Online Reinforcement Learning in Structured Environments

Sayak Ray Chowdhury

Department of Electrical Communication Engineering
Indian Institute of Science

June 18, 2018

Motivation

Reinforcement Learning is concerned with learning to take actions to maximize rewards, by trial and error, in environments that can evolve in response to actions

Motivation

Reinforcement Learning is concerned with learning to take actions to maximize rewards, by trial and error, in environments that can evolve in response to actions

- Traditional Search Goal: Find a policy with high total reward using as few interactions with the environment as possible

Motivation

Reinforcement Learning is concerned with learning to take actions to maximize rewards, by trial and error, in environments that can evolve in response to actions

- Traditional Search Goal: Find a policy with high total reward using as few interactions with the environment as possible
- Optimization Goal: Maximize total reward / Minimize regret (shortfall) in total reward compared to an optimal policy

Motivation

Reinforcement Learning is concerned with learning to take actions to maximize rewards, by trial and error, in environments that can evolve in response to actions

- Traditional Search Goal: Find a policy with high total reward using as few interactions with the environment as possible
- Optimization Goal: Maximize total reward / Minimize regret (shortfall) in total reward compared to an optimal policy
  - **Applications:** Recommendation systems/ Sequential investment/ Dynamic resource allocation in communication systems ....
  - No separate budget to purely exploring the unknown environment
  - Exploration and Exploitation must be carefully balanced

Intended Goal of Thesis

Develop RL algorithms for Regret Minimization in large structured (unknown) environments

Intended Goal of Thesis

Develop RL algorithms for Regret Minimization in large structured (unknown) environments

- **Part 1:** Online learning in large scale **Multi-armed Bandits**
  - Nonparametric model: Infinite number of actions
  - Generalization across arms: Kernelized structure

Intended Goal of Thesis

Develop RL algorithms for Regret Minimization in large structured (unknown) environments

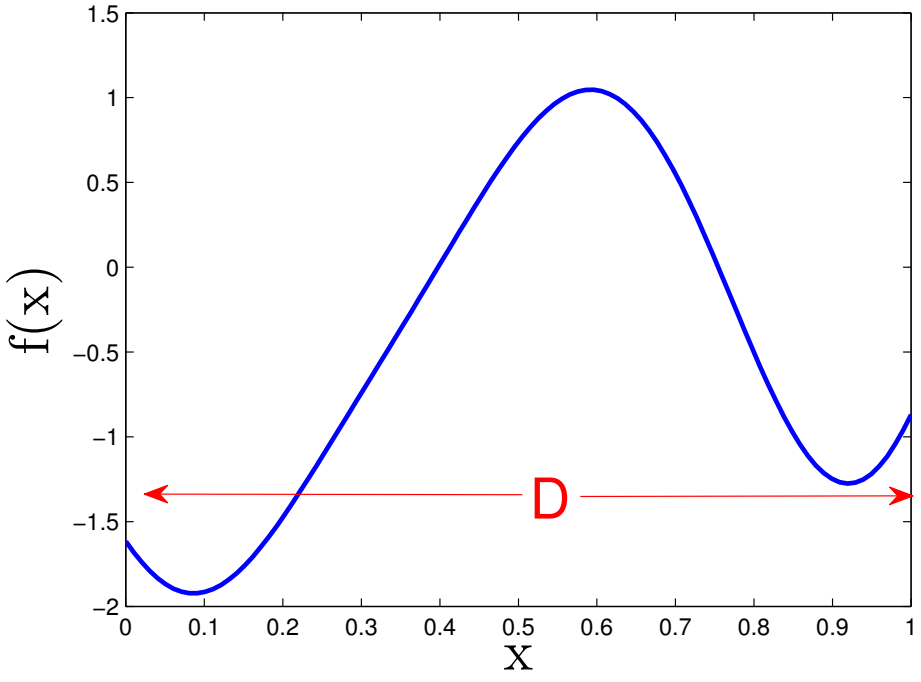
- **Part 1:** Online learning in large scale **Multi-armed Bandits**
  - Nonparametric model: Infinite number of actions
  - Generalization across arms: Kernelized structure
- **Part 2:** Online learning in large unknown **Markov Decision Processes**
  - Nonparametric MDP model: Uncertainty is represented over Infinite dimensional function classes via Kernelized structure

Part 1: Online Learning in Kernelized Multi-armed Bandits[1]

[1]S. R. Chowdhury and A. Gopalan, *On kernelized multi-armed bandits*, In Proceedings of the 34th International Conference on Machine Learning (**ICML**), pp. 844853, 2017.

Problem Statement

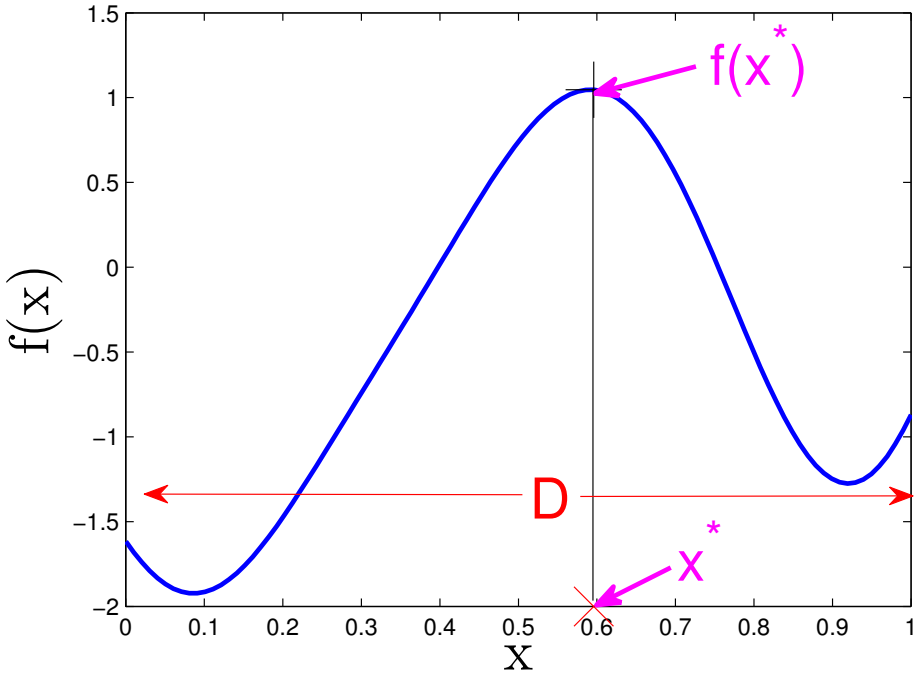
Sequentially Maximize $f : D \to \mathbb{R}$



- f **unknown**, $D \subset \mathbb{R}^d$

Problem Statement

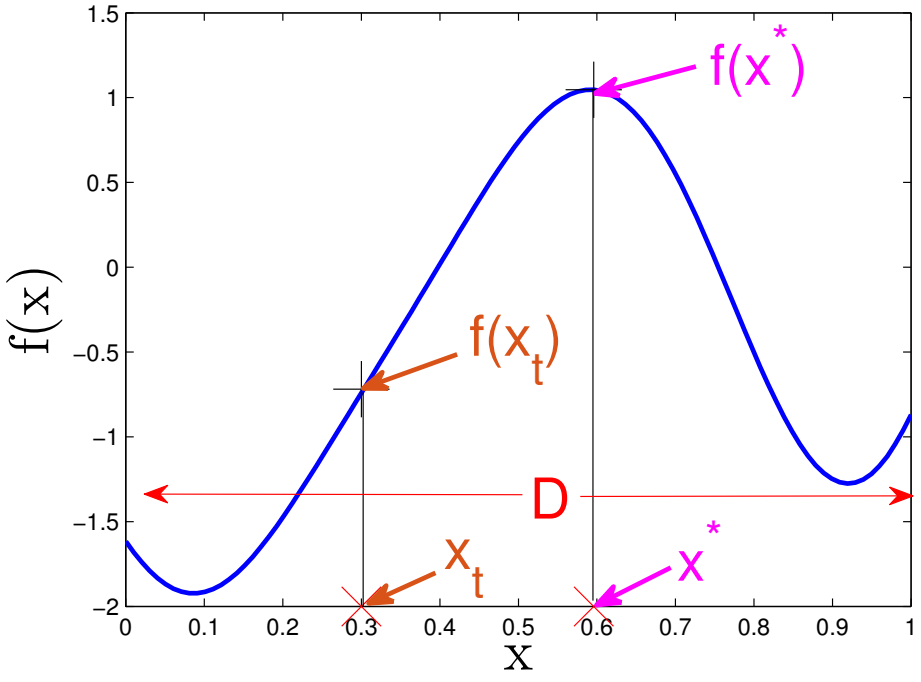
Sequentially Maximize $f : D \to \mathbb{R}$



- f **unknown**, $D \subset \mathbb{R}^d$
- $x^\star \in \underset{x \in D}{\operatorname{argmax}}\, f(x)$

Problem Statement

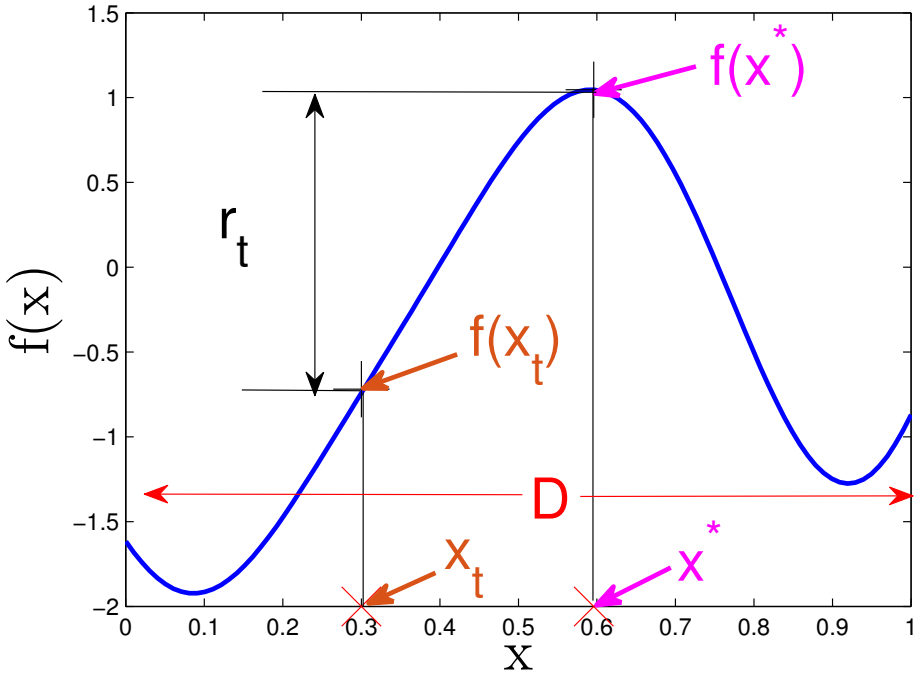
Sequentially Maximize $f : D \to \mathbb{R}$



- f **unknown**, $D \subset \mathbb{R}^d$
- $x^\star \in \underset{x \in D}{\operatorname{argmax}} f(x)$
- At each round t:
  - Learner chooses $x_t \in D$ based on past
  - Observes noisy reward $y_t = f(x_t) + \varepsilon_t$

Problem Statement

Sequentially Maximize $f : D \to \mathbb{R}$



- f **unknown**, $D \subset \mathbb{R}^d$
- $x^\star \in \operatorname*{argmax}_{x \in D} f(x)$
- At each round t:
  - Learner chooses $x_t \in D$ based on past
  - Observes noisy reward $y_t = f(x_t) + \varepsilon_t$

Performance Metric

- Minimize Cumulative Regret: $\sum_{t=1}^{T} \Big(f(x^\star) - f(x_t) \Big)$

Assumptions

- Smoothness: f lies in **R**eproducing **K**ernel **H**ilbert **S**pace (**RKHS**) of functions $D \to \mathbb{R}$
  - Positive semi-definite kernel function $k : D \times D \to \mathbb{R}$ (known)
  - Reproducing property: $f(x) = \langle f, k(x, \cdot) \rangle_k$
  - Induces smoothness: $|f(x) - f(y)| \leq \|f\|_k \, \|k(x, \cdot) - k(y, \cdot)\|_k$

Assumptions

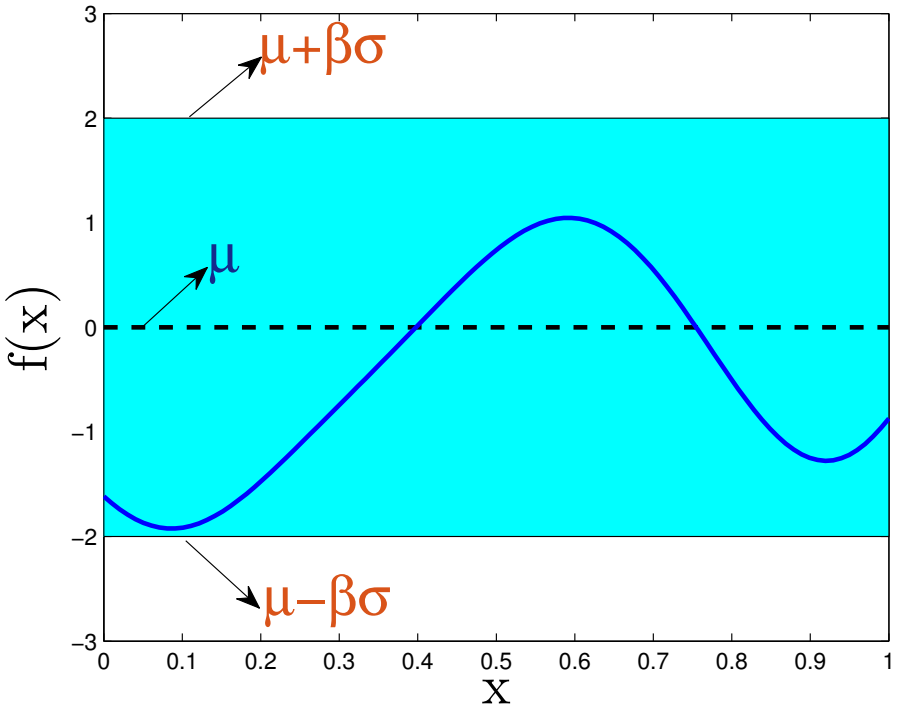
- Smoothness: f lies in **R**eproducing **K**ernel **H**ilbert **S**pace (**RKHS**) of functions $D \to \mathbb{R}$
  - Positive semi-definite kernel function $k : D \times D \to \mathbb{R}$ (known)
  - Reproducing property: $f(x) = \langle f, k(x, \cdot) \rangle_k$
  - Induces smoothness: $|f(x) - f(y)| \leq \|f\|_k \, \|k(x, \cdot) - k(y, \cdot)\|_k$
- Example Kernels
  - Squared Exponential kernel: $k(x, y) = \exp\left(\frac{-\|x-y\|_2^2}{2l^2}\right)$
  - Linear Kernel: $k(x, y) = x^T y$
    - Maximize $f(x) = \theta^T x$, $\theta \in \mathbb{R}^d$ unknown

Assumptions

- Smoothness: f lies in **R**eproducing **K**ernel **H**ilbert **S**pace (**RKHS**) of functions $D \to \mathbb{R}$
  - Positive semi-definite kernel function $k : D \times D \to \mathbb{R}$ (known)
  - Reproducing property: $f(x) = \langle f, k(x, \cdot) \rangle_k$
  - Induces smoothness: $|f(x) - f(y)| \leq \|f\|_k \|k(x, \cdot) - k(y, \cdot)\|_k$
- Example Kernels
  - Squared Exponential kernel: $k(x, y) = \exp\left(\frac{-\|x-y\|_2^2}{2l^2}\right)$
  - Linear Kernel: $k(x, y) = x^T y$
    - Maximize $f(x) = \theta^T x$, $\theta \in \mathbb{R}^d$ unknown
- Noise ε_t is zero mean, Sub-Gaussian

Algorithm Design Philosophy

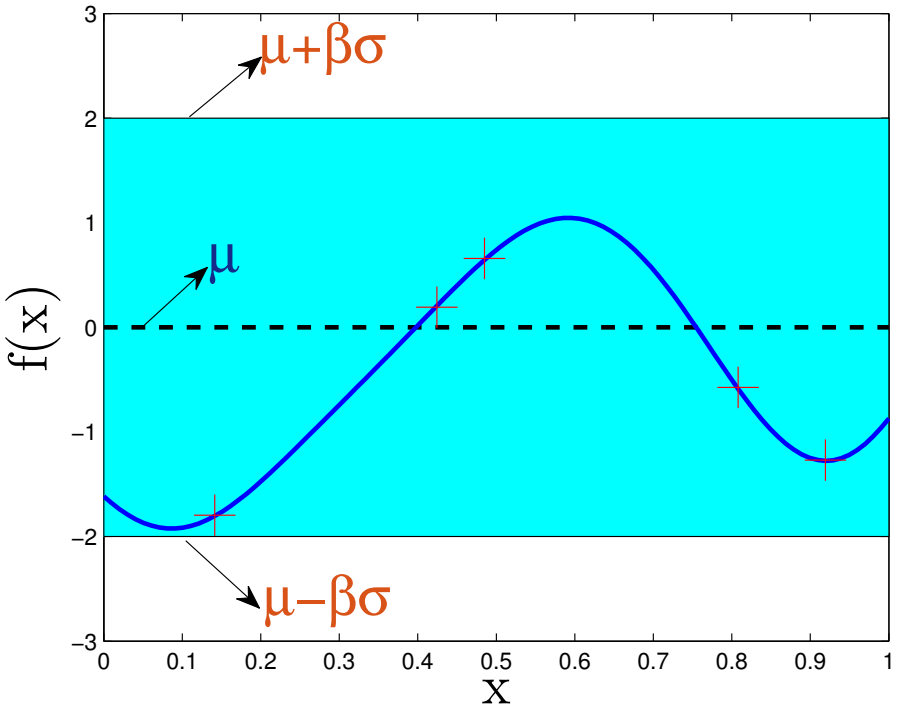
Key Idea: Represent uncertainty over f using **G**aussian **P**rocess (**GP**)



- Assume Gaussian Process Prior $GP\Big(0, k(x, y)\Big)$
- Assume Gaussian Noise $\varepsilon_t \sim \mathcal{N}(0, \lambda)$

Algorithm Design Philosophy

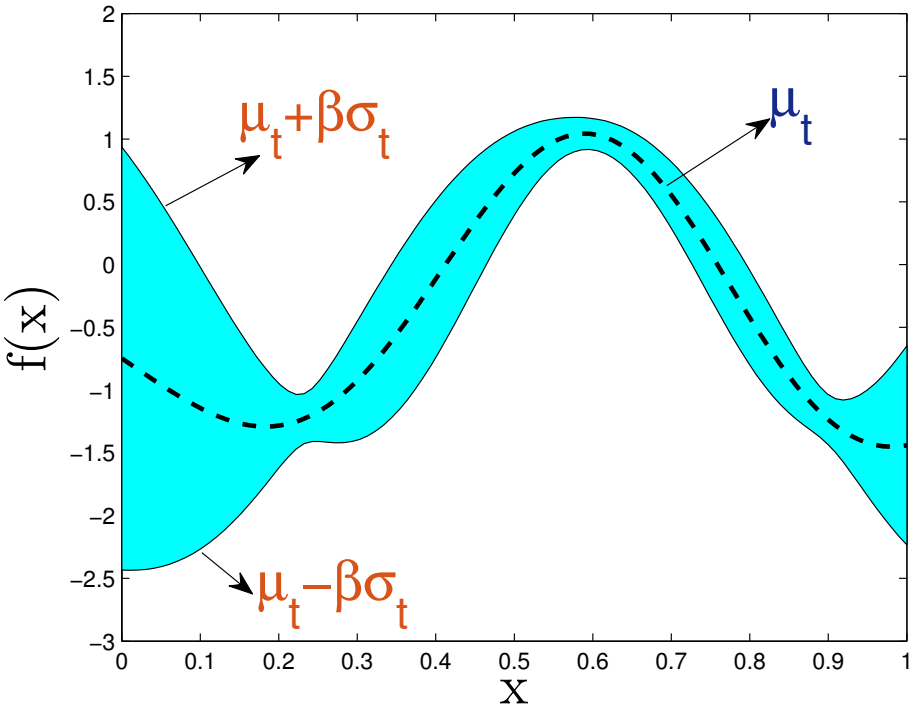
Key Idea: Represent uncertainty over f using **G**aussian **P**rocess (**GP**)



- Assume Gaussian Process Prior
  $GP\Big(0, k(x, y)\Big)$
- Assume Gaussian Noise
  $\varepsilon_t \sim \mathcal{N}(0, \lambda)$
- Observe noisy rewards
  $y_{1:t} = [y_1, \ldots, y_t]$, where
  $y_t = f(x_t) + \varepsilon_t$

Algorithm Design Philosophy

Key Idea: Represent uncertainty over f using **G**aussian **P**rocess (**GP**)



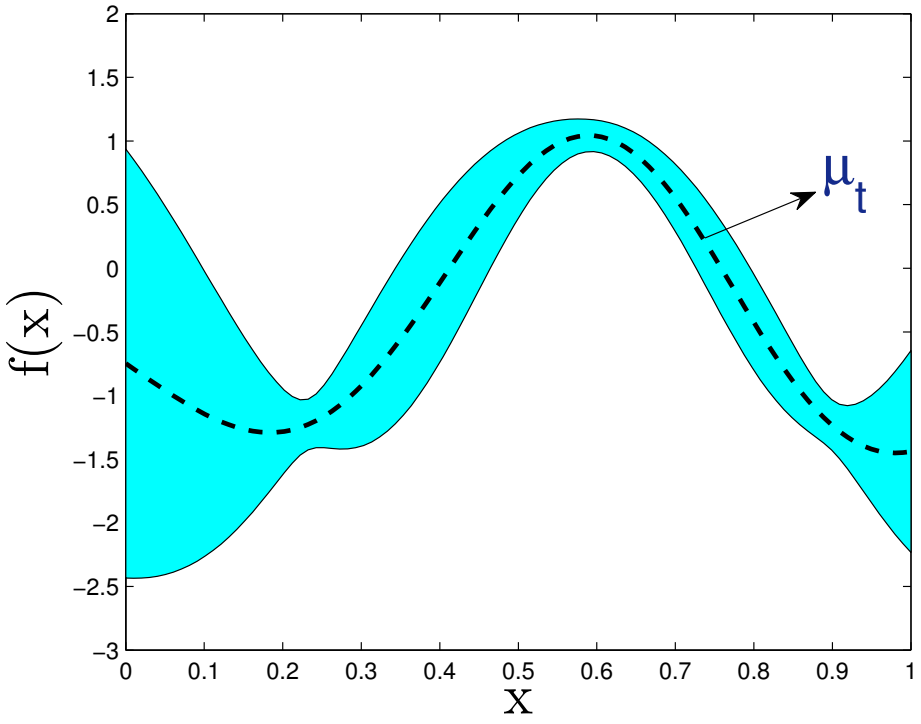
- Assume Gaussian Process Prior $GP\Big(0, k(x,y)\Big)$
- Assume Gaussian Noise $\varepsilon_t \sim \mathcal{N}(0, \lambda)$
- Observe noisy rewards $y_{1:t} = [y_1, \ldots, y_t]$, where $y_t = f(x_t) + \varepsilon_t$

Posterior of f after t rounds: $GP(\mu_t(x), k_t(x,y))$

$$\begin{aligned} \mu_t(x) &= k_t(x)^T (K_t + \lambda I)^{-1} y_{1:t} \\ k_t(x,y) &= k(x,y) - k_t(x)^T (K_t + \lambda I)^{-1} k_t(y) \end{aligned}$$

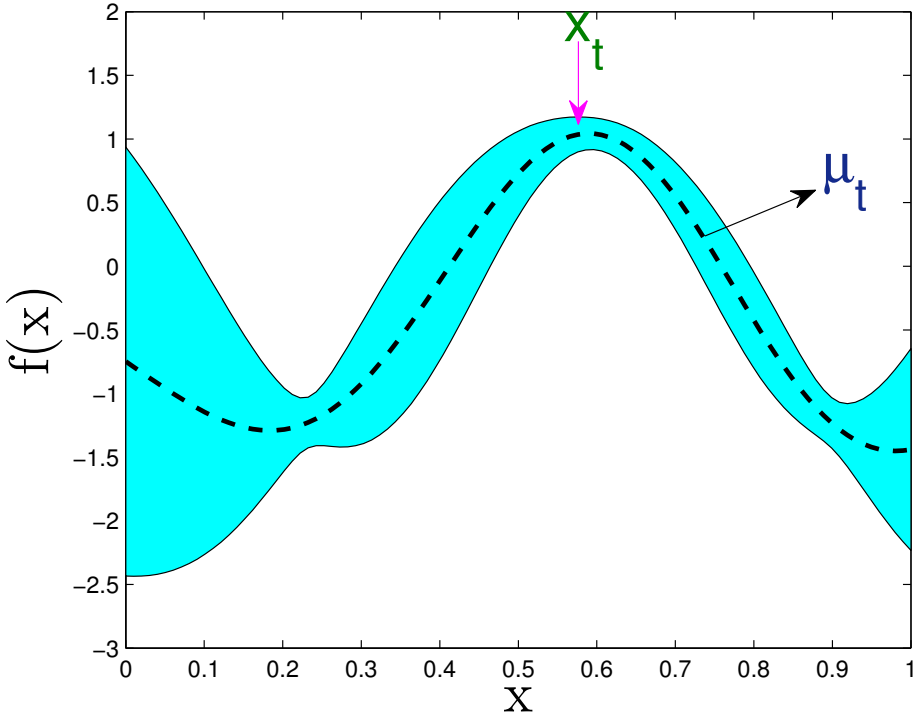
Algorithm 1: Improved GP-UCB (IGP-UCB)

Key Idea: Play the arm with highest Upper Confidence Bound (UCB)



Algorithm 1: Improved GP-UCB (IGP-UCB)

Key Idea: Play the arm with highest Upper Confidence Bound (UCB)

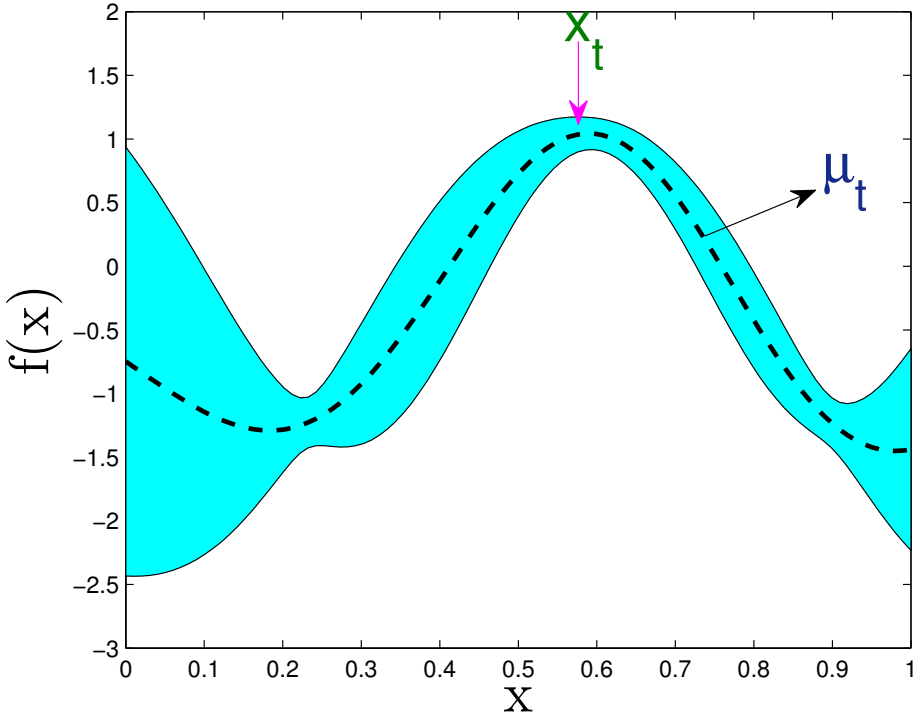


At each round t, play:

$$x_t = \underset{x \in D}{\operatorname{argmax}}\, \mu_t(x) + \beta_t \sigma_t(x)$$

Algorithm 1: Improved GP-UCB (IGP-UCB)

Key Idea: Play the arm with highest Upper Confidence Bound (UCB)



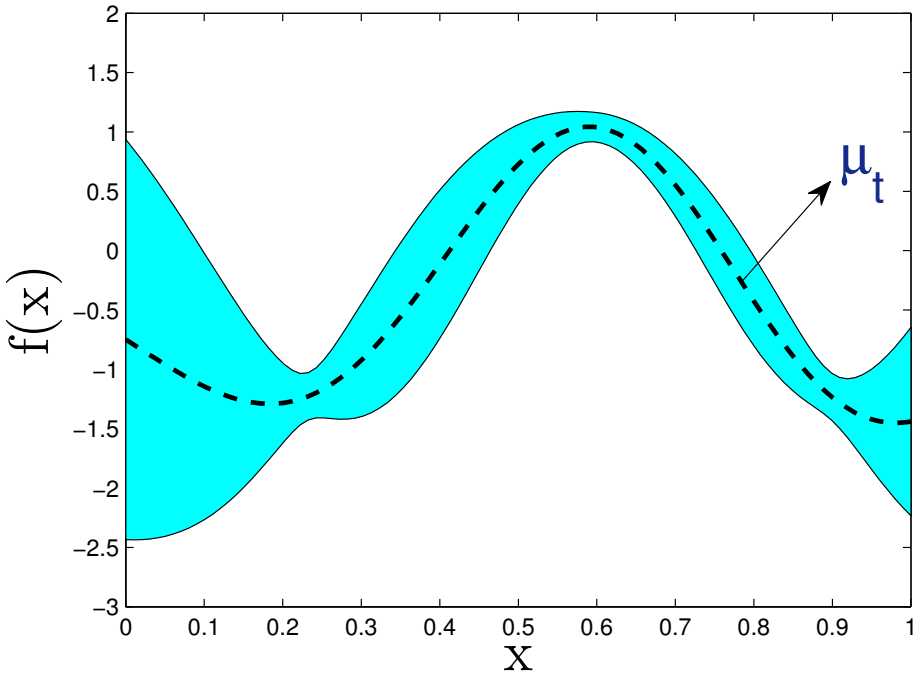
At each round t, play:

$$x_t = \underset{x \in D}{\operatorname{argmax}}\, \mu_t(x) + \beta_t \sigma_t(x)$$

- β_t trades off b/w **exploration** and **exploitation**
- First appeared as **GP-UCB** (Srinivas et al., ICML 2010) $\rightarrow$ We Reduced (Improved) width (β_t) of confidence interval

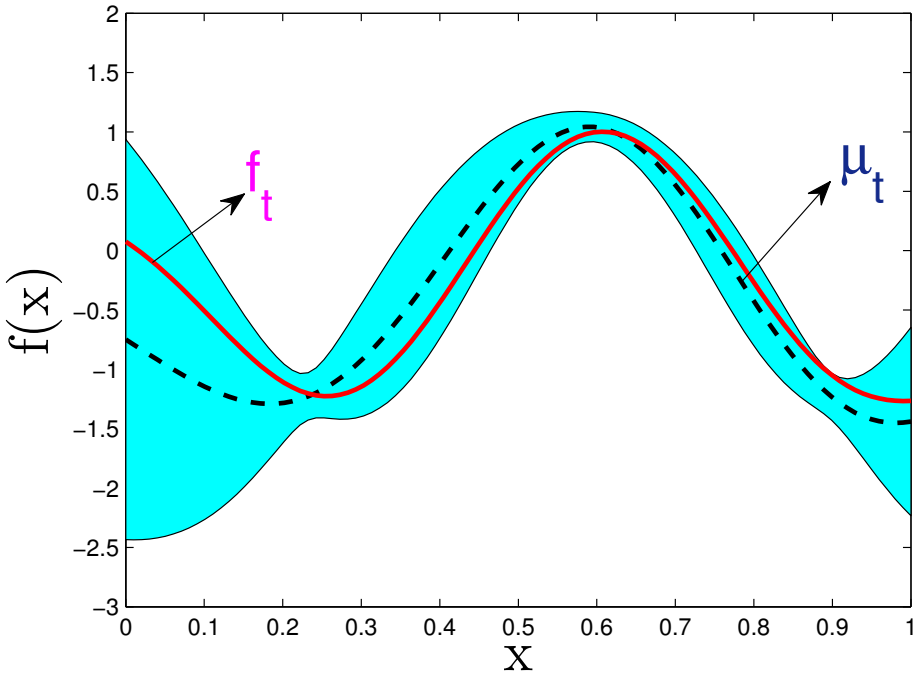
Algorithm 2: Gaussian Process Thompson Sampling (GP-TS)

Key Idea: Sample a random function and play its maximizer



Algorithm 2: Gaussian Process Thompson Sampling (GP-TS)

Key Idea: Sample a random function and play its maximizer

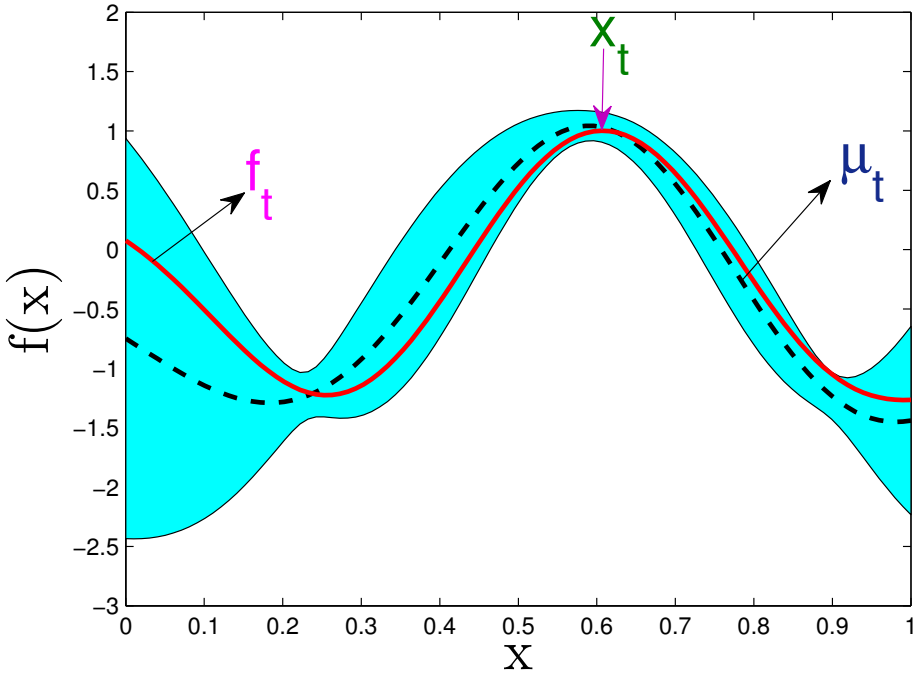


At each round t:

- Sample f_t from posterior of f

Algorithm 2: Gaussian Process Thompson Sampling (GP-TS)

Key Idea: Sample a random function and play its maximizer



At each round t:

- Sample f_t from posterior of f
- Play $x_t = \underset{x \in D_t}{\operatorname{argmax}}\, f_t(x)$

$D_t \subset D$: suitably chosen Discretization sets

Regret Bounds [ICML, 2017]

Result 1

Cumulative regret of **IGP-UCB** is $O\left(\sqrt{T}(\sqrt{\gamma_T} + \gamma_T)\right)$ *with high probability (whp)*

- γ_T is **Maximum Information Gain** about f after T rounds and quantifies Reduction in uncertainty after observing rewards
- Squared Exponential kernel: $\gamma_T = O((\ln T)^{d+1}) \rightarrow$ **Sublinear** regret

Regret Bounds [ICML, 2017]

Result 1

Cumulative regret of **IGP-UCB** is $O\left(\sqrt{T}(\sqrt{\gamma_T} + \gamma_T)\right)$ *with high probability (whp)*

- γ_T is **Maximum Information Gain** about f after T rounds and quantifies Reduction in uncertainty after observing rewards
- Squared Exponential kernel: $\gamma_T = O((\ln T)^{d+1}) \rightarrow$ **Sublinear** regret
- Cumulative regret of **GP-UCB** (Srinivas et al., ICML 2010) is $O\left(\sqrt{T}(\sqrt{\gamma_T} + \gamma_T \ln^{3/2} T)\right)$ and so we **improve** by $O(\ln^{3/2} T)$!

Regret Bounds [ICML, 2017]

Result 1

Cumulative regret of **IGP-UCB** is $O\left(\sqrt{T}(\sqrt{\gamma_T} + \gamma_T)\right)$ *with high probability (whp)*

- γ_T is **Maximum Information Gain** about f after T rounds and quantifies Reduction in uncertainty after observing rewards
- Squared Exponential kernel: $\gamma_T = O((\ln T)^{d+1}) \rightarrow$ **Sublinear** regret
- Cumulative regret of **GP-UCB** (Srinivas et al., ICML 2010) is $O\left(\sqrt{T}(\sqrt{\gamma_T} + \gamma_T \ln^{3/2} T)\right)$ and so we **improve** by $O(\ln^{3/2} T)$!

Result 2

- Cumulative regret of **GP-TS** is $O\left(\sqrt{Td\ln(dT)}(\sqrt{\gamma_T} + \gamma_T)\right)$ *whp*
- First **frequentist** regret guarantee of TS in the non-parametric setting of infinite action spaces

Recovering Regret Bounds for Linear Bandits

Linear Kernel

- $k(x, y) = x^T y$
- $f(x) = \theta^T x$, $\theta \in \mathbb{R}^d$ unknown parameter
- **Maximum Information Gain** $\gamma_T = O(d \ln T)$
- Regret of IGP-UCB is $\tilde{O}(d\sqrt{T})$ and GP-TS is $\tilde{O}(d^{3/2}\sqrt{T})$

Recovering Regret Bounds for Linear Bandits

Linear Kernel

- $k(x, y) = x^T y$
- $f(x) = \theta^T x$, $\theta \in \mathbb{R}^d$ unknown parameter
- **Maximum Information Gain** $\gamma_T = O(d \ln T)$
- Regret of IGP-UCB is $\tilde{O}(d\sqrt{T})$ and GP-TS is $\tilde{O}(d^{3/2}\sqrt{T})$

Exactly recovers regrets of **Linear Bandit** algorithms (Abbasi-Yadkori et al., NIPS 2011, Agrawal and Goyal, ICML 2013)

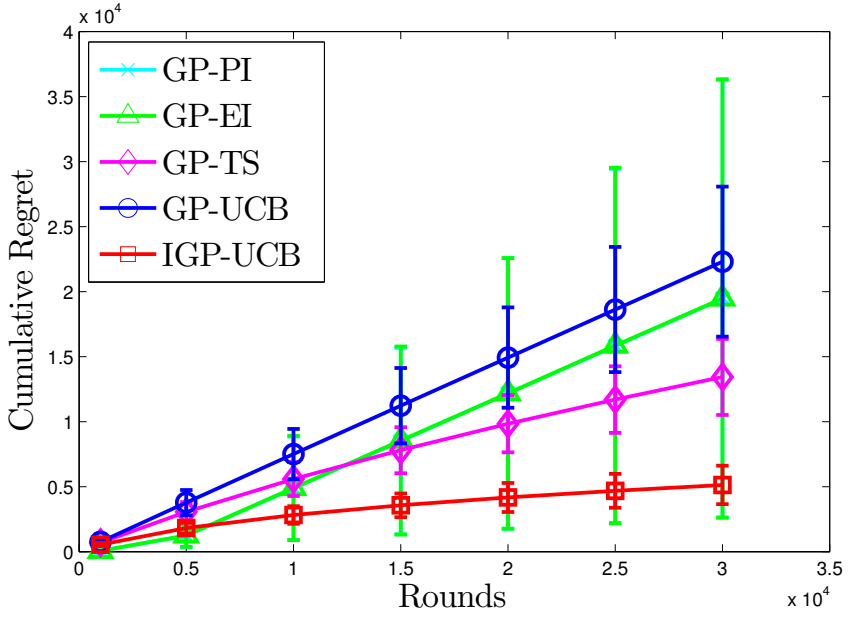
Numerical Results

Algorithms Compared:

1. GP-Expected Improvement (Močkus, 1975)
2. GP-Probabilistic Improvement (Kushner, 1964)
3. GP-UCB (Srinivas et al., 2010)
4. IGP-UCB (this work)
5. GP-TS (this work)

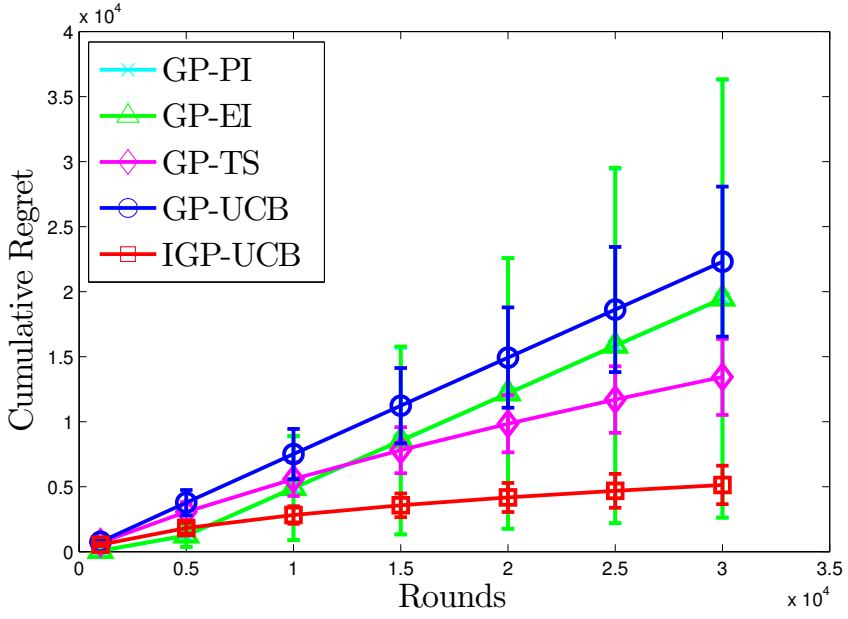
Numerical Results

f sampled from RKHS
(Squared Exponential kernel)



Numerical Results

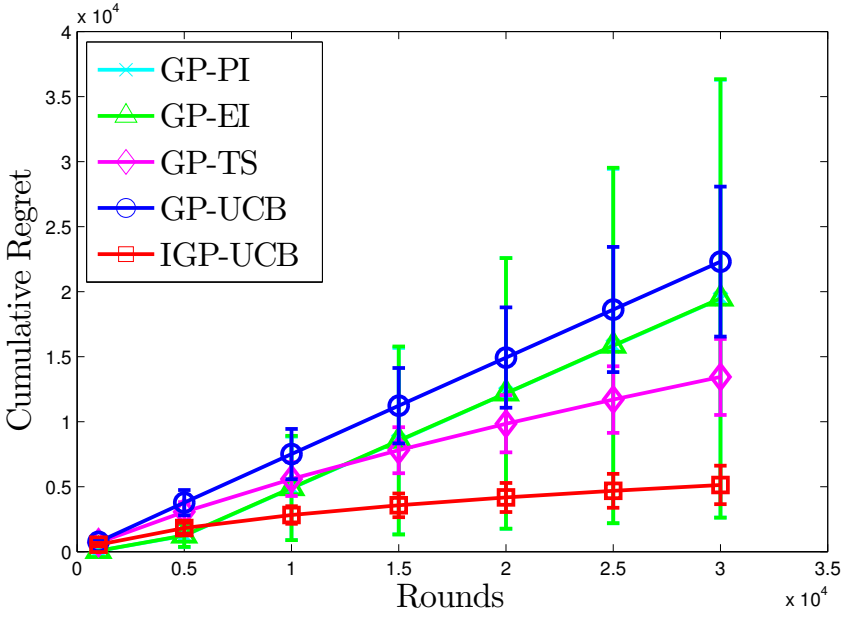
f sampled from RKHS
(Squared Exponential kernel)



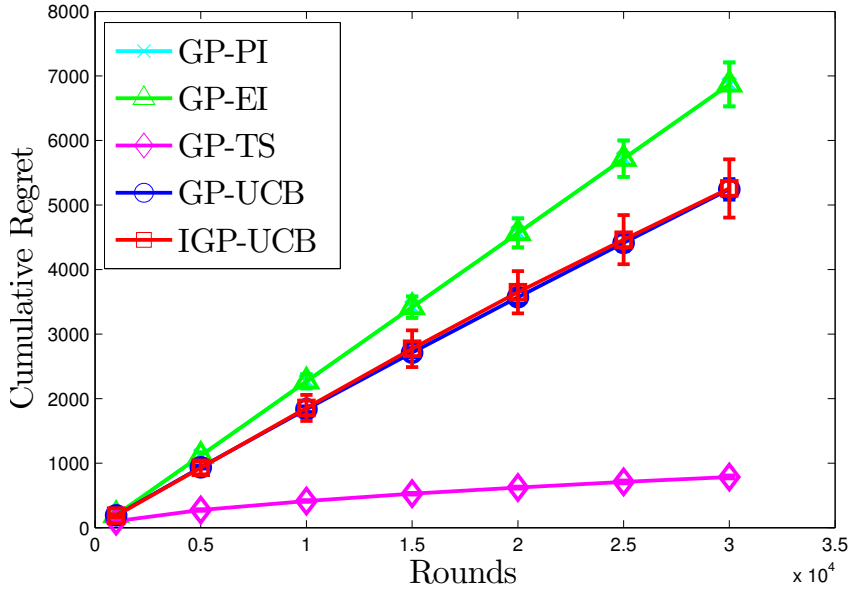
- IGP-UCB improves over GP-UCB ☺☺
- GP-TS fares reasonably well ☺

Numerical Results

f sampled from RKHS
(Squared Exponential kernel)



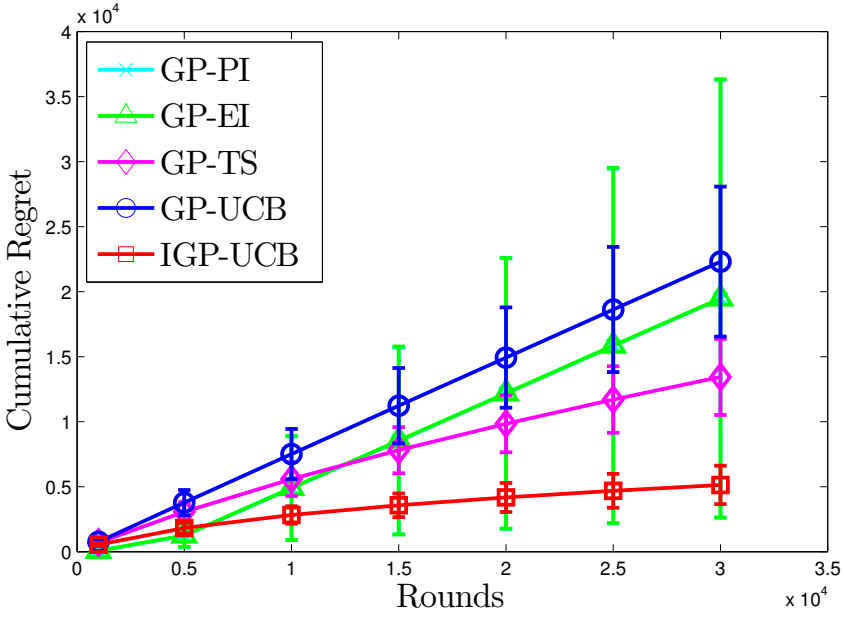
Temperature Sensor Data
(Intel Berkeley Research lab)



- IGP-UCB improves over GP-UCB ☺☺
- GP-TS fares reasonably well ☺

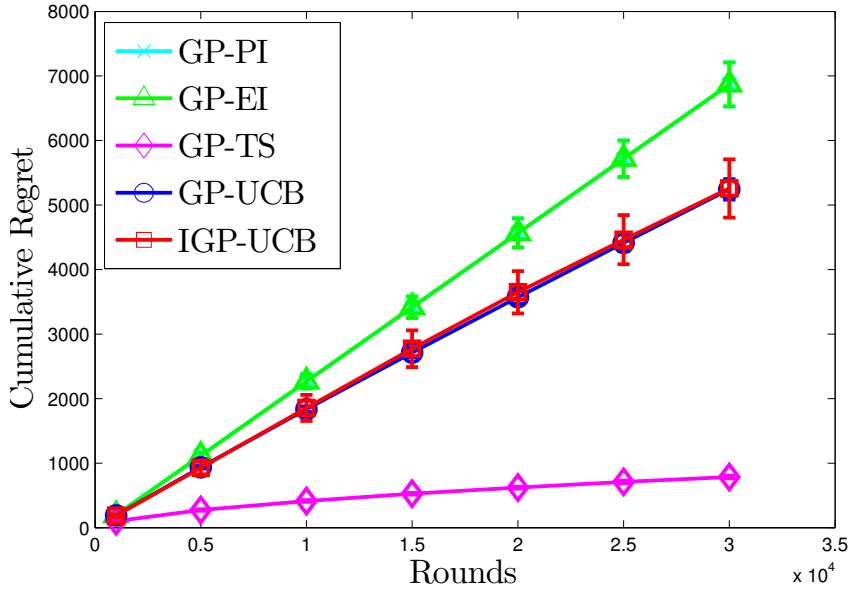
Numerical Results

f sampled from RKHS
(Squared Exponential kernel)



- IGP-UCB improves over GP-UCB ☺☺
- GP-TS fares reasonably well ☺

Temperature Sensor Data
(Intel Berkeley Research lab)



- IGP-UCB performs similar to GP-UCB ✓
- GP-TS performs the best ☺

Key Technique: Posterior Concentration

Lemma: Concentration of Posterior Distribution

For all t and for all $x \in D$:

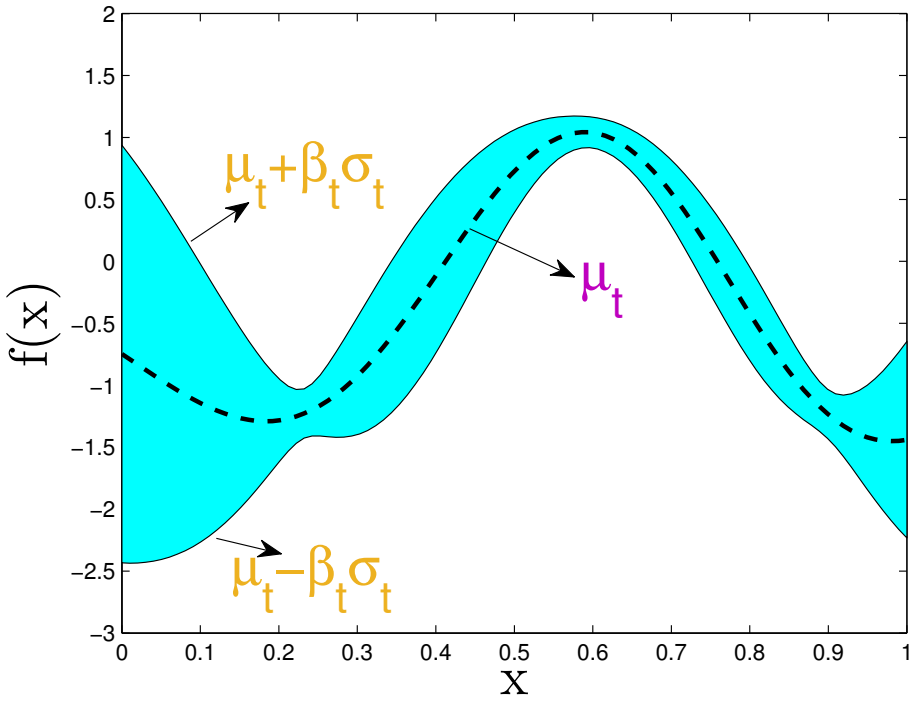
$$\mu_t(x) - \beta_t \sigma_t(x) \leq f(x) \leq \mu_t(x) + \beta_t \sigma_t(x) \qquad \text{whp}$$

Key Technique: Posterior Concentration

Lemma: Concentration of Posterior Distribution

For all t and for all $x \in D$:

$$\mu_t(x) - \beta_t \sigma_t(x) \leq f(x) \leq \mu_t(x) + \beta_t \sigma_t(x) \qquad \text{whp}$$

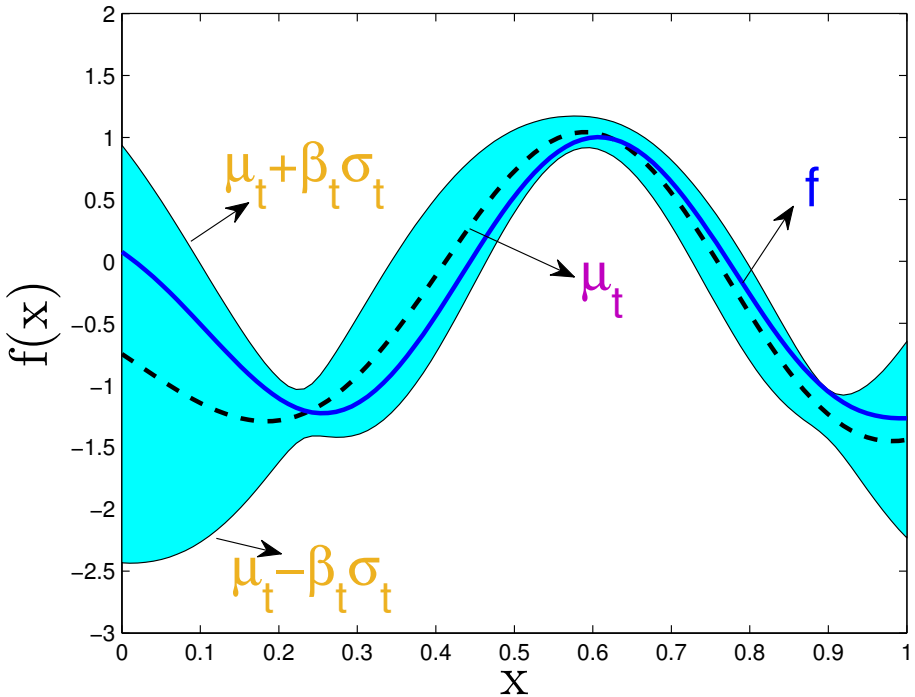


Key Technique: Posterior Concentration

Lemma: Concentration of Posterior Distribution

For all t and for all $x \in D$:

$$\mu_t(x) - \beta_t \sigma_t(x) \leq f(x) \leq \mu_t(x) + \beta_t \sigma_t(x) \qquad \text{whp}$$



At every round, the unknown original function lies within properly constructed confidence interval of shrinking width

Key Technique: New Concentration Inequality (CI)

Result 3: Self-Normalized CI for RKHS-valued Martingales

- For all t: $\|S_t\|^2_{V_t^{-1}} \leq 2R^2 \ln(\frac{\sqrt{\det(K_t+I)}}{\delta})$ with probability at least $1-\delta$ if K_t is positive-definite
- **Generalizes** finite-dimensional Inequality for vector-valued Martingales (Abbasi-Yadkori et al., NIPS 2011) to infinite dimensions

Key Technique: New Concentration Inequality (CI)

Result 3: Self-Normalized CI for RKHS-valued Martingales

- For all t: $\|S_t\|^2_{V_t^{-1}} \leq 2R^2 \ln(\frac{\sqrt{\det(K_t+I)}}{\delta})$ with probability at least $1-\delta$ if K_t is positive-definite
- **Generalizes** finite-dimensional Inequality for vector-valued Martingales (Abbasi-Yadkori et al., NIPS 2011) to infinite dimensions

Example: Let $\varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots$ be a sequence of independent random variables and $x_1, x_2, x_3, \ldots$ be a predictable sequence of random variables. Define $S_t = \sum_{s=1}^{t} \varepsilon_s x_s$ and $V_t = 1 + \sum_{s=1}^{t} x_s^2$. Then $S_t/\sqrt{V_t}$ is a Self-normalized process and its fluctuations are $O(\ln t)$

Summary

For Non-parametric Bandits, we

- **Improved** existing UCB based algorithm
- **Introduced** new Thompson Sampling based algorithm
- **Developed** new self-normalized concentration inequality for RKHS-valued martingales

Summary

For Non-parametric Bandits, we

- **Improved** existing UCB based algorithm
- **Introduced** new Thompson Sampling based algorithm
- **Developed** new self-normalized concentration inequality for RKHS-valued martingales

Possible Extensions:

- Kernel function not known to the learner
- Time varying functions from RKHS

Part 2: Online Learning in Kernelized Markov Decision Processes[2]

[2]S. R. Chowdhury and A. Gopalan, *Online Learning in Kernelized Markov Decision Processes*, ArXiv e-prints, May 2018. (Under review in **NIPS**)

Problem Statement

Episodically maximize reward in (unknown) MDP $M = \{\mathcal{S}, \mathcal{A}, R, P, H, \rho\}$

- State space $\mathcal{S} \subseteq \mathbb{R}^m$, known
- Action space $\mathcal{A} \subseteq \mathbb{R}^n$, known
- Reward distribution $R(s, a)$, unknown
- Transition distribution $P(s, a)$, unknown
- Episode length H, known
- State distribution ρ, known

Problem Statement

Episodically maximize reward in (unknown) MDP $M = \{\mathcal{S}, \mathcal{A}, R, P, H, \rho\}$

- State space $\mathcal{S} \subseteq \mathbb{R}^m$, known
- Action space $\mathcal{A} \subseteq \mathbb{R}^n$, known
- Reward distribution $R(s, a)$, unknown
- Transition distribution $P(s, a)$, unknown
- Episode length H, known
- State distribution ρ, known

At each period h within an episode:

- Learner takes action $a_h \in \mathcal{A}$ based on current state s_h and past observations
- Receives reward $r_h \sim R(s_h, a_h)$
- Observes next state $s_{h+1} \sim P(s_h, a_h)$

Definitions

- Policy $\pi : \mathcal{S} \times \{1, \ldots, H\} \to \mathcal{A}$

Definitions

- Policy $\pi : \mathcal{S} \times \{1, \ldots, H\} \to \mathcal{A}$
- Value function $V_{\pi,h}(s) = \mathbb{E}\left[\sum_{j=h}^{H} \overline{R}(s_j, a_j) \mid s_h = s \right]$
  - *Finite horizon*, **Undiscounted** value function
  - Mean reward $\overline{R}(s, a)$

Definitions

- Policy $\pi : \mathcal{S} \times \{1, \ldots, H\} \to \mathcal{A}$
- Value function $V_{\pi,h}(s) = \mathbb{E}\Big[\sum_{j=h}^{H} \overline{R}(s_j, a_j) \;\Big|\; s_h = s \Big]$
  - *Finite horizon*, **Undiscounted** value function
  - Mean reward $\overline{R}(s, a)$
- Optimal policy $\pi_\star \in \operatorname{argmax}_\pi V_{\pi,h}(s) \;\; \forall s, \; \forall h$

Definitions

- Policy $\pi : \mathcal{S} \times \{1, \ldots, H\} \to \mathcal{A}$
- Value function $V_{\pi,h}(s) = \mathbb{E}\Big[\sum_{j=h}^{H} \overline{R}(s_j, a_j) \;\Big|\; s_h = s\Big]$
  - *Finite horizon*, **Undiscounted** value function
  - Mean reward $\overline{R}(s, a)$
- Optimal policy $\pi_\star \in \operatorname{argmax}_\pi V_{\pi,h}(s) \;\; \forall s, \; \forall h$
- Cumulative Regret $= \sum_I \mathbb{E}\Big[V_{\pi_\star,1}(s) - V_{\pi_I,1}(s)\Big]$

Goal: Minimize the loss incurred in Value function due to not knowing the optimal policy $\pi_\star$ of the unknown MDP M and instead using any other policy π_I at episode I

Assumptions

- View $r \sim R(s,a) \Longrightarrow r = \overline{R}(s,a) + \varepsilon_R$
- View $s' \sim P(s,a) \Longrightarrow s' = \overline{P}(s,a) + \varepsilon_P$

Assumptions

- View $r \sim R(s,a) \implies r = \overline{R}(s,a) + \varepsilon_R$
- View $s' \sim P(s,a) \implies s' = \overline{P}(s,a) + \varepsilon_P$
  - ε_R and ε_P samples of zero-mean, additive sub-Gaussian noise

Assumptions

- View $r \sim R(s,a) \implies r = \overline{R}(s,a) + \varepsilon_R$
- View $s' \sim P(s,a) \implies s' = \overline{P}(s,a) + \varepsilon_P$
  - ε_R and ε_P samples of zero-mean, additive sub-Gaussian noise
- Unknown Mean reward function $\overline{R}$ and mean transition function $\overline{P}$ lies in **RKHS** of functions $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$

Assumptions

- View $r \sim R(s,a) \Longrightarrow r = \overline{R}(s,a) + \varepsilon_R$
- View $s' \sim P(s,a) \Longrightarrow s' = \overline{P}(s,a) + \varepsilon_P$
  - ε_R and ε_P samples of zero-mean, additive sub-Gaussian noise
- Unknown Mean reward function $\overline{R}$ and mean transition function $\overline{P}$ lies in **RKHS** of functions $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$
  - Positive semi-definite kernel functions in the product space
  - **Product kernels**:
    $(k_{\mathcal{S}} \otimes k_{\mathcal{A}})\Big((s,a),(s',a')\Big) = k_{\mathcal{S}}(s,s') \times k_{\mathcal{A}}(a,a')$

Assumptions

- View $r \sim R(s,a) \implies r = \overline{R}(s,a) + \varepsilon_R$
- View $s' \sim P(s,a) \implies s' = \overline{P}(s,a) + \varepsilon_P$
  - ε_R and ε_P samples of zero-mean, additive sub-Gaussian noise
- Unknown Mean reward function $\overline{R}$ and mean transition function $\overline{P}$ lies in **RKHS** of functions $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$
  - Positive semi-definite kernel functions in the product space
  - **Product kernels**:
    $(k_{\mathcal{S}} \otimes k_{\mathcal{A}})\Big((s,a),(s',a')\Big) = k_{\mathcal{S}}(s,s') \times k_{\mathcal{A}}(a,a')$
- Additional Regularity condition on Value function $\to$ Transition distributions with same means are identical

Algorithm 1: Gaussian Process-UCRL (GP-UCRL)

Same philosophy as earlier: Put *separate* Gaussian process priors over mean reward and mean transition functions and update posteriors at the end of every episode

Algorithm 1: Gaussian Process-UCRL (GP-UCRL)

Same philosophy as earlier: Put *separate* Gaussian process priors over mean reward and mean transition functions and update posteriors at the end of every episode

At each episode l:

1. Construct confidence sets, one each for mean reward and mean transition functions, using parameters of posterior distributions respectively

Algorithm 1: Gaussian Process-UCRL (GP-UCRL)

Same philosophy as earlier: Put *separate* Gaussian process priors over mean reward and mean transition functions and update posteriors at the end of every episode

At each episode l:

1. Construct confidence sets, one each for mean reward and mean transition functions, using parameters of posterior distributions respectively
2. Build the set $\mathcal{M}_l$ of all plausible MDPs such that mean reward and mean transition functions lie within respective confidence sets

Algorithm 1: Gaussian Process-UCRL (GP-UCRL)

Same philosophy as earlier: Put *separate* Gaussian process priors over mean reward and mean transition functions and update posteriors at the end of every episode

At each episode l:

1. Construct confidence sets, one each for mean reward and mean transition functions, using parameters of posterior distributions respectively
2. Build the set $\mathcal{M}_l$ of all plausible MDPs such that mean reward and mean transition functions lie within respective confidence sets
3. Choose the optimistic policy π_l for the **set of MDPs** $\mathcal{M}_l$ and execute it for the entire episode

Algorithm 2: Thompson Sampling for Reinforcement Learning (TSRL)

General Philosophy: Start with **any** prior distribution over MDPs and at the start of every episode sample an MDP from the posterior

Algorithm 2: Thompson Sampling for Reinforcement Learning (TSRL)

General Philosophy: Start with **any** prior distribution over MDPs and at the start of every episode sample an MDP from the posterior

At each episode l:

1. Build an MDP M_l such that mean reward and mean transition functions are samples from respective posterior distributions
2. Choose the optimal policy π_l for the **sampled MDP** M_l and execute it for the entire episode

Algorithm 2: Thompson Sampling for Reinforcement Learning (TSRL)

General Philosophy: Start with **any** prior distribution over MDPs and at the start of every episode sample an MDP from the posterior

At each episode I:

1. Build an MDP M_I such that mean reward and mean transition functions are samples from respective posterior distributions
2. Choose the optimal policy π_I for the **sampled MDP** M_I and execute it for the entire episode

GP-TSRL: For Gaussian Process prior and Gaussian likelihood model, the posteriors admit nice closed form of Gaussian processes

Regret Bounds

Result 1

Cumulative Regret of **GP-UCRL** is $\tilde{O}\left((\gamma_{R,T} + \gamma_{P,T})\sqrt{T}\right)$ *whp*

Regret Bounds

Result 1

Cumulative Regret of **GP-UCRL** is $\tilde{O}\left((\gamma_{R,T} + \gamma_{P,T})\sqrt{T}\right)$ *whp*

Result 2

Expected (over prior distribution of MDPs) cumulative Regret of **GP-TSRL** is $\tilde{O}\left((\gamma_{R,T} + \gamma_{P,T})\sqrt{T}\right)$

Regret Bounds

Result 1

Cumulative Regret of **GP-UCRL** is $\tilde{O}\left((\gamma_{R,T} + \gamma_{P,T})\sqrt{T}\right)$ *whp*

Result 2

Expected (over prior distribution of MDPs) cumulative Regret of **GP-TSRL** is $\tilde{O}\left((\gamma_{R,T} + \gamma_{P,T})\sqrt{T}\right)$

- $\gamma_{R,T}$ and $\gamma_{P,T}$ are **Maximum Information Gain** about (unknown) mean reward and mean transition functions after T rounds
- Grow sub-linearly with T for common kernels (e.g. Squared Exponential) and for their compositions (products, sums)

Summary

Proved First regret guarantees of UCRL and TSRL in kernel based MDPs, where

- Mean reward and transition functions are elements from Reproducing Kernel Hilbert Spaces
- Mean reward and transition functions are samples from Gaussian Processes (Bayesian setup, not presented here)

Summary

Proved First regret guarantees of UCRL and TSRL in kernel based MDPs, where

- Mean reward and transition functions are elements from Reproducing Kernel Hilbert Spaces
- Mean reward and transition functions are samples from Gaussian Processes (Bayesian setup, not presented here)

Future work:

- Frequentist regret bound of GP-PSRL
- Online learning in Model free MDPs (obviate complicated planning step)

Thank You