(u_t, x_t)$$

Pessimistic, computationally ill-behaved for non-quadratics (even convex costs!), non-adaptive

A notion of optimality for arbitrary noise?

1. **Regret analysis (adaptive performance metric)**
2. **Efficient methods for general losses**
   Tyrrell Rockafellar ‘87: model constraints: complicated optimal policy!

Motivating example

- Fly a drone from source to destination w. unknown weather / wind / rain / other uncertainties (non-stochastic!)
(or: track a clinician prescribed waveform - changing costs)

- Optimal/Robust control theory: all possible wind conditions
  → H_∞ overly pessimistic
  → H_2 overly optimistic

- Goal: adaptive control w. best of both worlds:
  - efficient + fast when weather permits, careful when needed
  - Optimal to instance perturbations
  - Finite time **provable** guarantees

The non-stochastic control problem

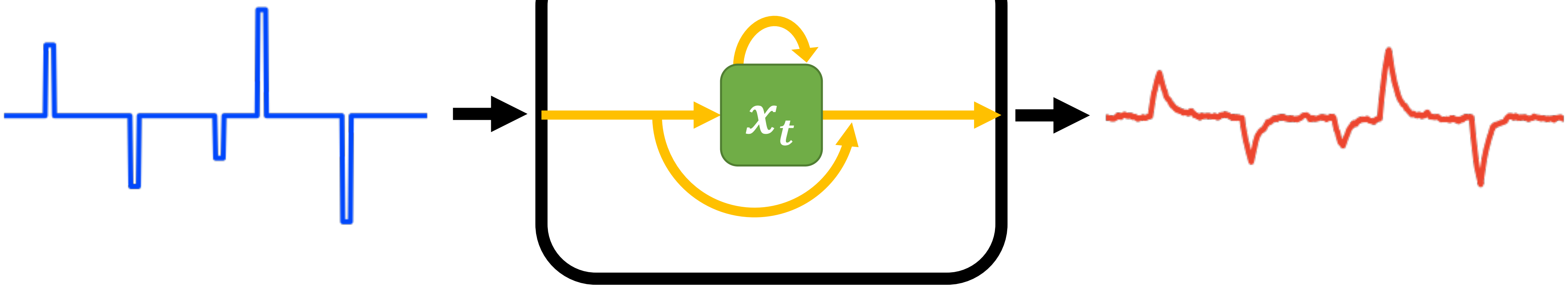
Adversarial noise in the dynamics!

Known/unknown system, full/partial observation

$$x_{t+1} = A_t x_t + B_t u_t + w_t$$
$$y_t = C_t x_t + D_t u_t + \zeta_t$$
$$c_t(y_t, u_t)$$

Input time series u_t

Output time series y_t



The non-stochastic control problem

Initially:
known system,
full observation

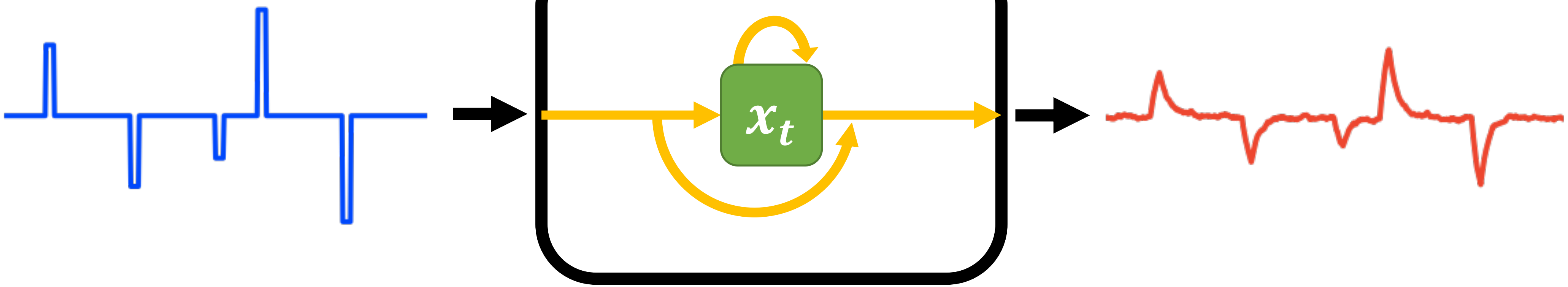
Adversarial
noise in the
dynamics!

$$x_{t+1} = A_t x_t + B_t u_t + w_t$$

$$c_t(x_t, u_t)$$

Input time series u_t

Output time series x_t



Online control of dynamical systems

- Online sequence prediction, $t = 1, \dots, T$:
  - Observe x_t , select input $u_t \in R^n$
  - Incur loss. $c_t(u_t, x_t)$

- Goal: **POLICY REGRET** (compete with “what would have happened”)

$$\max_{w_{1:T}} \left(\sum_{t=1}^{T} c_t(x_t, u_t) - \min_{\pi \in \Pi} \sum_{t=1}^{T} c_t(\hat{x}_t , \pi(\hat{x}_t)) \right)$$

- $\hat{x}_t$ = **counterfactual state sequence** under $\hat{u}_t = \pi(\hat{x}_t)$, $\hat{x}_{t+1} = A_t \hat{x}_t + B_t \hat{u}_t + w_t$
- Bounded noise $|w_t| \leq 1$

What's a reasonable comparator class? (and why do we even need one?)

- Linear Policies:

$$\Pi_{\mathrm{K}} = \{\pi_K \mid u_t = Kx_t\}$$

- Linear Dynamical Controllers: (optimal for partial observation w. Gaussian noise)

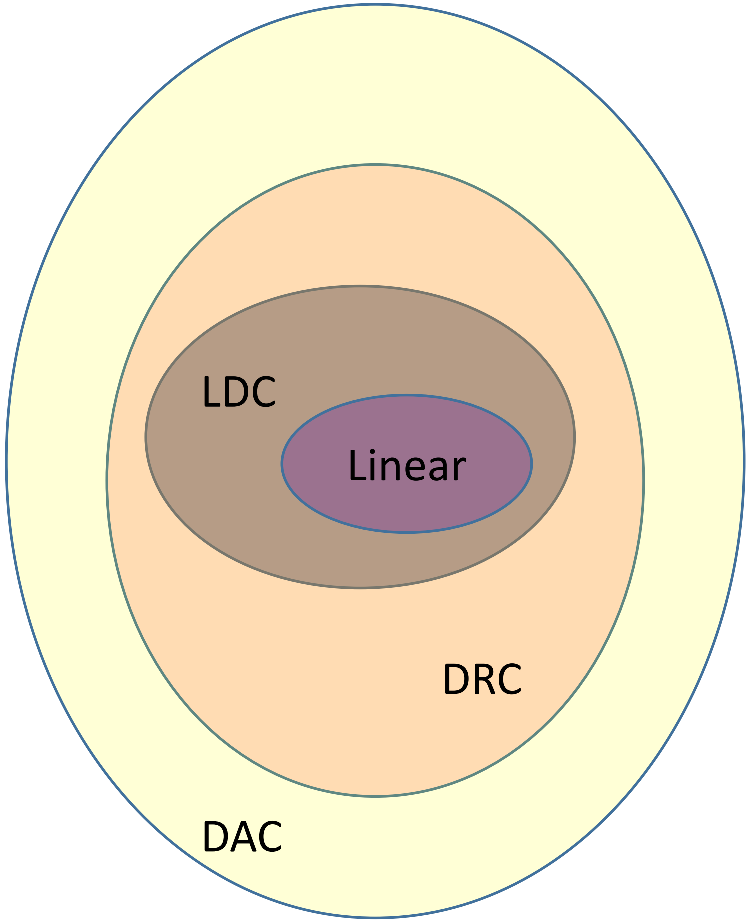
$$\Pi_{\mathrm{LDC}} = \{\pi_{A,B,C,B} \mid u_t = Cs_t + Dy_t \,, s_{t+1} = As_t + By_t\}$$

- Disturbance-action controllers:

$$\Pi_{\mathrm{DAC}} = \left\{\pi_{M_{1:H}} \mid u_t = K_t x_t + \sum_{i}^{H} M_i w_{t-i}\right\}$$

- Disturbance-response controllers:

$$\Pi_{\mathrm{DRC}} = \left\{\pi_{M_{1:H}} \mid u_t = K_t y_t + \sum_{i}^{H} M_i y_{t-i}^{nat}\right\}$$



Hierarchy for LTI systems only!

1st basic result

Efficient algorithm s.t.

$$\sum_{t=1}^{T} c_t(x_t, u_t) - \min_{\pi \in \Pi_{DAC}} \left(\sum_{t=1}^{T} c_t\left(\widehat{x_t}, \pi(\widehat{x_t})\right) \right) \leq O(\sqrt{T})$$

- Efficient → Polynomial in system parameters, logarithmic in T

Up next: analysis main ideas+algorithm

Ingredient 1: Convex Relaxation of Π_K

to simplify derivation, **assume LTI**

- With $w_{1:T}$ known, optimal K is non-convex problem:

$$u_{t+1}(K) = Kx_{t+1} = K \cdot \left(\sum_{i=0}^{t} (A + BK)^i w_{t-i} \right)$$

- Relaxation ($\vec{M} = \{M_1 \ \ldots M_t\}$):

$$\min_{M} \left(\sum_{t=1}^{T} c\left(x_t(\vec{M}), u_t(\vec{M}) \right) \right)$$

is convex!

$$u_{t+1}(\vec{M}) = \overrightarrow{M_t} \cdot \overrightarrow{w_t} = \left(\sum_{i=0}^{t} M_i w_{t-i} \right)$$

Ingredient 2: Enforcing stability & learnability

- $\mathrm{K_t}$ = stabilizing linear policy (for A_t, B_t)
- Optimal controls:

$$\boxed{u_t = K_t x_t + \sum_{i=1}^{H} M_i^t w_{t-i}}$$

- **Representation Power:** With $H \approx \frac{1}{\epsilon}$, can ϵ -emulate any stable policy.
- **Stability:** K stablizing $\Rightarrow$ any (non-stationary) error feedback policy is stable.

- How do we find stabilizing K?
  [“black-box control”… TBD!]

Ingredient 3: OCO with memory

- Adversarial sequence with time dependency:

$$f_t(M_{1:H}^t | \dots) = f_t\left(M_{1:H}^t, M_{1:H}^{t-1}, \dots, M_{1:H}^{t-q}\right)$$

- Regret vs. best fixed decision

$$\sum_{t=1}^{T} f_t(M_{1:H}^t, \dots, M_{1:H}^{t-q}) - \min_{M_{1:H}} \sum_t f_t(M_{1:H}, \dots, M_{1:H}) = O(qH\sqrt{T})$$

- slow-moving iterative methods, exploiting Lipschitzness

Fundamentally new method: Gradient Perturbation Controller (GPC)

Initialize $\vec{M} = M_1, \dots, M_H$
For $t = 1, \dots, T$ do

1. Use control $u_t = K_t x_t + \sum_{i \le H} M_i \, w_{t-i}$

2. Observe state x_{t+1}, compute noise $w_t = x_{t+1} - A_t x_t - B_t u_t$.

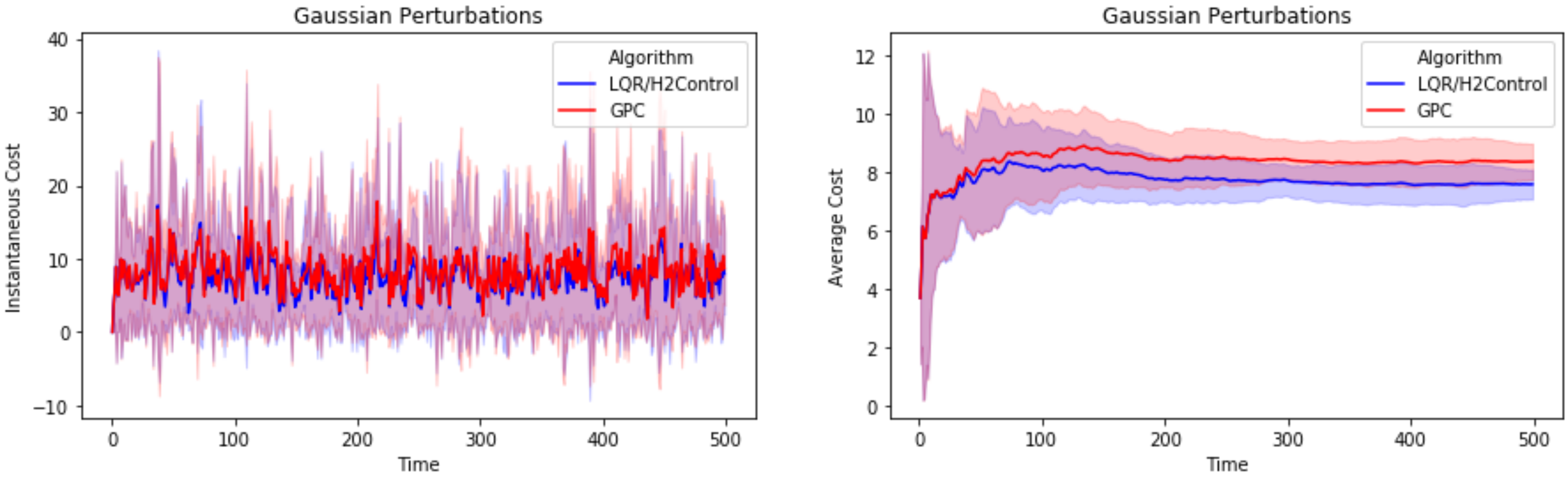
3. Construct cost function:

$$\ell_t(\vec{M}) = c_t\big(x_t(M_{1:H}), u_t(M_{1:H})\big)$$

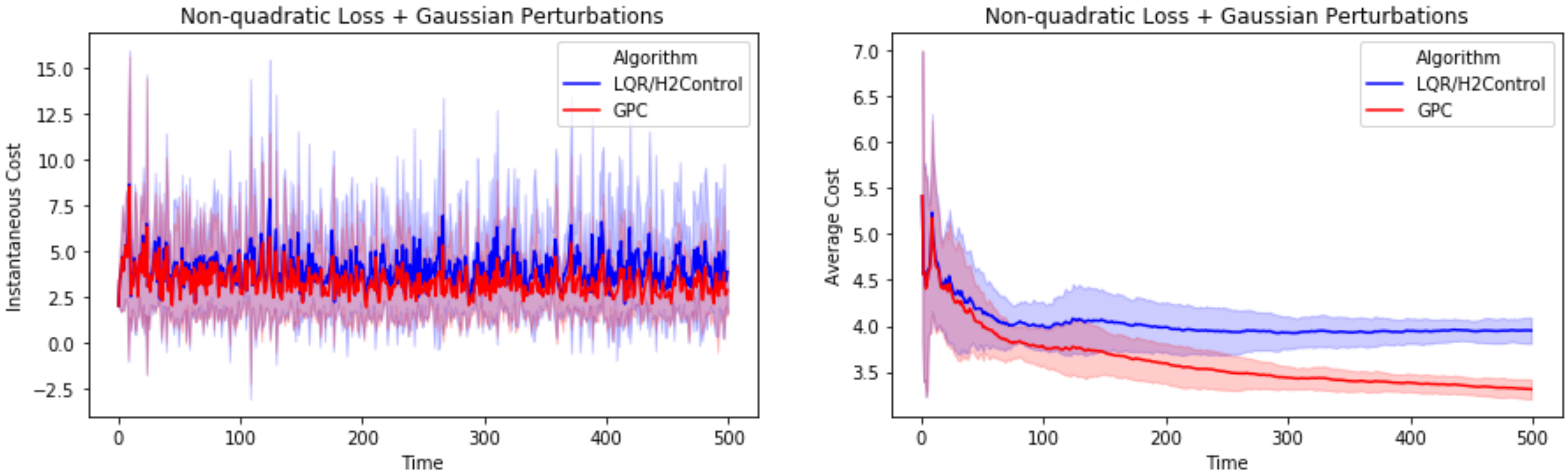
4. Update $\vec{M}$

$$\vec{M} \leftarrow \vec{M} - \eta \, \nabla_{\vec{M}} \, \ell_t(\vec{M})$$

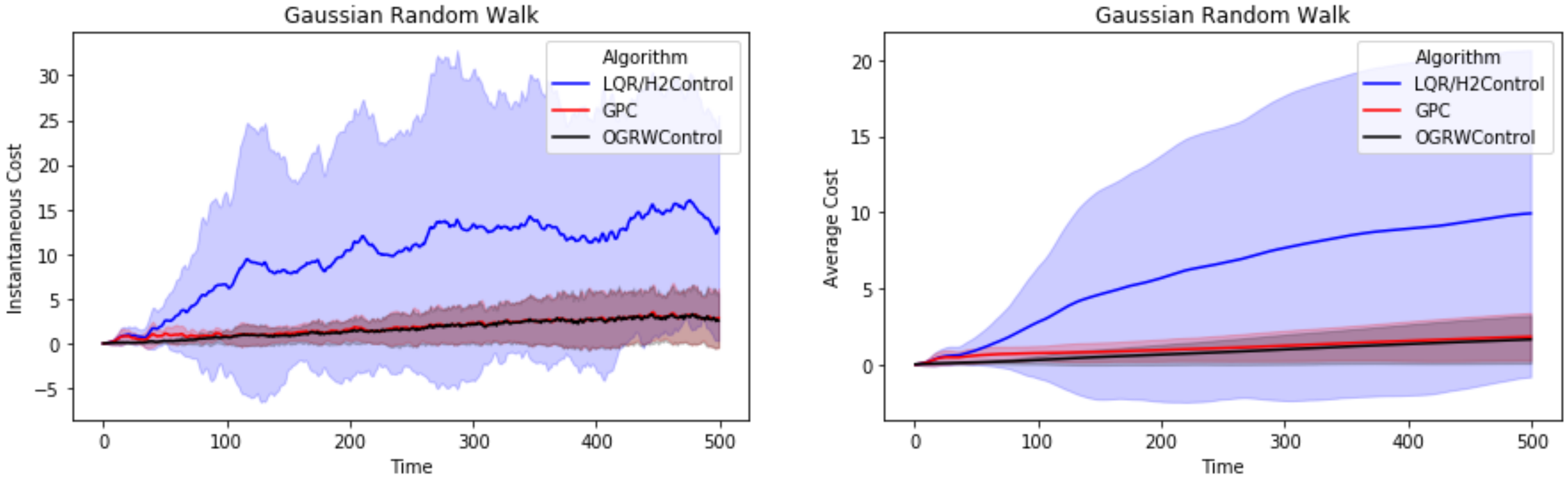
Gradient perturbation controller: in action



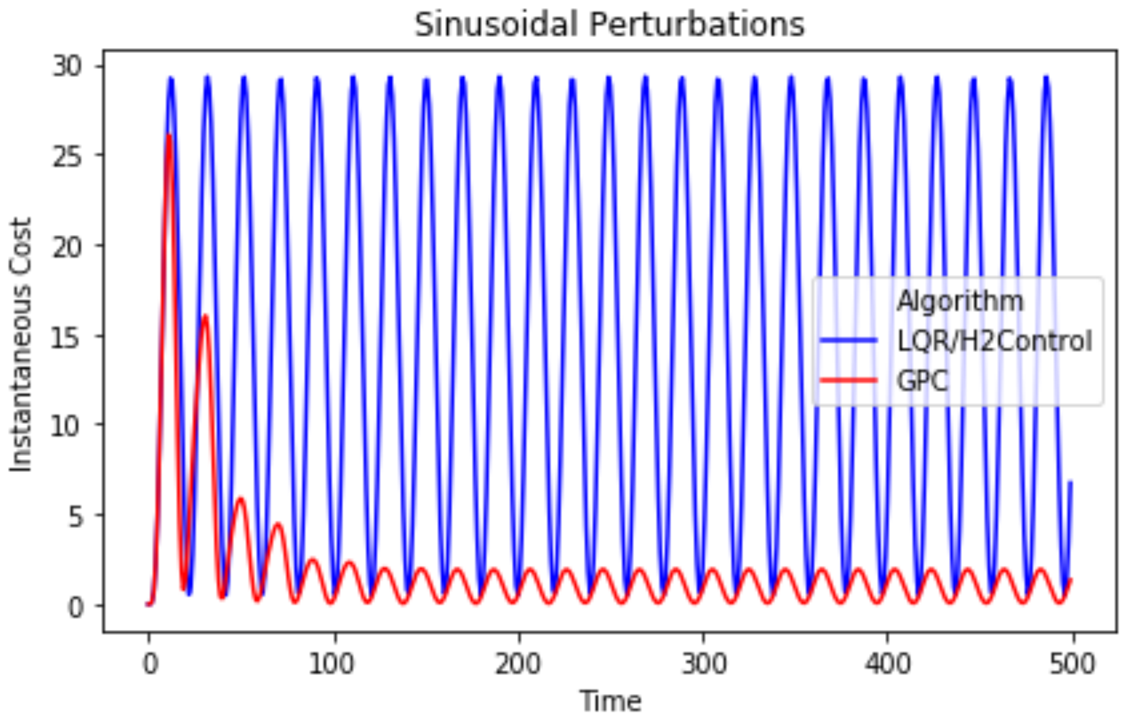
Gradient perturbation controller: in action

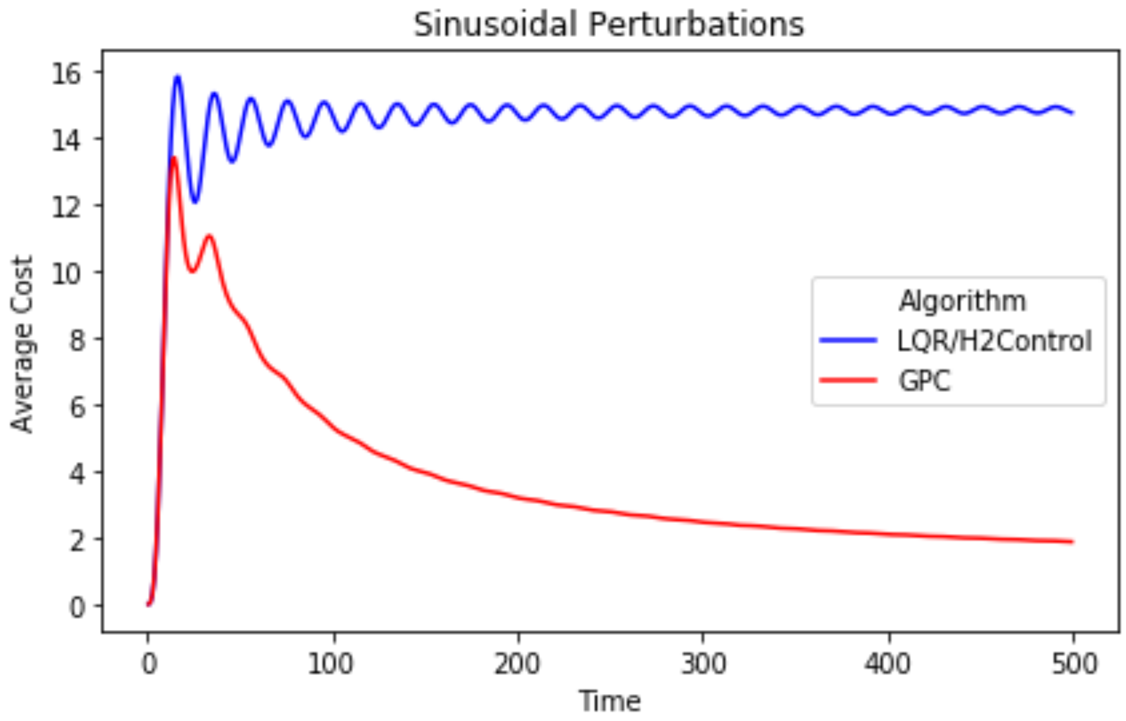


Gradient perturbation controller: in action

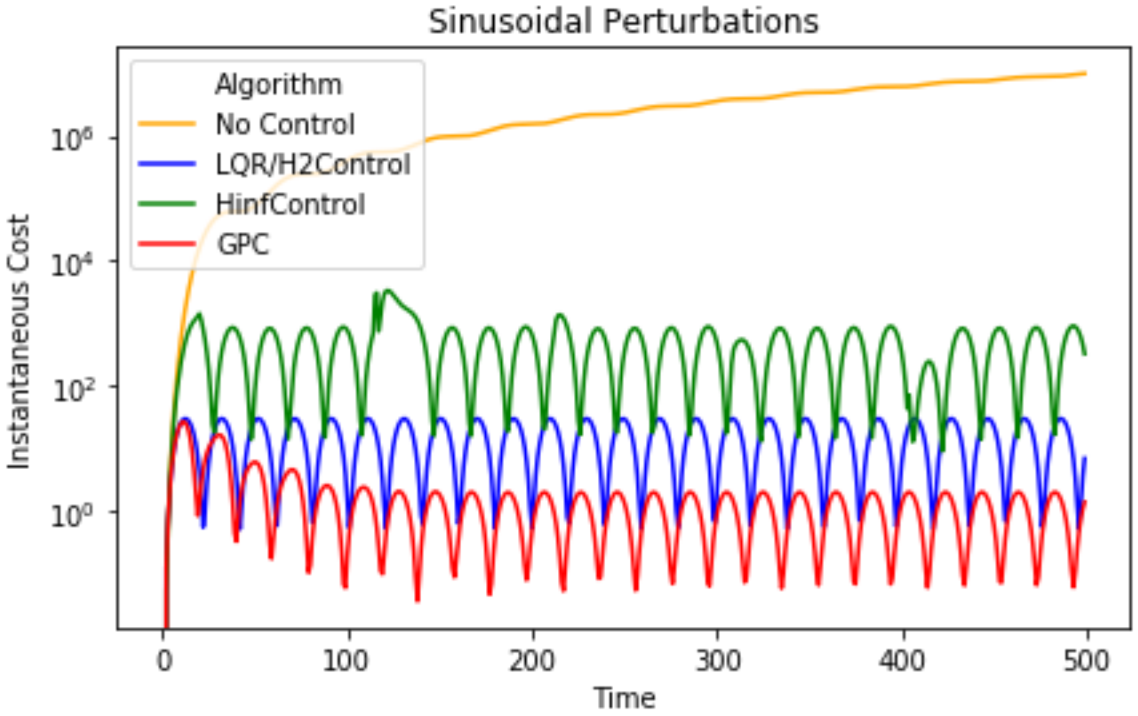


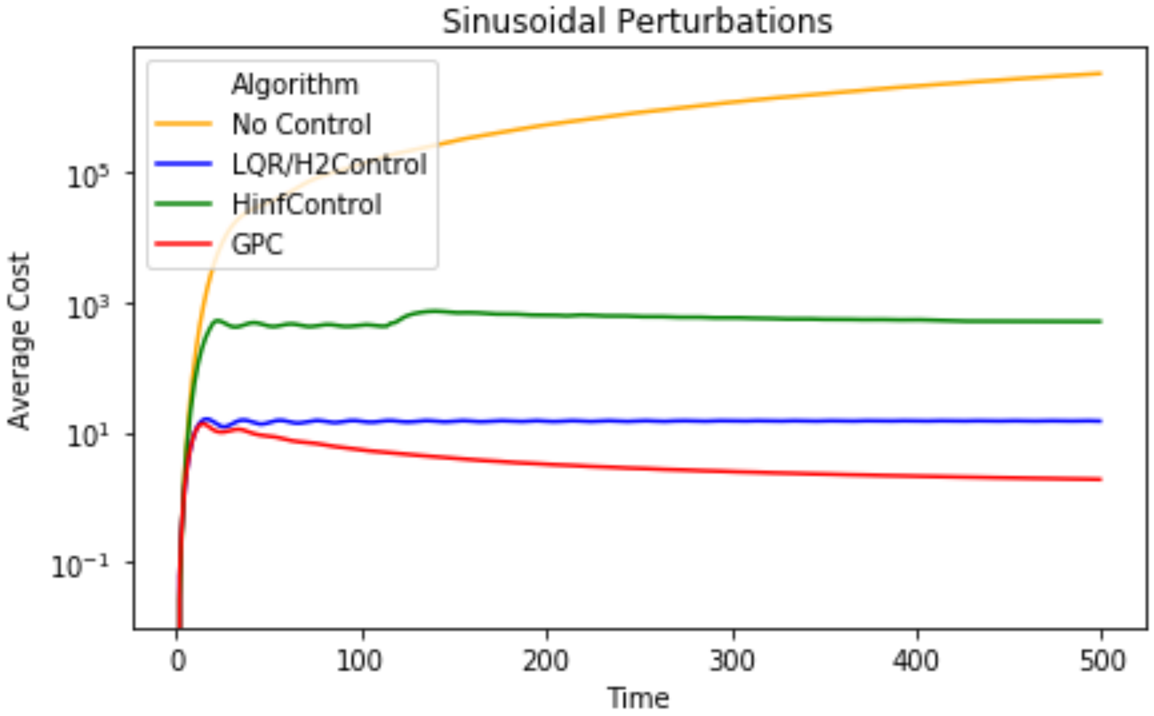
Gradient perturbation controller: in action





Gradient perturbation controller: in action



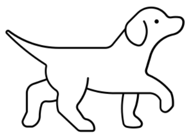


Agenda

1. The basic paradigm of non-stochastic control:
   - Pre-tutorial on OCO
   - Setting
   - Performance metric
   - Methods
2. Extensions:
   partial observation, unknown systems, bandit feedback, black-box control, time-varying systems and non-linearity
3. Advanced settings:
   adversarial noise design and controller verification, planning

Summary – 1st part

1. The power of control: differentiation through the environment
2. Motivation for more robust (adversarial noise), scalable (iterative gradient method, environment differentiation) new methods
3. The power of online convex optimization and convex relaxation: mini-tutorial on OCO
4. Deriving the Gradient Perturbation Controller (GPC)
5. Resources:
   [Tutorial website]
   [More info on OCO]
   [COLAB NOTEBOOKS FOR ALL EXPERIMENTS]



Thank you ! Part 2 coming up!

Questions for this tutorial

1. What’s the need for innovation in differentiable reinforcement learning? What applications are you thinking of and how can they benefit from new methods?

2. How is online non-stochastic control what you’re doing different from RL/classical control? Why is this important?

3. What’s the essence of the new methods? What techniques are they using?

4. Where do you see this field going? What potential extensions are there? What are the hardest unsolved problems?

Agenda

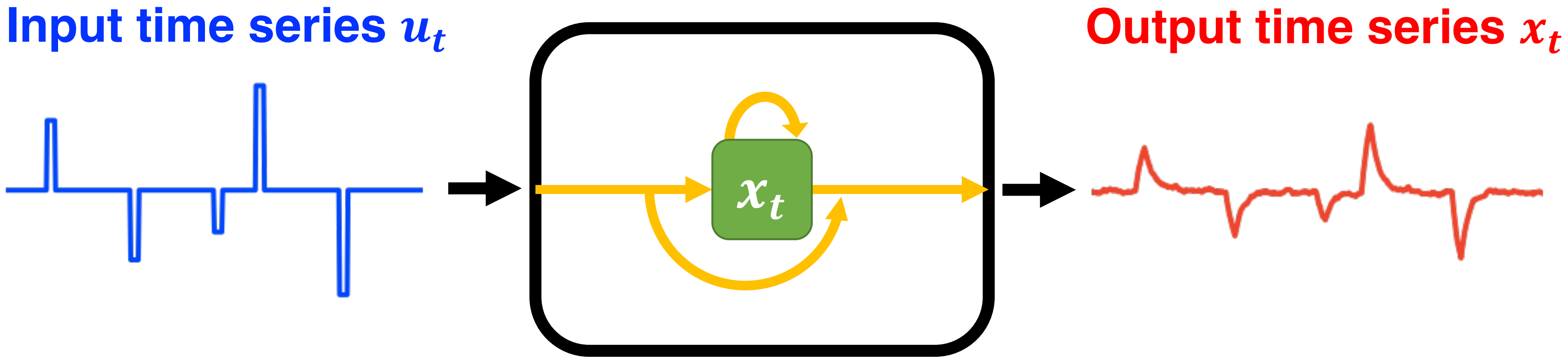
1. The basic paradigm of non-stochastic control:
   - Pre-tutorial on OCO
   - Setting
   - Performance metric
   - Methods
2. Extensions:
   partial observation, unknown systems, bandit feedback, black-box control, time-varying systems and non-linearity
3. Advanced settings:
   adversarial noise design and controller verification, planning

What if we don't know the system?

A,B = system

$$x_{t+1} = Ax_t + Bu_t + w_t$$

$$c_t(x_t, u_t)$$



Non-stochastic control w/o system

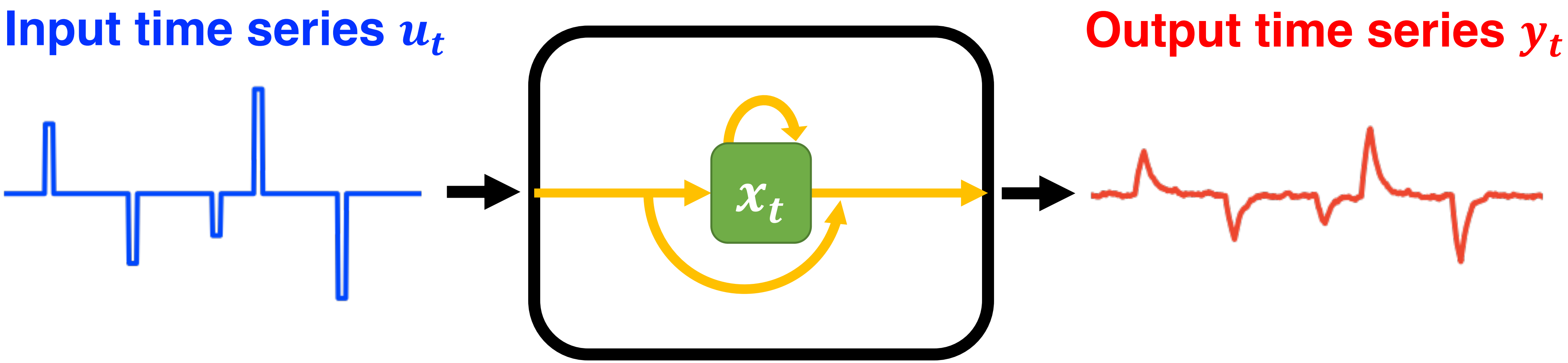
- Identify the system with adversarial (small!) noise!
- Key idea: activate w. random noise (or additive component):
  $x_{t+1} = Ax_t + Bu_t + w_t, \;\; u_t \sim N(0, \Sigma)$
- Now: $\mathrm{E}[\, x_{t+k} u_t] = E\left[\sum_{i=0:k} A^i \, (Bu_{t+k-i} + w_t) u_t\right] = \; A^k B$
- From here on: Kalman matrix reconstruction, sys-id, GPC…

The nonstochastic control problem
Hazan, Kakade, Singh, ALT '19

NSC w. partial observation

State and system are unknown!

$$x_{t+1} = A x_t + B u_t + w_t$$
$$y_t = \boldsymbol{C} x_t + \boldsymbol{D} u_t + \zeta_t$$
$$c_t(y_t, u_t)$$

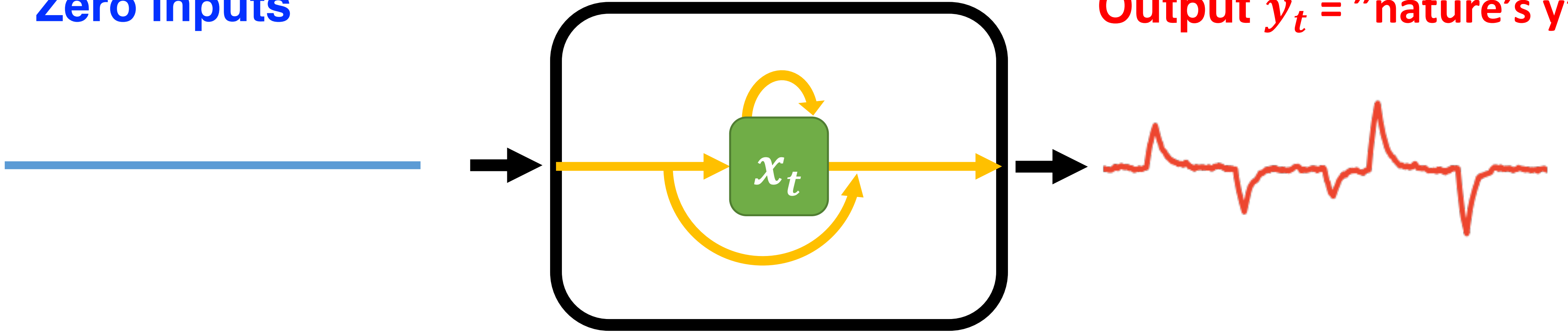


“Nature’s y’s “   (Youla reparametrization)

$$x_{t+1} = Ax_t + Bu_t + w_t$$
$$y_t = \boldsymbol{C}x_t + \boldsymbol{D}u_t + \zeta_t$$
$$c_t(y_t, u_t)$$

Zero inputs

Output y_t = ”nature’s y’s”



What’s a reasonable comparator class?

- Linear Policies:

$$\Pi_{\mathrm{K}} = \{\pi_K \,| u_t = K x_t\}$$

- Linear Dynamical Controllers: (optimal for partial observation w. Gaussian noise)

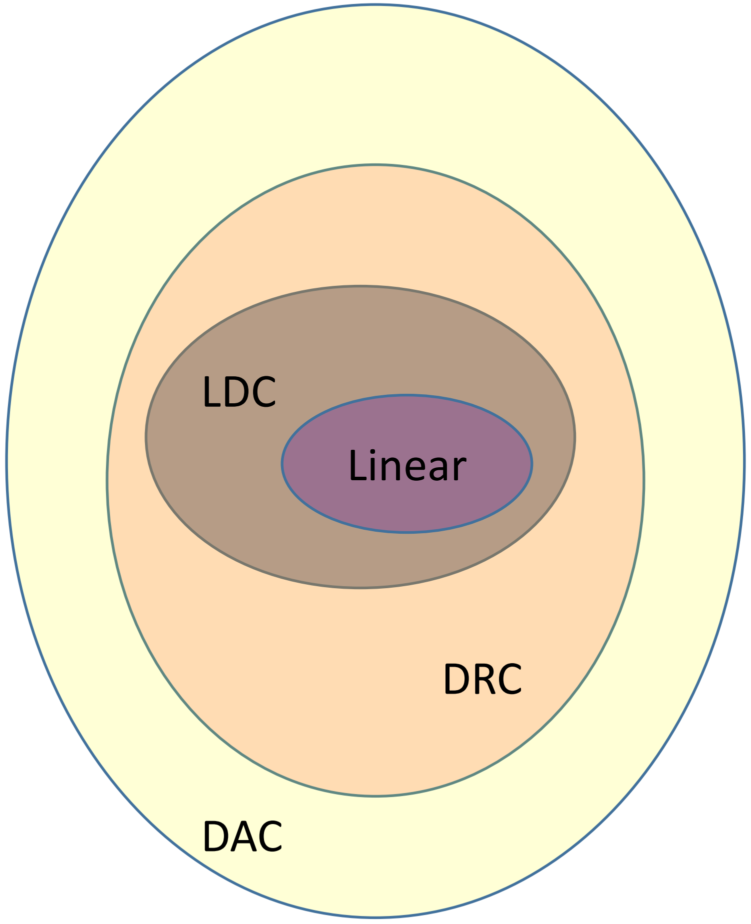
$$\Pi_{\mathrm{LDC}} = \{\pi_{A,B,C,B} \,| u_t = C s_t + D y_t \,, s_{t+1} = A s_t + B y_t\}$$

- Disturbance-action controllers:

$$\Pi_{\mathrm{DAC}} = \left\{\pi_{M_{1:H}} \mid u_t = K_t x_t + \sum_{i}^{H} M_i w_{t-i}\right\}$$

- Disturbance-response controllers:

$$\Pi_{\mathrm{DRC}} = \left\{\pi_{M_{1:H}} \mid u_t = K_t y_t + \sum_{i}^{H} M_i y_{t-i}^{nat}\right\}$$



Gradient Response Controller (LTI, full obs.)

Initialize $\vec{M} = M_1, \ldots, M_H$

For $t = 1, \ldots, T$ do

1. Use control $u_t = Kx_t + \sum_{i \le H} M_i \, x_{t-i}^{nat}$

2. Observe state x_{t+1}, compute noise and nature's x:

   $w_t = x_{t+1} - Ax_t - Bu_t \,,\; x_{t+1}^{nat} = Ax_t^{nat} + w_t.$

3. Construct cost function:

$$\ell_t(\vec{M}) = c_t\big(x_t(M_{1:H}), u_t(M_{1:H})\big)$$

4. Update $\vec{M}$

$$\vec{M} \leftarrow \vec{M} - \eta \, \nabla_{\vec{M}} \, \ell_t(\vec{M})$$

NSC w. Partial observation

Non-stochastic control ,unknown system & **partially observed state:**

1. Compete w. Π_{DFC}
2. $O(T^{2/3})$ regret
3. $O(T^{1/2})$ regret for quadratics: “improper LQG”
   (first efficient algorithm even for stochastic setting)

Via: Youla reparametrization, “Nature’s y’s”, online gradient methods

Improper learning for non-stochastic control
Simchowitz, Singh, Hazan, COLT ‘20

ICML Tutorial:
Online & Non-stochastic control

PRINCETON UNIVERSITY

Google AI Princeton

Microsoft Research

Elad Hazan

Karan Singh

https://sites.google.com/view/nsc-tutorial/home

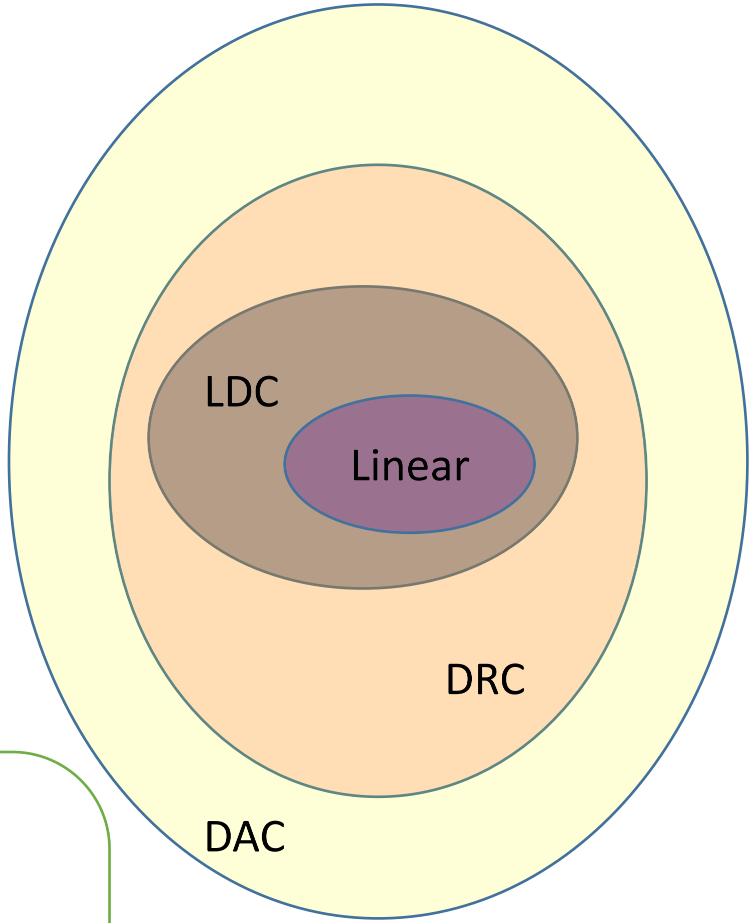
Recap from Part I

Adversarial noise in the dynamics!

Part I:
Time-invariant known system, full observation

$$x_{t+1} = Ax_t + Bu_t + w_t$$

$$c_t(x_t, u_t)$$



Efficient gradient-based algorithm s.t.

$$\sum_{t=1}^{T} c_t(x_t, u_t) \; - \; \min_{\pi \in \Pi_{DAC}} \left(\sum_{t=1}^{T} c_t\left(\hat{x}_t, \pi(\widehat{x_t})\right) \right) \; \le O(\sqrt{T})$$

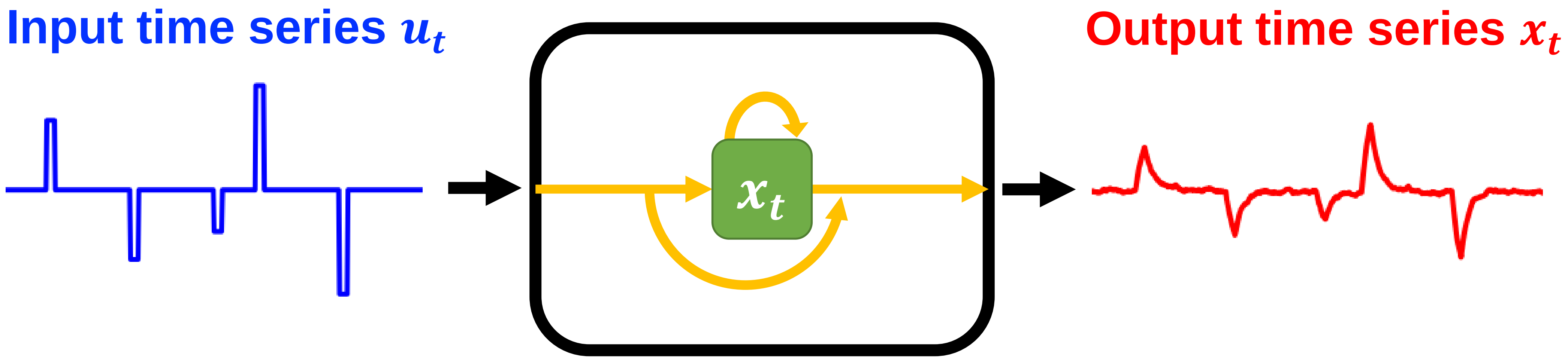
Agenda

1. The basic paradigm of non-stochastic control:
   - Pre-tutorial on OCO
   - Setting
   - Performance metric
   - Methods
2. Extensions:
   unknown systems, partial observability, bandit feedback, black-box control, time-varying systems
3. Applications:
   adversarial noise design and controller verification, planning

What if we don't know the system?

A,B = system

$$x_{t+1} = Ax_t + Bu_t + w_t$$
$$c_t(x_t, u_t)$$



Non-stochastic control for unknown system

Stochastic Noise: Use MLE/least-squares to recover parameters.

Non-stochastic perturbations ⇒ inconsistent estimates.

Any coarse stabilizing matrix K.

Here: inject random noise (or as an additive component):

$$x_{t+1} = Ax_t + Bu_t + w_t, \quad u_t \sim N(0, \mathrm{I}) \; or \; K\mathrm{x}_\mathrm{t} + N(0, \mathrm{I})$$

- Now: $\mathrm{E}[\, x_{t+k} u_t] = E\left[\sum_{i=0:k} A^i \, (Bu_{t+k-i} + w_t + x_t) u_t\right] = \; A^k B$
- Then: sys-id (i.e. recover A, B), do GPC with estimated system → $T^{\frac{2}{3}}$ regret.

The nonstochastic control problem
Hazan, Kakade, Singh, ALT '19

Non-stochastic control for unknown system

Stochastic Noise: Use MLE/least-squares to recover parameters.

Non-stochastic perturbations ⇒ inconsistent estimates.

Here: inject random noise (or as an additive component):

Any coarse stabilizing matrix K.

$$x_{t+1} = Ax_t + Bu_t + w_t, \quad u_t \sim N(0, \mathrm{I}) \ or \ K\mathrm{x}_\mathrm{t} + N(0, \mathrm{I})$$

- Now: $\mathrm{E}[\, x_{t+k} u_t] = E\left[\sum_{i=0:k} A^i \,(Bu_{t+k-i} + w_t + x_t)u_t\right] = \ A^k B$
- Then: sys-id (i.e. recover A, B), do GPC with estimated system → $T^{\frac{2}{3}}$ regret.

The nonstochastic control problem
Hazan, Kakade, Singh, ALT '19

Without any prior knowledge

How to construct a stabilizing control?

Can construct a stabilizing controller at $O\left(2^d\right)$ cost.

Leading to $O\left(2^d + T^{\frac{2}{3}}\right)$ regret.

Theorem: Even for noiseless linear systems, ANY control algorithm has worst case regret:

$$\sum_{t \in T} c_t(x_t, u_t) \; - \; \min_{\pi \in \Pi_{LC}} \left(\sum_{t \in T} c_t\left(\widehat{x_t}, \pi(\widehat{x_t})\right) \right) \; \geq \Omega(2^d)$$

Blackbox Control of Linear Dynamical Sys
Chen, Hazan COLT ‘21

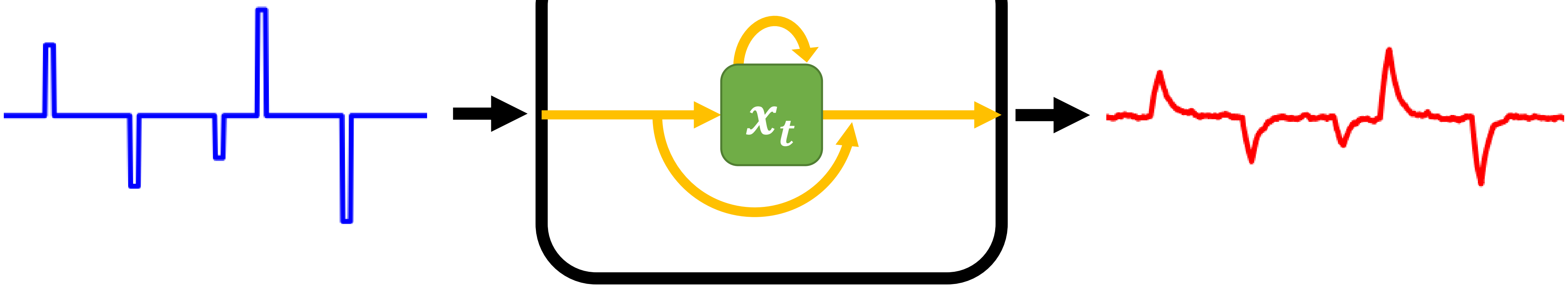
NSC under partial observability

$$x_{t+1} = Ax_t + Bu_t + w_t$$
$$y_t = \boldsymbol{C}x_t$$
$$c_t(y_t, u_t)$$

Can only observe y_t!

Input time series u_t

Output time series y_t



What’s a reasonable comparator class?

- Linear Policies:

$$\Pi_{\mathrm{K}} = \{\pi_K \mid u_t = K x_t\}$$

- Linear Dynamical Controllers: (optimal for partial observation w. Gaussian noise)

$$\Pi_{\mathrm{LDC}} = \{\pi_{A,B,C,B} \mid u_t = C s_t + D y_t \, , s_{t+1} = A s_t + B y_t\}$$

- Disturbance-action controllers:

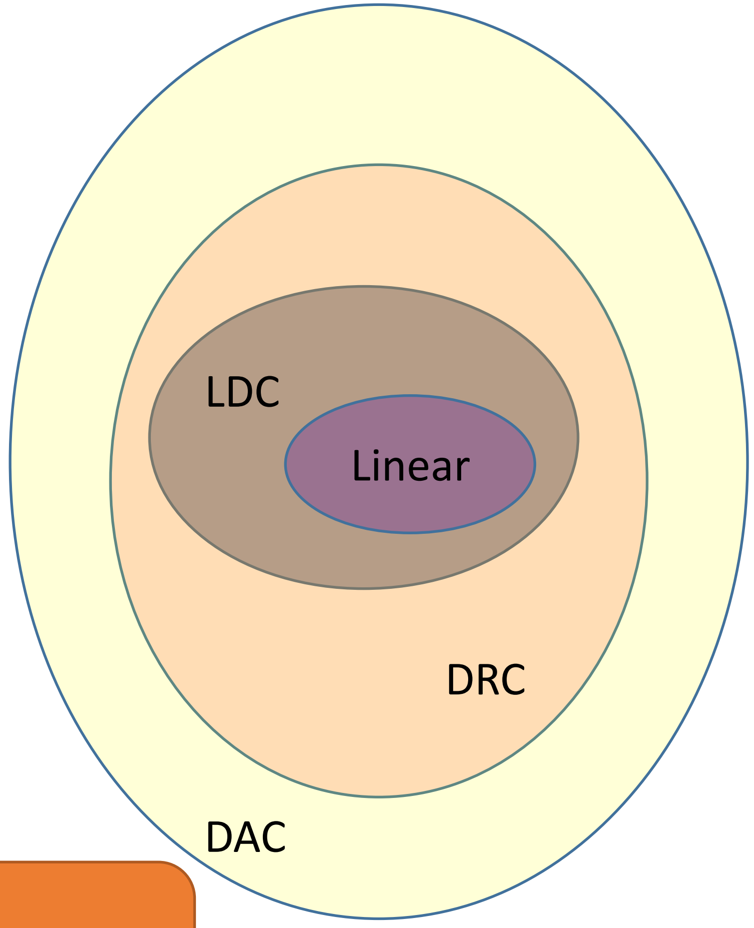
$$\Pi_{\mathrm{DAC}} = \left\{ \pi_{M_{1:H}} \mid u_t = K_t x_t + \sum_{i}^{H} M_i w_{t-i} \right\}$$

- Disturbance-response controllers:

$$\Pi_{\mathrm{DRC}} = \left\{ \pi_{M_{1:H}} \mid u_t = K_t y_t + \sum_{i}^{H} M_i y_{t-i}^{nat} \right\}$$

GPC plays / competes against DACs.

Can’t calculate w_t even when A, B, C are known.

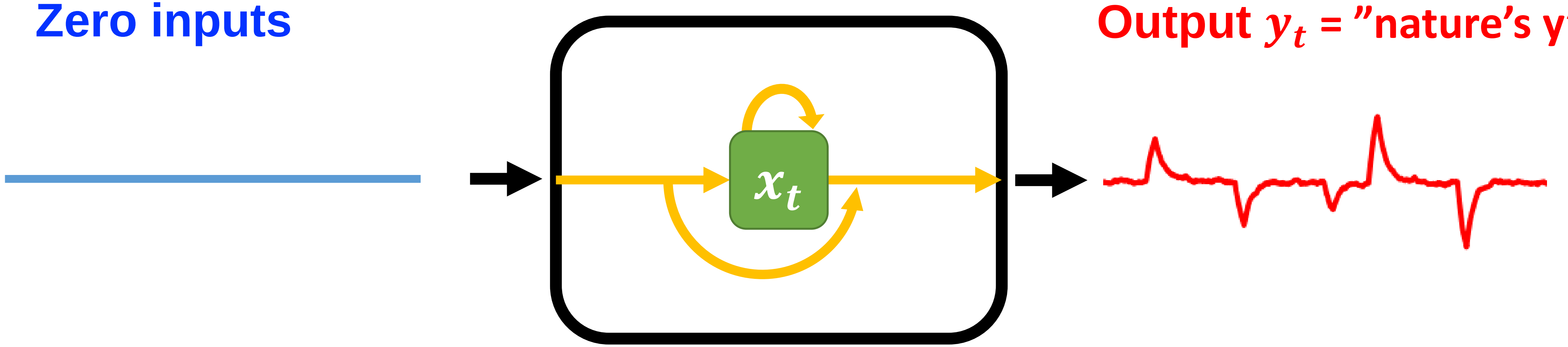


“Nature’s y’s” (rel. Youla reparametrization)

$$x_{t+1} = Ax_t + Bu_t + w_t$$
$$y_t = \boldsymbol{C}x_t$$
$$c_t(y_t, u_t)$$

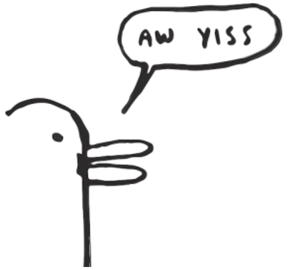
Zero inputs

Output y_t = ”nature’s y’s”



Improper learning for non-stochastic control
Simchowitz, Singh, Hazan, COLT ‘20

Source: Hark, A Vagrant



What’s a reasonable comparator class?

- Linear Policies:

$$\Pi_{\mathrm{K}} = \{\pi_K \mid u_t = K x_t\}$$

- Linear Dynamical Controllers: (optimal for partial observation w. Gaussian noise)

$$\Pi_{\mathrm{LDC}} = \{\pi_{A,B,C,B} \mid u_t = C s_t + D y_t \,, s_{t+1} = A s_t + B y_t\}$$

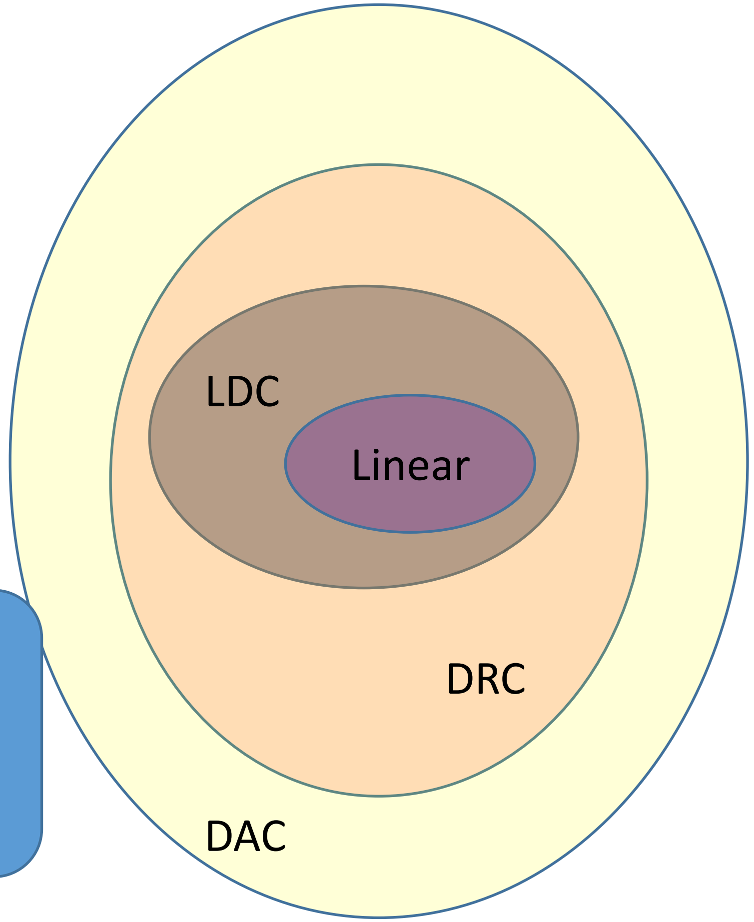
- Disturbance-action controllers:

$$\Pi_{\mathrm{DAC}} = \left\{ \pi_{M_{1:H}} \mid u_t = K_t x_t + \sum_{i}^{H} M_i w_{t-i} \right\}$$

- Disturbance-response controllers:

$$\Pi_{\mathrm{DRC}} = \left\{ \pi_{M_{1:H}} \mid u_t = K_t y_t + \sum_{i}^{H} M_i y_{t-i}^{nat} \right\}$$

Can be computed purely from y_t and A, B, C.



$$x_{t+1} = A x_t + B u_t + w_t$$
$$y_t = \boldsymbol{C} x_t$$

Gradient Response Controller (partial obs.)

Initialize $\vec{M} = M_1, \dots, M_H$
For $t = 1, \dots, T$ do

1. Use control $u_t = \sum_{i \le H} M_i \, y_{t-i}^{nat}$

2. Observe y_{t+1}, compute nature's y:
   $y_{t+1}^{nat} = y_{t+1} - CABu_t - CA^2Bu_{t-1} + \cdots ..$

3. Construct cost function:
$$\ell_t(\vec{M}) = c_t\big(y_t(M_{1:H}), u_t(M_{1:H})\big)$$

4. Update $\vec{M}$
$$\vec{M} \leftarrow \vec{M} - \eta \, \nabla_{\vec{M}} \, \ell_t(\vec{M})$$

Improper learning for non-stochastic control
Simchowitz, Singh, Hazan, COLT '20

NSC under partial observability

Non-stochastic control for **partially observed state:**

1. Compete w. Π_{DFC}
2. $O(T^{1/2})$ regret for known systems.
3. $O(T^{2/3})$ regret for unknown systems.
4. $O(T^{1/2})$ regret for *smoothed* noise, quadratic loss: "improper LQG" (first efficient algorithm even for stochastic setting)

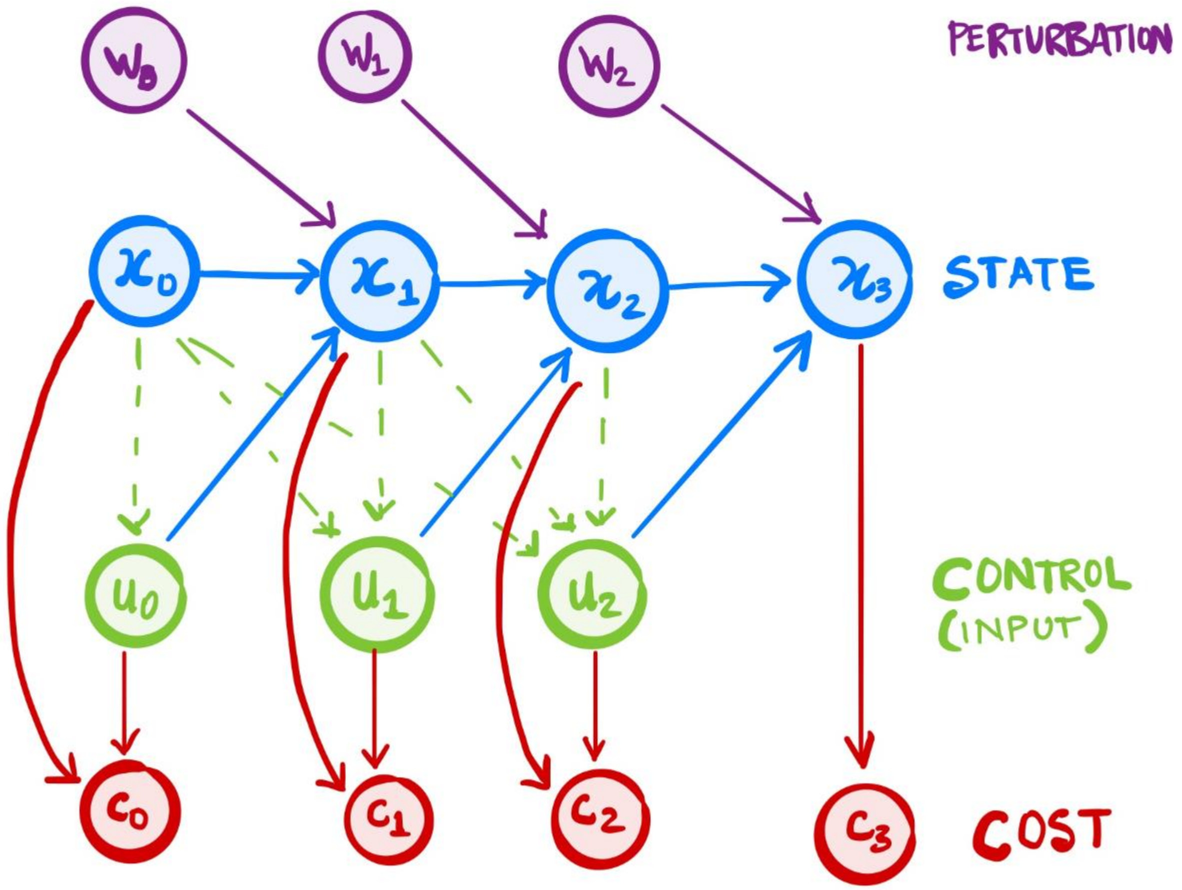
Via: Youla reparametrization, "Nature's y's", online gradient methods

Improper learning for non-stochastic control
Simchowitz, Singh, Hazan, COLT '20

Changing LDS

$$\min_{u(x)} \ \mathbb{E}\left[\sum_{t=1}^{T} c_t(x_t, u_t)\right]$$

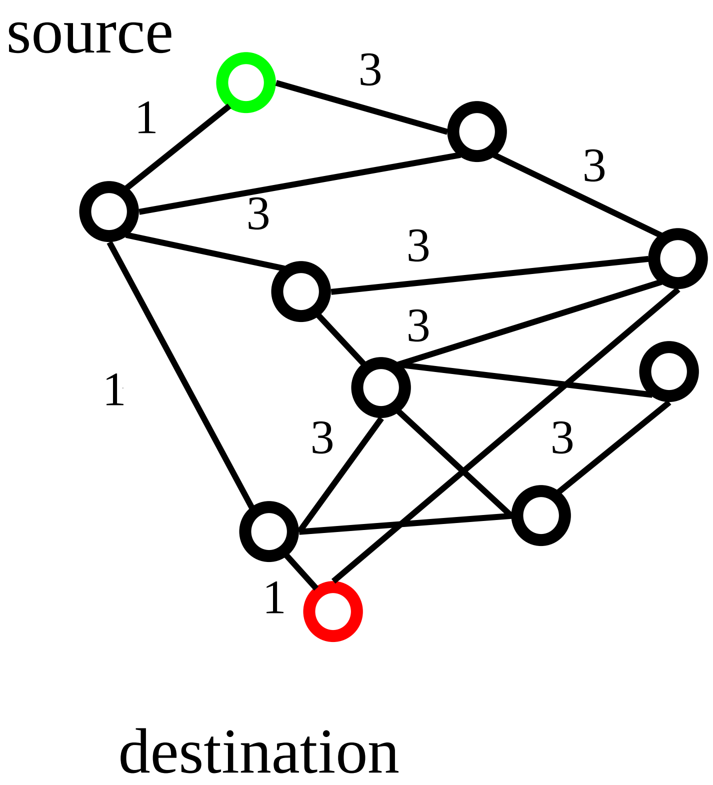
s.t. $x_{t+1} = A_t x_t + B_t u_t + w_t$



What's a reasonable metric?

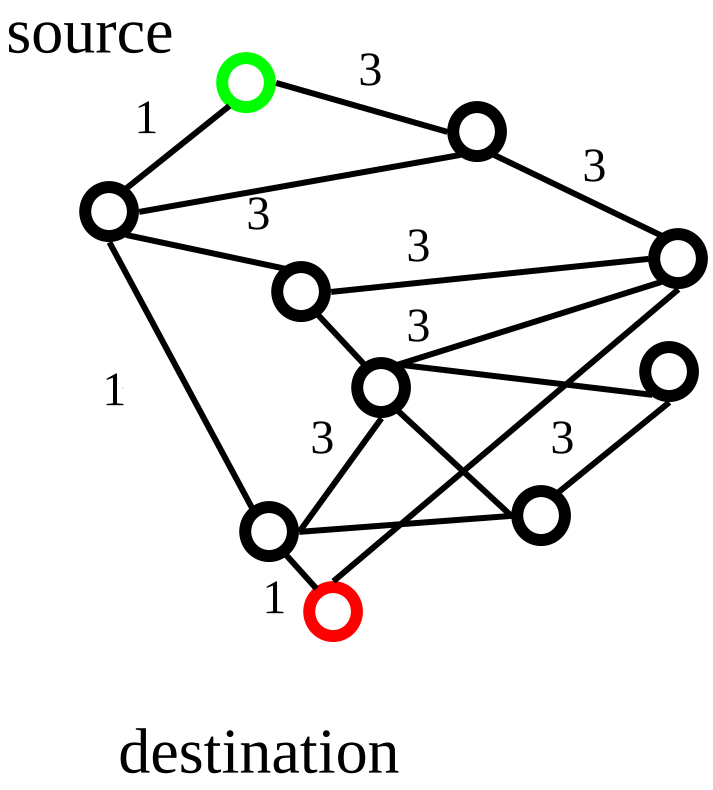
Let's go back to online convex optimization…

Learning in changing environments: online shortest paths



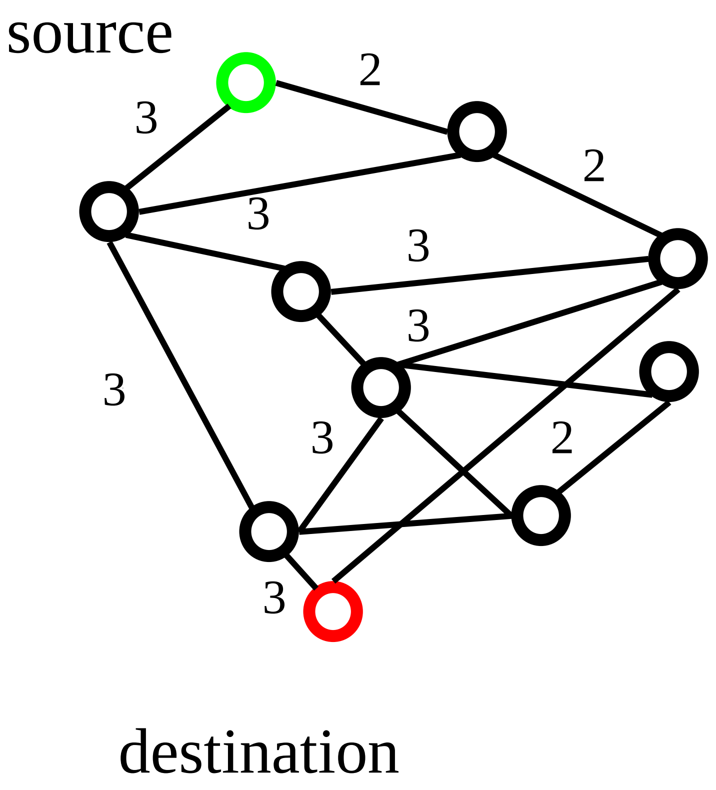
Learning in changing environment

Summer
congestion



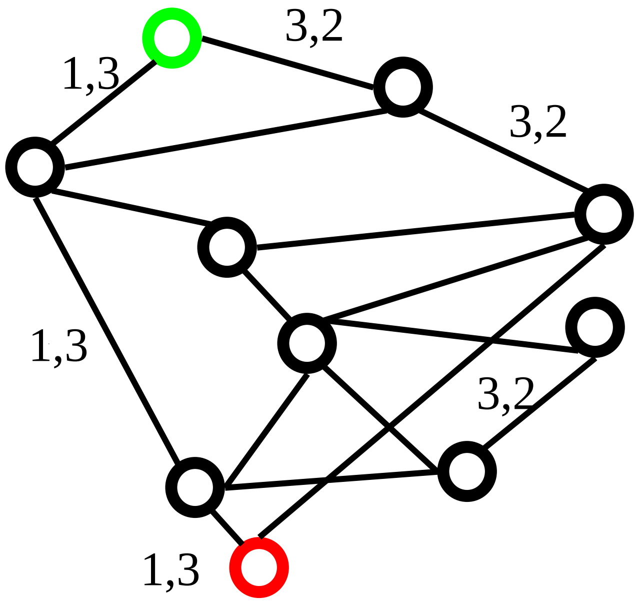
Learning in changing environment

Winter
congestion



Regret minimization (OGD, FTRL, ONS,…)

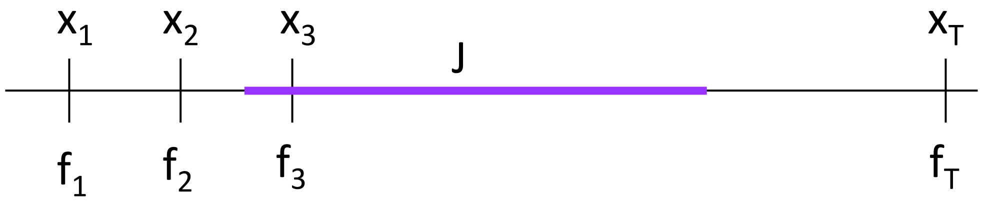
summer – optimal path.
winter – very slow shift
from p_1 to p_2



regret does not capture movement!

Convergence is good for regret, but…

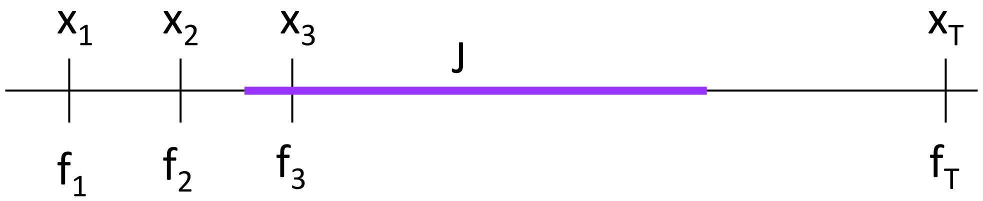
Adaptive Regret



$$\sum_{t \in J} f_t(x_t) - \min_{x_J^*} \sum_{t \in J} f_t(x_J^*)$$

Efficient learning for changing environments
Hazan, Seshadhri ICML '09

Adaptive Regret



Adaptive Regret = $\sup_J \left[\sum_{t \in J} f_t(x_t) - \min_{x^*_J} \sum_{t \in J} f_t(x^*_J) \right]$

- Max regret over all intervals
  - Different optimum x^*_J for every interval J
  - Captures movement of optimum as time progresses

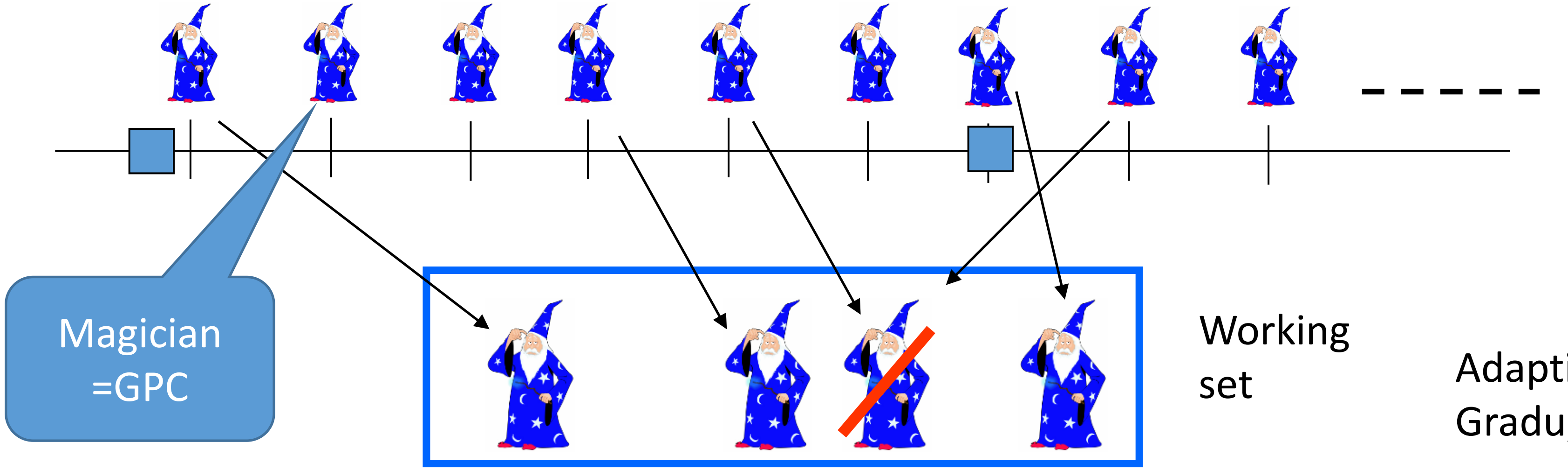
- We want Adaptive Regret = o(T)
  - In any interval of size ω(AR), algorithm converges to optimum (on smaller interval we cannot guarantee anything)
  - More general than "dynamic regret" and other notions

Efficient learning for changing environments
Hazan, Seshadhri ICML '09

Adaptive Regret for Control

$$\sup_I \left\{ \sum_{t\in I} c_t(x_t, u_t) - \min_{\pi\in\Pi_{DFC}} \left(\sum_{t\in I} c_t\left(\widehat{x_t}, \pi(\widehat{x_t})\right) \right) \right\} \leq \sqrt{L} \leq \sqrt{T}$$

- Maintain a working set of log(T) GPC algorithms
- Merge their control according to an exponential weighting scheme
- Adaptation of FLH (follow-the-leading-history) method for OCO
- log(T) overhead in running time & memory



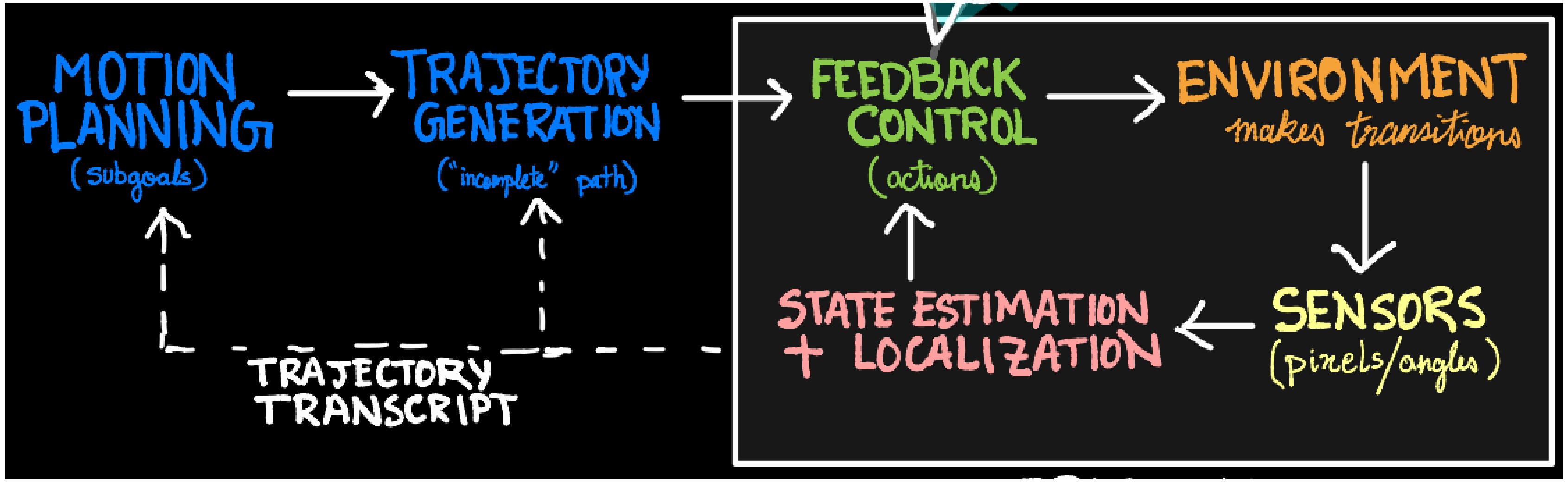
Adaptive Regret for Control of Time-Varying Dyn
Gradu, Hazan, Minasyan '20

Agenda

1. The basic paradigm of non-stochastic control:
   - Pre-tutorial on OCO
   - Setting
   - Performance metric
   - Methods
2. Extensions:
   unknown systems, partial observability, bandit feedback, black-box control, time-varying systems
3. Applications:
   adversarial noise design and controller verification, planning

How it fits in: Nonstochastic Control

Control-based strategies are often modular.



Convex Policy Parametrization for Linear Control

$$x_{h+1} = Ax_h + Bu_h + w_h$$

$$u_h = Kx_h \quad \text{vs.} \quad u_h = \sum_{i=1}^{\tau} M_i w_{h-i}$$

Controller verification

How can we certify a controllers' correct behavior?

→ Generate maximally adversarial online perturbation

Controller →

→ Observed performance

↑

Noise

Generating Adversarial Disturbances for Controller Verification
Ghai, Snyder, Majumdar, Hazan L4DC '21

Noise <-> control

Symmetric!

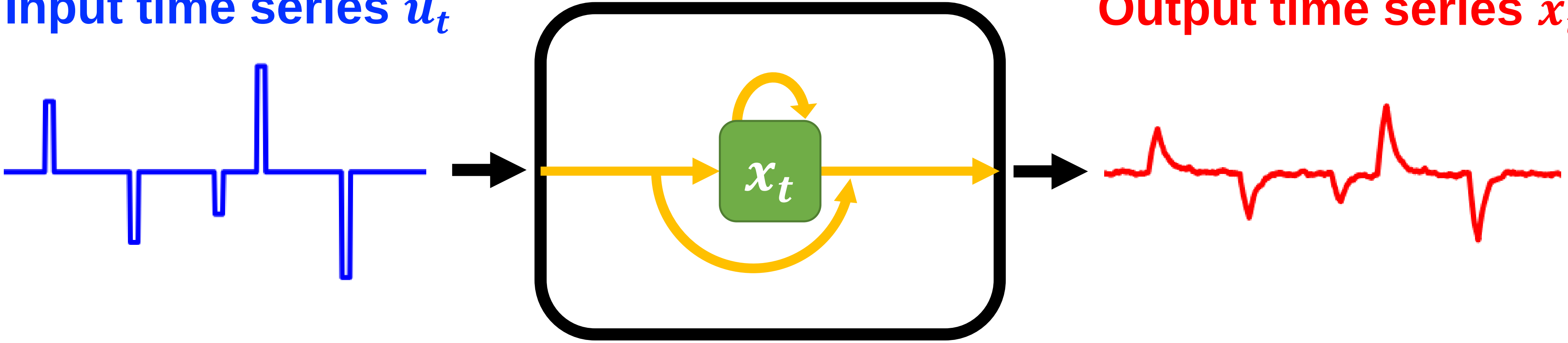
$$x_{t+1} = A_t x_t + B_t u_t + w_t$$

$$c_t(x_t, u_t)$$

Maximization vs. Min
Online Trust Region

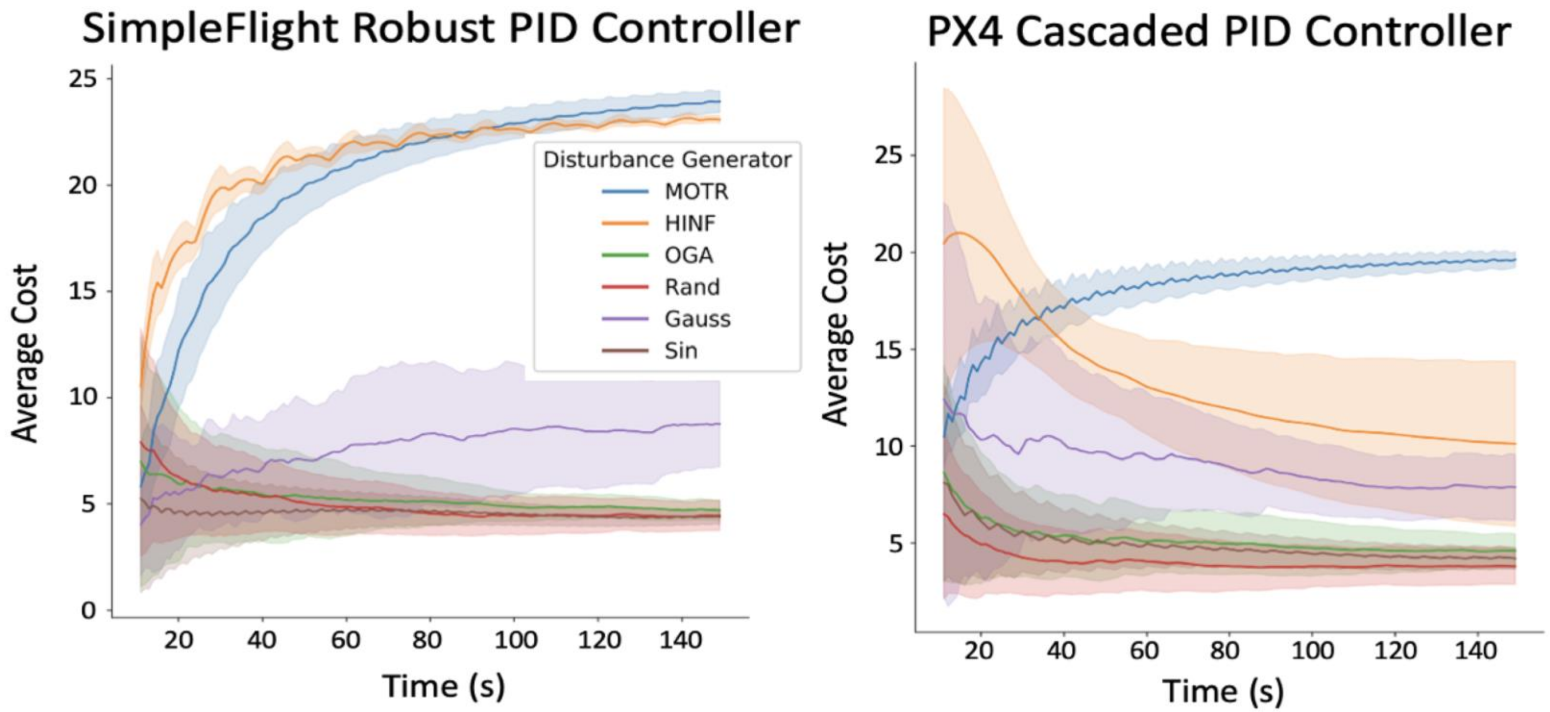
Input time series u_t

Output time series x_t

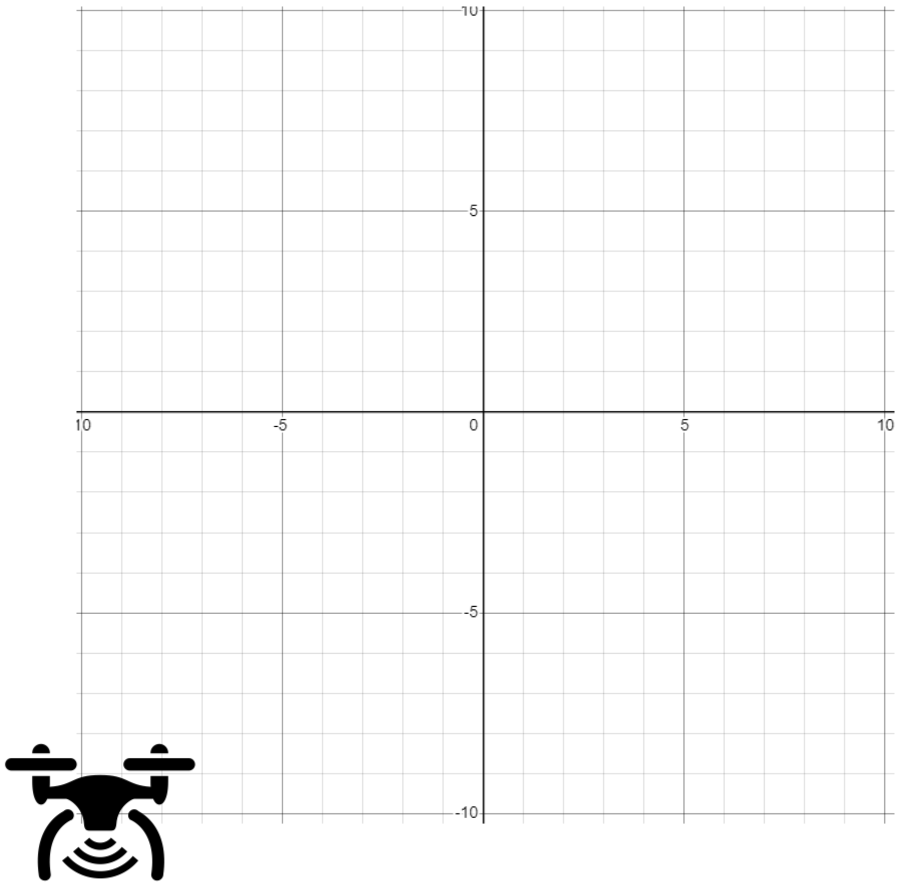


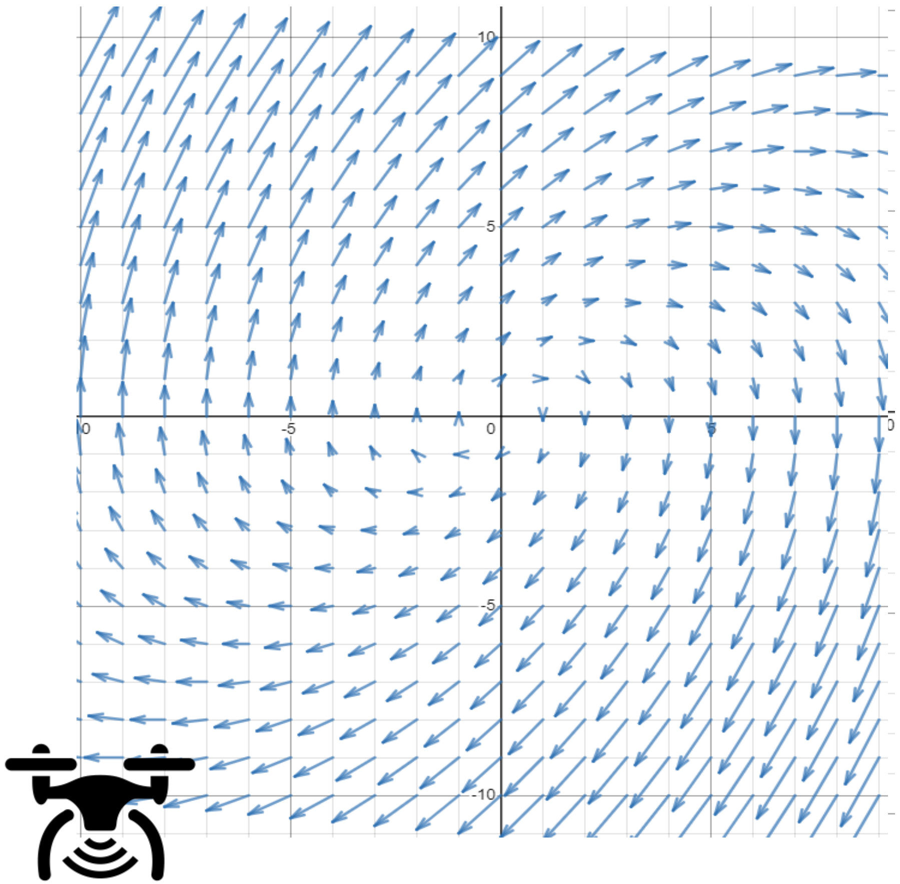
Generating Adversarial Disturbances for Controller Verification
Ghai, Snyder, Majumdar, Hazan L4DC '21

Experiments with airsim



Generating Adversarial Disturbances for Controller Verification
Ghai, Snyder, Majumdar, Hazan L4DC '21

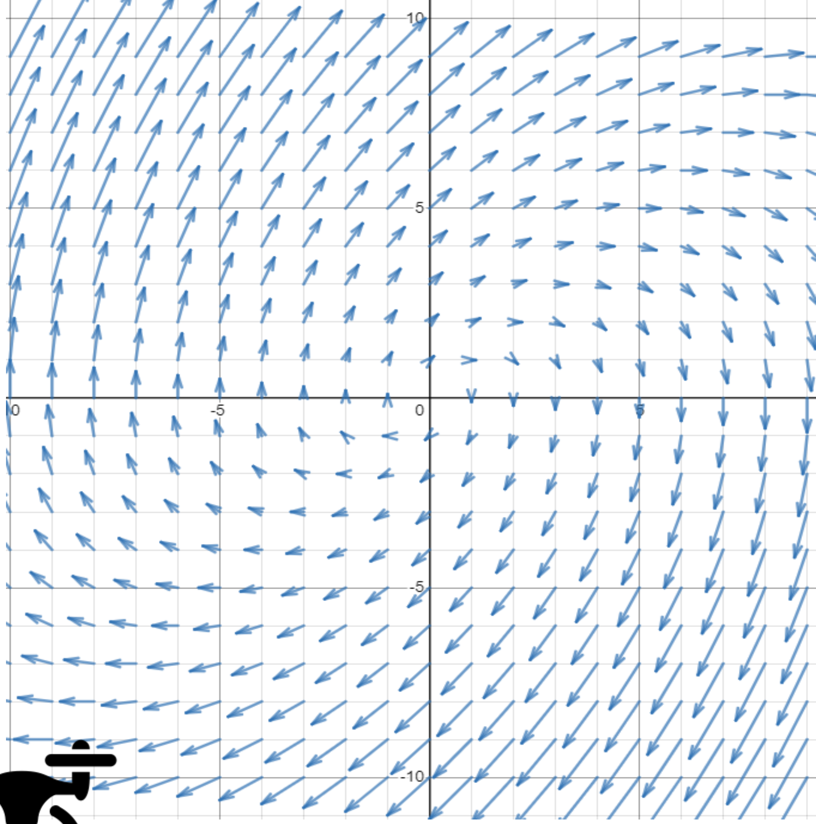

10
5
10
-5
0
5
10
-5
-10


10
5
0
-5
-10

Data-driven Planning

Learn an (adaptive) <u>policy</u>

+ given an <u>approximate model</u>
+ subject to changing, <u>unknown</u> perturbations
+ in a *handful* of <u>episodes</u>



Why?

+ Arises in real-world applications
+ Sandboxed setup for sim2real, meta-learning, policy transfer

Problem Setting

Episode 1 of T

State

Action

Learner has a model $f(x,u)$

Timestep 1 of H

Play action u_h

$x_{h+1} = f(x_h, u_h) + w_h$

Suffer $c(x_h, u_h)$

Episodic $Cost_t = \sum_h c_h(x_h, u_h)$

Compare:
Iterative LQR > *no perturbations*
Iterative LQG > *Gaussian perturbations*
Model Predictive Control > *One-shot*
Iterative Learning Control > *Same setup (here)*

Unknown, Nonstationary Perturbations

(***arbitrary***, *no dist. assumption)*
(changes every ***step****, every* ***episode****)*

Objective: *Planning Regret*

Inter-episodic learning
Best Overall Open-Loop Plan

Intra-episodic learning
Instance-optimal Adaptation

$$\sum_{t=1}^{T} Cost_t(Alg) \approx \min_{U^*_{1:H}} \sum_{t=1}^{T} \min_{\pi^*_t} Cost_t(U^*_{1:H} + \pi^*_t)$$

A regret minimization approach to iterative learning control
Agarwal, Hazan, Majumdar, Singh ICML ‘21

Planning Regret Bound

For time-varying linear dynamical system

Subject to arbitrary perturbation

An efficient gradient-based algorithm

$$\frac{1}{TH}\left(\sum_{t=1}^{T} Cost_t(Alg) - \min_{U^*_{1:H}} \sum_{t=1}^{T} \min_{\pi^*_t} Cost_t(U^*_{1:H} + \pi^*_t)\right) \leq \frac{1}{\sqrt{T}} + \frac{1}{\sqrt{H}}$$

Inter-episodic learning

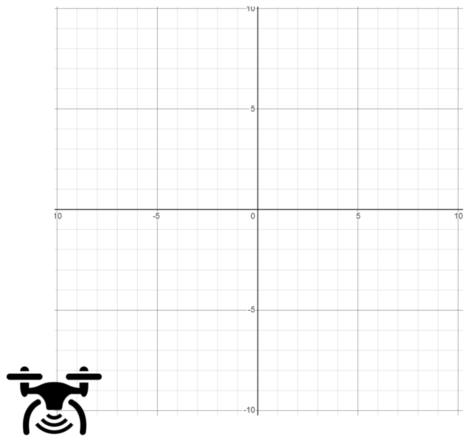
Best Overall Open-Loop Plan

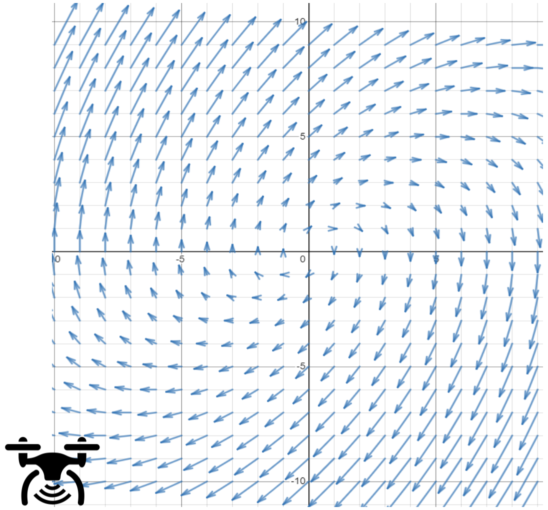
Intra-episodic learning

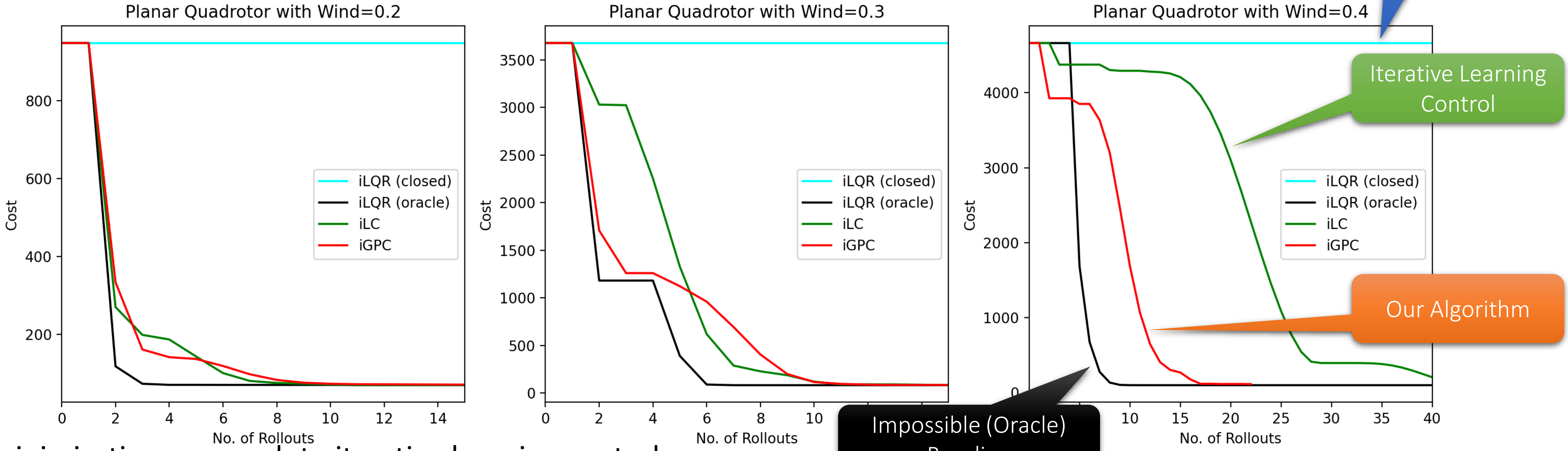
Instance-optimal Adaptation

A regret minimization approach to iterative learning control
Agarwal, Hazan, Majumdar, Singh ICML '21

Experiment 1: *Quadcopter in Wind*



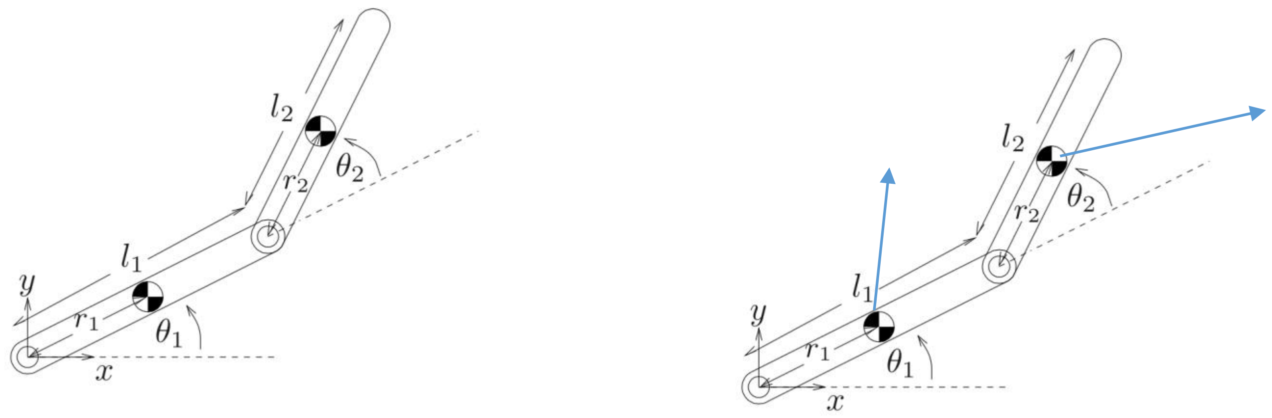


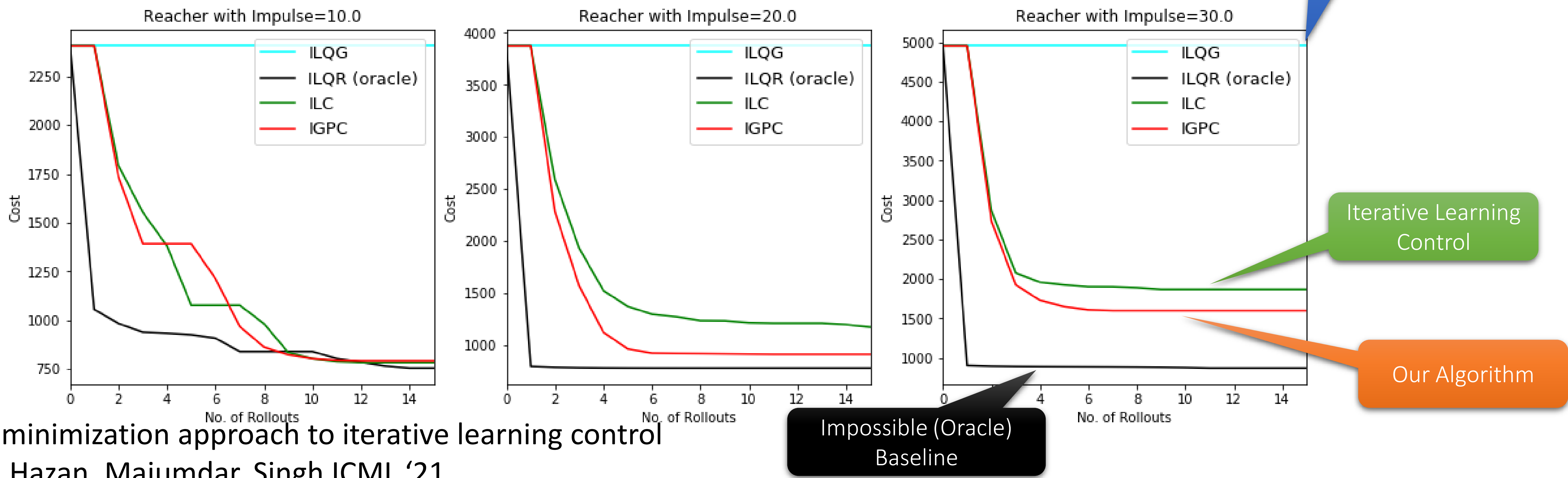


A regret minimization approach to iterative learning control

Agarwal, Hazan, Majumdar, Singh ICML ‘21

Experiment 2: *Reacher w. Impulses*





A regret minimization approach to iterative learning control
Agarwal, Hazan, Majumdar, Singh ICML '21

Agenda

1. The basic paradigm of non-stochastic control:
   - Pre-tutorial on OCO
   - Setting
   - Performance metric
   - Methods
2. Extensions:
   unknown systems, partial observability, bandit feedback, black-box control, time-varying systems
3. Applications:
   adversarial noise design and controller verification, planning

Summary

Emerging theory of online non-stochastic control

1. Performance metric, motivation, setting
2. Gradient-based regret-minimizing controllers (GPC)
3. Controlling unknown systems, partially observed & unknown
4. Adaptive regret for time varying systems
5. Black-box control
6. Applications: Controller verification, perturbation-resilient planning
7. More info on OCO/regret/adaptive-regret: https://ocobook.cs.princeton.edu/
   More info on NSC: https://sites.google.com/view/nsc-tutorial/home
   Code: https://www.deluca.fyi/

Thank you !