Tutorial on Bayesian Optimization

Probabilistic Artificial Intelligence, Fall 2022

Parnian Kassraie

ETH zürich

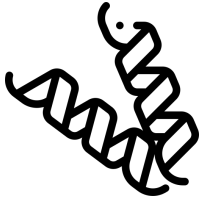
Learning &
Adaptive Systems

These slides are not mathematically rigorous.
Just to be safe, take everything with a grain of salt.

Bayesian Optimization everywhere

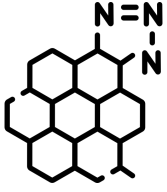
we want to optimize an unknown f

$$x^\star \in \arg\max_{x \in \mathcal{X}} f^\star(x)$$

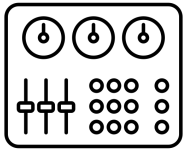


Protein Design

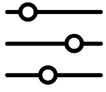
Drug Discovery



Molecule Synthesis



Online Control



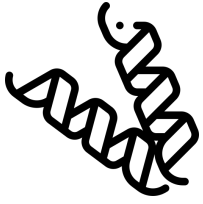
Hyperparameter Tuning

Recommender Systems

image source: flaticon

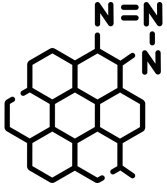
Bayesian Optimization everywhere

$$x^\star \in \arg\max_{x \in \mathcal{X}} f^\star(x)$$

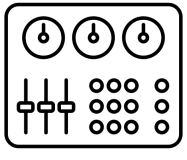


Protein Design

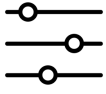
Drug Discovery



Molecule Synthesis



Online Control



Hyperparameter Tuning

Recommender Systems

Choose x_t

Observe $y_t = f^\star(x_t) + \epsilon_t$

Repeat

Costly

Need to be sample efficient

We can model all as a ✨ bandit ✨ problem

image source: flaticon

Problem Setting

Problem

f^* unknown, $\mathcal{X}$ find $x^* = \arg\max_{x \in \mathcal{X}} f^*(x)$

at t, choose x_t, observe $y_t = f^*(x_t) + \epsilon_t$

↑ Based on $(x_{1:t-1}, y_{1:t-1})$

Eventually ↘

Goal

Find an optima f^* + Sample efficient + generally pick good actions

cumulative regret: $R_T = \sum f^*(x^*) - f^*(x_t)$ $\quad R_T/T \to 0$ sublinearity

max reward $\Leftrightarrow$ min regret

Assumptions

Noise: ϵ_t : iid, zero mean, sub-G dist. σ^2

Domain: compact, within $\mathbb{R}^d$ (d-dim. Euclidean set)

Reward: $f^* \sim GP(0, k)$, $f^* \in \mathcal{H}_k$ ~ frequentist approach

known kernel

Other BO(-related) setting

There are many.

gap dependent analysis

Simple Regret

Best of both worlds

Instance-dependent Regret

Representation Learning

Sparse bandits

Adversarial noise

delayed feedback

Meta-Learning and model selection

batched algorithms

Experiment design

Online learning

$\mathcal{X}$-armed bandits

Best arm identification

Optimistic Policies

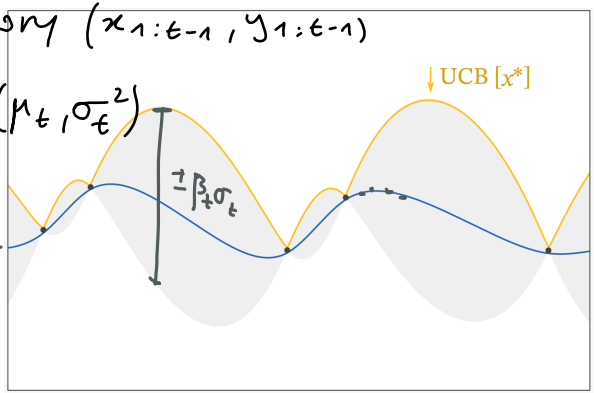
policy: what action to pick

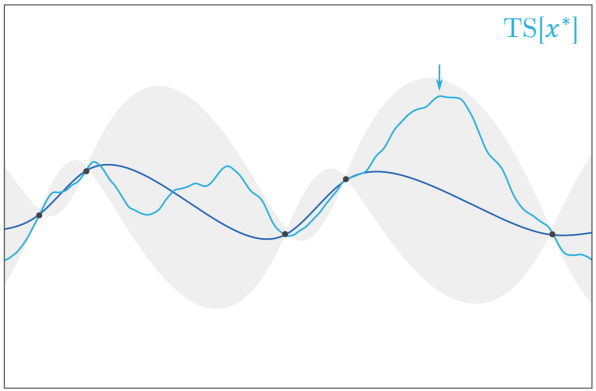
$$x_{t+1} = \arg\max_{x \in \mathcal{X}} \mathrm{AF}\left(x, \mathrm{history}_{1:t}\right)$$

Assume $f^* \sim GP(0, K)$

↳ history $(x_{1:t-1}, y_{1:t-1})$

↳ $GP(\mu_t, \sigma_t^2)$





$$\mathrm{UCB}(x) = \mu_t(x) + \beta_t \sigma_t(x)$$

UpperConfidenceBound

$$\mathrm{TS}(x) = f_t(x),\ f_t \sim \mathrm{GP}(\mu_t, \sigma_t^2)$$

ThompsonSampling

draws sample from GP

What's the role of β_t? should be non-decreasing

How would you change it with t?

$\beta_t \uparrow$ incentivises exploration (weighs variance)

In terms of guarantees, these two are pretty equivalent. UCB is more practical.

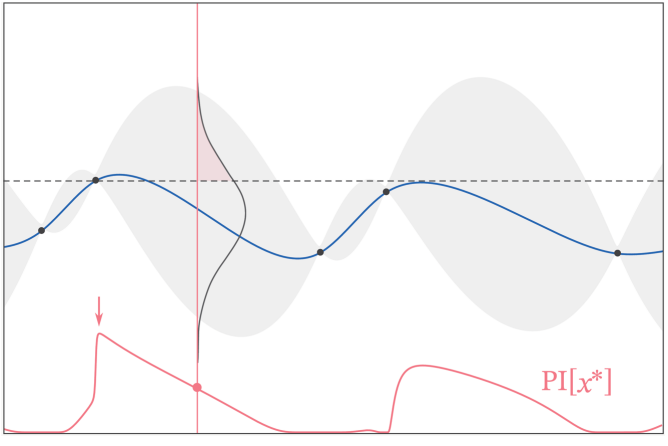
as posterior var σ_t naturally ↓ (to counteract: $\beta_t \uparrow$)

Optimistic Policies

$$x_{t+1} = \arg\max_{x \in \mathcal{X}} \mathrm{AF}\left(x, \mathrm{history}_{1:t}\right)$$

largest reward seen before

$y_t^* = \text{argmax}_{i \le t} \; y_i$



$$\mathrm{PI}(x) = \mathbb{P}\left(f^\star(x) > y_t^\star \middle| \mathrm{history}_{1:t}\right) = \Phi\left(\frac{\mu_t(x) - y_t^\star}{\sigma_t(x)}\right)$$

Probability of Improvement

incentivises picking points close to previous ones (b/c there σ_t is low $\Rightarrow$ PI ↑)

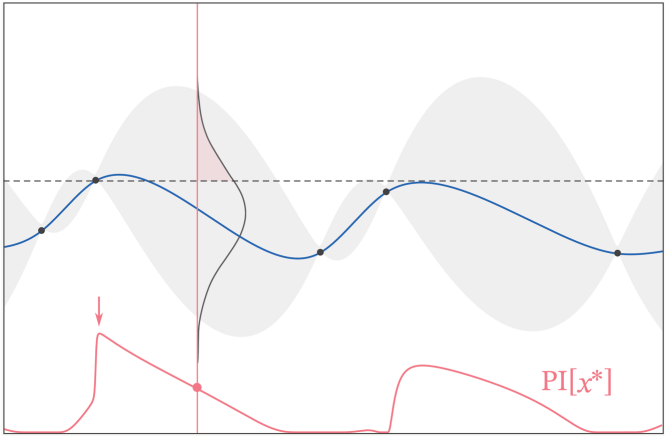
What kind of actions are incentivised here?

$\mathbb{P}(f^* \mid \text{history}) \propto GP(\mu_t, \sigma_t)$

image source: borealis AI

Optimistic Policies

$$x_{t+1} = \arg\max_{x \in \mathcal{X}} \mathrm{AF}\left(x, \mathrm{history}_{1:t}\right)$$



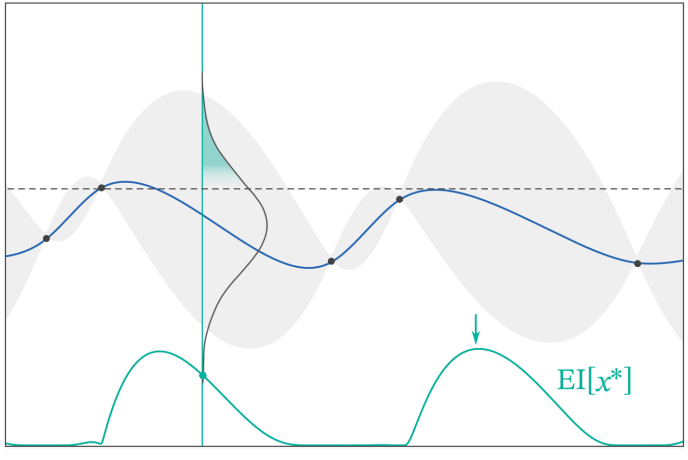
$$\mathrm{PI}(x) = \mathbb{P}\left(f^\star(x) > y_t^\star \middle| \mathrm{history}_{1:t}\right) = \Phi\left(\frac{\mu_t(x) - y_t^\star}{\sigma_t(x)}\right)$$

What kind of actions are incentivised here?
Points that are close to the previous maximas (therefore, small posterior variance) which have a posterior mean larger than the best previously observed point. May not explore sufficiently...

image source: borealis AI

Optimistic Policies

$$x_{t+1} = \arg\max_{x \in \mathcal{X}} \mathrm{AF}\left(x, \mathrm{history}_{1:t}\right)$$

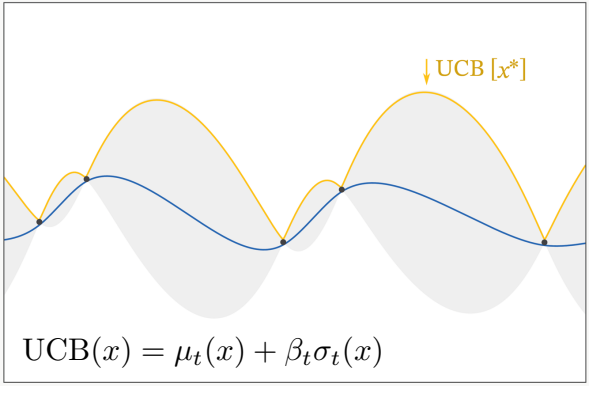


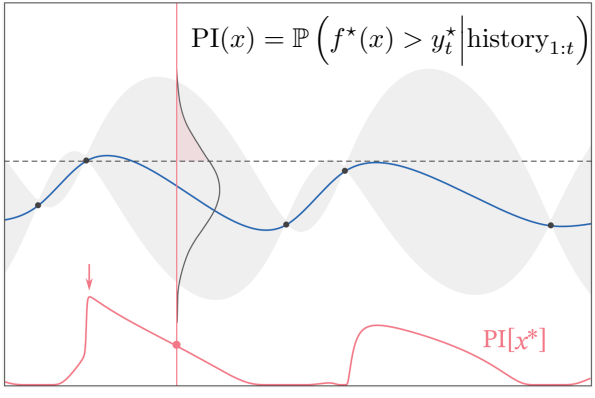
$$\mathrm{EI}(x) = \mathbb{E}\left[(f^\star(x) - y_t^\star)_+ \middle| \mathrm{history}_{1:t}\right]$$

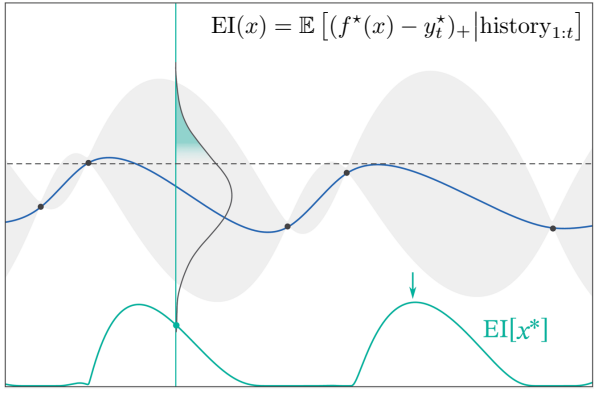
Note that the unknown reward, conditioned on the history, is a sample from the posterior GP distribution. So the expectation has a closed form expression in terms of posterior mean and variance. EI doesn't just look at probability of improvement, it also takes into acount the magniture of improvement.

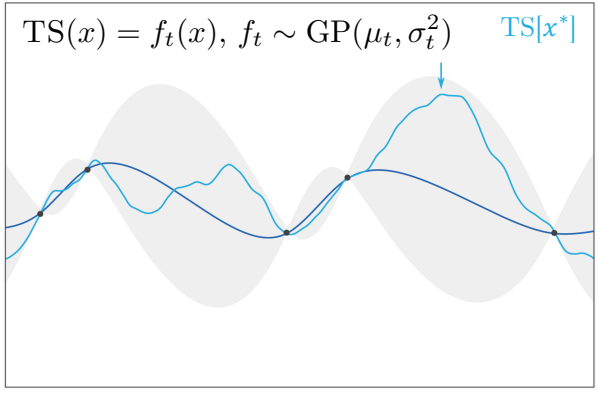
Optimistic Policies

If one works, then all do. However, one might be more sample efficient than the others. EI is a good practical choice. UCB is a good theoretical choice.







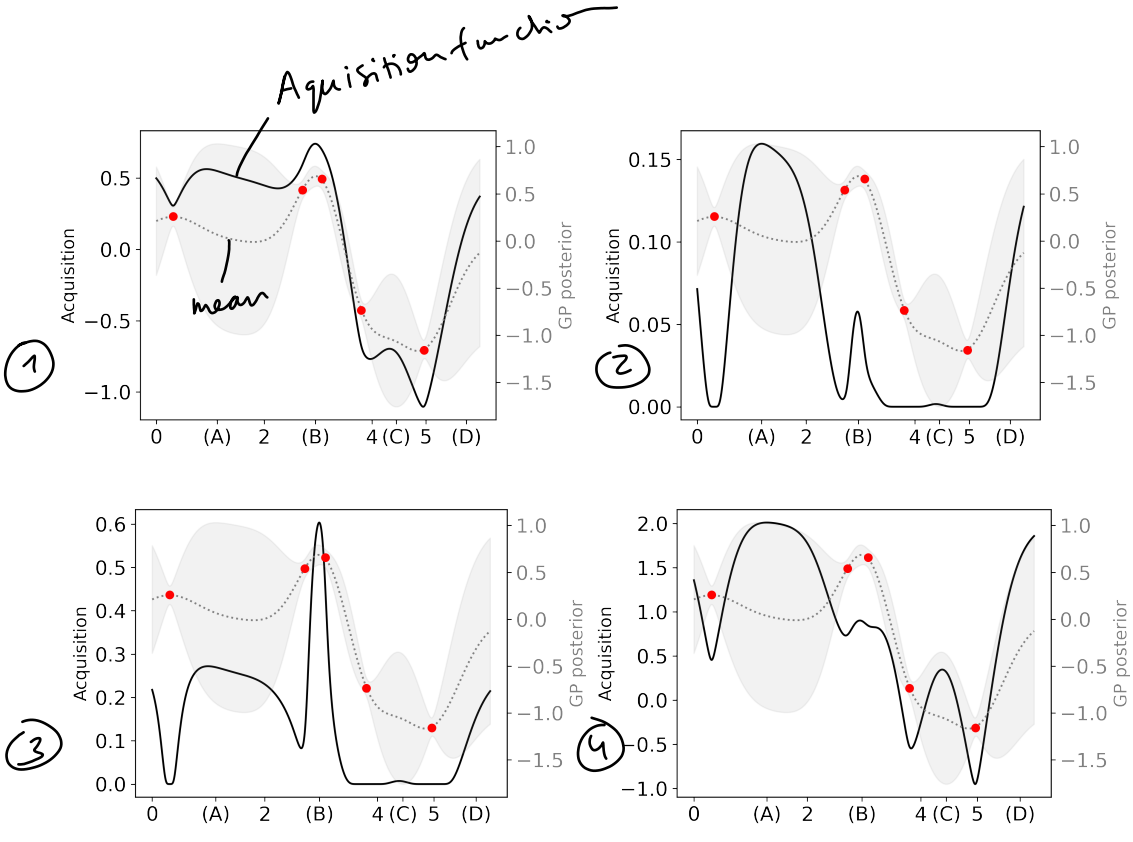


$$x_{t+1} = \arg\max_{x \in \mathcal{X}} \mathrm{AF}\left(x, \mathrm{history}_{1:t}\right)$$

Check "A Contextual Bandit Bake-off" Bietti et al 2021 for a benchmark.

image source: borealis AI

From HW4



EI (2) PI (3) UCB, $\beta = 2$ (4) UCB, $\beta = 0.5$ (1)

large exploration ⇐⇒ small exploration

industry standard

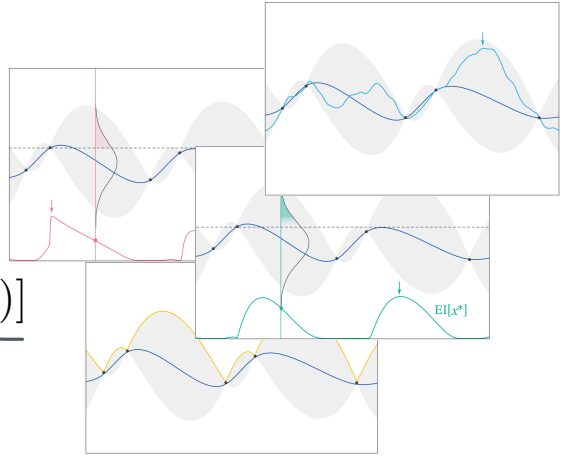
Exploration and Exploitation

$f^\star$ is unknown.

To find an optimal policy for selecting actions, the agent should

explore

exploit



The sets $\mathcal{C}_t(x) = [\mu_t(x) - \beta_t \sigma_t(x), \mu_t(x) + \beta_t \sigma_t(x)]$

Are a tool to guide explorations

monotonically decreasing with number of samples

width $\longleftrightarrow$ current uncertainty

center $\longleftrightarrow$ current knowledge

At every time t, we use the set $\mathcal{C}_t(x)$ as a proxy for the unknown reward $f^\star(x)$

Confidence Sets at the center of Exploration-Exploitation

At every time t, we use the set $\mathcal{C}_t(x)$ as a proxy for the unknown reward $f^\star(x)$

But, why are we allowed to do so? Why does this help?

Theorem

If $f^\star \sim \mathrm{GP}(0, k)$, then for a certain choice of β_t, the confidence sets are any-time valid with high probability, i.e.

$$\mathbb{P}\Big(f^*(x) \in \mathcal{C}_{t-1}(x),\ \forall x \in \mathcal{X}, t \geq 1\Big) \geq 1 - \delta$$

value of reward is in this interval w/ high probability

What does this imply? $\mathcal{C}_t(x) = [\mu_t(x) - \beta_t \sigma_t(x), \mu_t(x) + \beta_t \sigma_t(x)]$

That by using the confidence set, instead of the unknown reward, the single-step regret will be controlled by the width of the set, whp.

Confidence sets and Regret Bounds

With high probability, confidence bound gives

$$|f(\boldsymbol{x}_t) - \mu_{t-1}(\boldsymbol{x}_t)| \leq \beta_t \sigma_{t-1}(\boldsymbol{x}_t)$$
$$|f(\boldsymbol{x}^*) - \mu_{t-1}(\boldsymbol{x}^*)| \leq \beta_t \sigma_{t-1}(\boldsymbol{x}^*)$$

$f^* \in C_{t-1}(x)$ w.h.p.

$|f^*(x) - \mu_t(x)| \leq \beta_t \sigma_t(x) \quad \forall x \in \mathcal{X}$

Use the UCB policy

regret: $r_t = f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t)$ (optimal reward)

$$\begin{aligned} &\leq \mu_{t-1}(\boldsymbol{x}^*) + \beta_t \sigma_{t-1}(\boldsymbol{x}_*) - f(\boldsymbol{x}_t) \\ &\leq \mu_{t-1}(\boldsymbol{x}_t) + \beta_t \sigma_{t-1}(\boldsymbol{x}_t) - f(\boldsymbol{x}_t) \\ &\leq 2\beta_t \sigma_{t-1}(\boldsymbol{x}_t) \end{aligned}$$

Invoke the confidence bound T times, with a special choice of β_t

$$\begin{aligned} R_T = \sqrt{\sum_{t=1}^{T} r_t} &\leq \sqrt{T} \sqrt{\sum_{t=1}^{T} r_t^2} \\ &\leq \sqrt{T} \sqrt{\sum_{t=1}^{T} 4\beta_t^2 \sigma_{t-1}^2(\boldsymbol{x}_t)} \\ &= \tilde{\mathcal{O}}\left(\sqrt{\gamma_T T}\right) \end{aligned}$$

Do some algebra to upper bound the variance term with information gain

Can this be achieved with other policies?

Other Policies

$$f^\star(x) = x^\top w^\star \qquad x \in \mathbb{R}^d$$

Which algorithms can be sublinear?

Algorithm 1 GREEDY

Pick first action uniformly at random and observe y_1.
for $t \in \{2, \dots, T\}$ **do**
  Estimate $\hat{w}_{t-1}$ using $(x_i, y_i)_{i<t}$.
  Choose $x_t = \arg\max_{x \in \mathcal{X}} x^\top \hat{w}_{t-1}$.
  Observe y_t.
end for

Algorithm 2 UCB

Require: β_t
for $t \in \{1, \dots, T\}$ **do**
  Estimate μ_{t-1} and σ_{t-1} using $(x_i, y_i)_{i<t}$.
  Choose $x_t = \arg\max_{x \in \mathcal{X}} \mu_{t-1}(x) + \beta_t \sigma_{t-1}(x)$.
  Observe y_t.
end for

Given a "good" choice of T_0 & β_t, all of them can be! But this is only true for the linear reward.

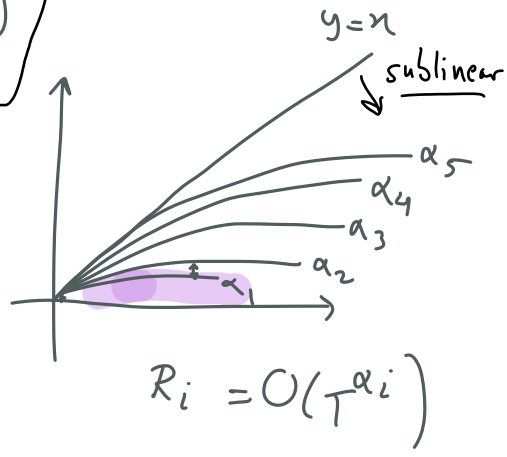
Algorithm 3 EXPLORE-THEN-COMMIT

Require: T_0
for $t \in \{1, \dots, T_0\}$ **do**
  Choose x_t uniformly at random.
  Observe y_t.
end for
Estimate $\hat{w}$ using $(x_t, y_t)_{t \le T_0}$.
for $t \in \{T_0, \dots, T\}$ **do**
  Choose $x_t = \arg\max_{x \in \mathcal{X}} x^\top \hat{w}$.
  Observe y_t.
end for

From HW4

$R_{(T)} = \mathcal{O}(T^{\alpha})$

sublinear: $\alpha < 1$

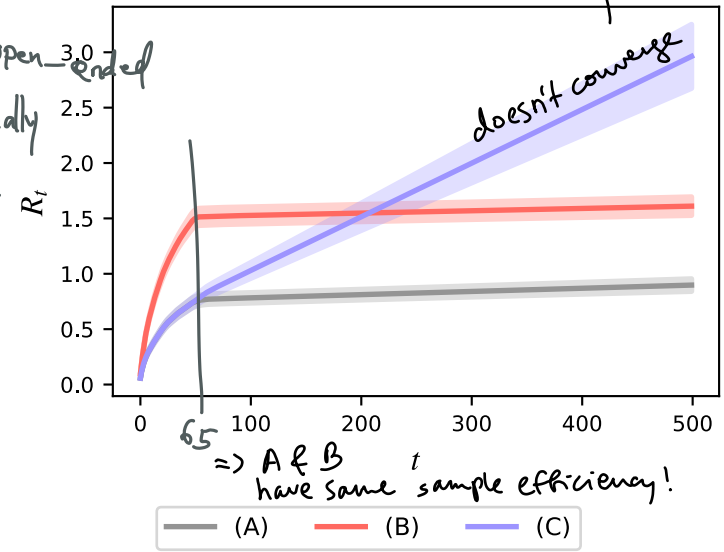


Which of the following scenarios are possible?

- ✗ Algorithm (A) is more sample efficient (finds sol. faster) than (B) and (C).
- (✓) Algorithm (B) on the average explores more than (A).
- (✗) Algorithm (C) on the average explores more than (B).
- ✓ Algorithm (C) has gotten stuck in a local minima. ← linear regret
- ✗ Algorithm (C) may eventually find an optimal action.

This is a high-level open-ended example, and intentionally kept vague, to induce some discussion.

In the exam, the questions are clear and open to no interpretation.



$$R_T^{(A)} = \sum_{t=1}^{T} f(x^\star) - f(x_t^{(A)})$$

smaller α => converge to optimum faster!

Information Gain: a notion of learning complexity

Hardness of sequential learning

Complexity of estimating the unknown function

Complexity of choosing the next action

What is information gain?

GP and i.i.d sub-G noise assumption

Kernel matrix depends on $x_1, \dots, x_T$ (all pts. sampled in time T)

$$I(y_T; f_T) = H(y_T) - H(y_T|f_T) = \frac{1}{2}\log\det(I + \sigma^{-2}K_T) \leq \frac{1}{2\sigma^2}\sum_{i\leq T}\lambda_i(K_T)$$

↳ eigenval's

Note that for i.i.d. gaussian noise, the information gain does not depend on the function evaluations. This hints that there might be better choices for an online measure of complexity...

This is actually an open research question. There are other notions of sequential complexity: Sequential Rademacher Complexity, Eluder Dimension, Star number, Decision Estimation Coefficient

However, Information gain is a pretty good choice. We saw that it naturally appears in (more or less) all worst-case regret bounds!

Information Gain and choice of kernel

If the reward is a complex function (e.g. non-differentiable), we need a rougher kernel for the GP assumption to be valid. The information gain of complex kernels grows more rapidly with T.

$$I(y_T; f_T) \leq \frac{1}{2\sigma^2} \sum_{i \leq T} \lambda_i(K_T)$$

info gain governed by: eigenvalues of the kernel matrix (data-dependent) ➡ Bound it using eigenvalues of the kernel function (data-independent)

The more complex a kernel function is, the slower its eigenvalues decay,

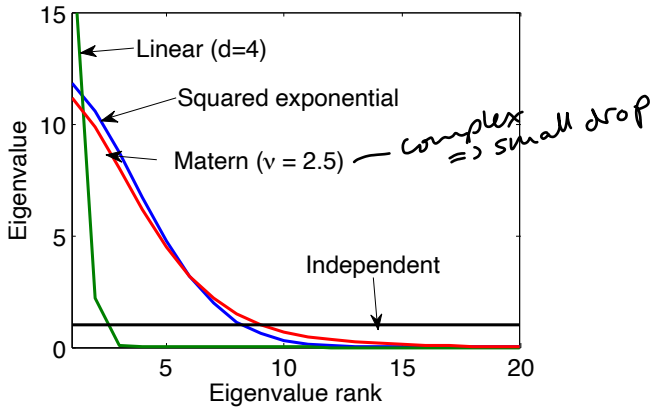
→ b/c more eigenfunctions needed to explain the kernel!

Matérn $\nu > 1/2$ $\quad \lambda_k = \mathcal{O}(k^{-(1+2\nu/d)})$

complex

RBF $\quad \lambda_k = \mathcal{O}(\exp(-k^{1/d}))$

d is input dimension.

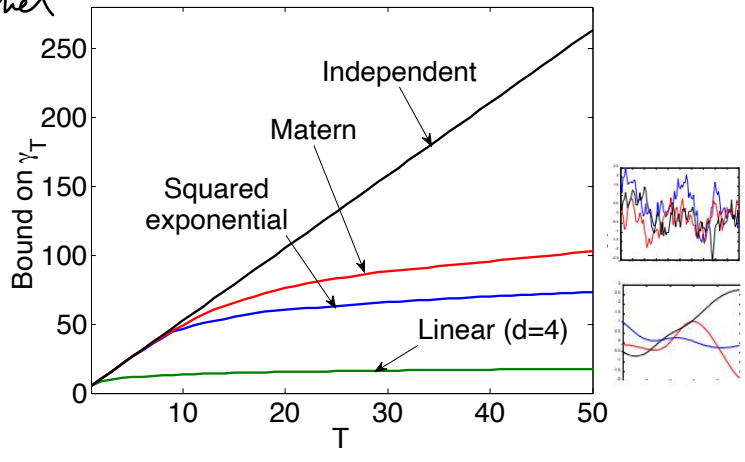


Complex rewards are harder to optimize?

Recall that $R_T = \tilde{\mathcal{O}}(\sqrt{T\gamma_T})$ and say $\gamma_T = \tilde{\mathcal{O}}(T^\alpha)$
For most kernels it has been shown that this rate matches the lower bound up to polylog factors of T.

- reward hard to model ⇒ pick complex kernel ⇒ α worse

If we use a more complex kernel, (unusal) samples from the GP can look crazy. So, while the worst-case regret may still be sublinear, the rate with T naturally gets worst. Since this is a harder problem in the worst-case.



The performance of the algorithm highly depends on the kernel function.

This does not necessarily imply that the algorithm will always do worse.
This is a worst-case guarantee, which holds for any sample from the GP.
The average case will be smoother than the worst-case sample.

Its tough to choose a good kernel

Say we use the UCB policy, and therefore the confince sets

$$\mathcal{C}_t(x) = [\mu_t(x) - \beta_t \sigma_t(x), \mu_t(x) + \beta_t \sigma_t(x)]$$

(increases epistemic uncertainty of model)

If you choose an "overly complex" kernel, the posterior variance will be too large, i.e. your confience sets will be too wide, and we don't want this to happen since:

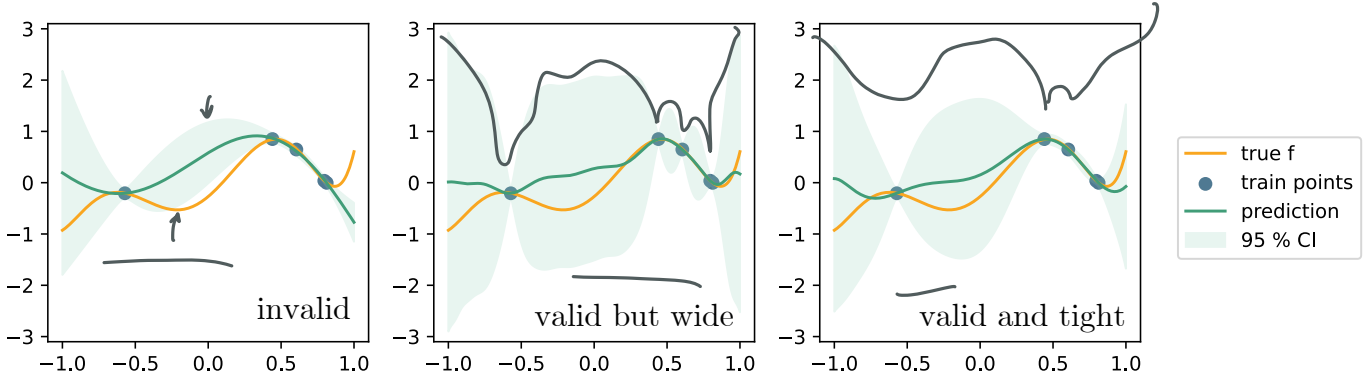
$$r_t = f(\boldsymbol{x}^*) - f(\boldsymbol{x}_t) \leq 2\beta_t \sigma_{t-1}(\boldsymbol{x}_t)$$

(i.e. worse guarantee)

If you choose an overly simple kernel, you don't even have the above inequality, anything can happen!

this doesn't hold

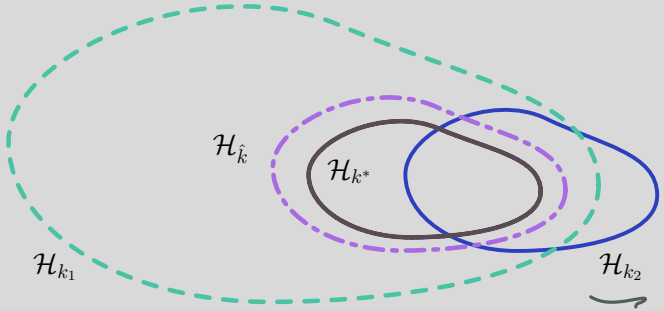
i.e. if you can't say $f^* \propto GP(0, k)$



Ok, but how do I hit this sweet spot?
I don't know! Maybe hand-tune a Matern? 🤡

One line of BO-related research at LAS

We work on Meta-Learning and Model-Selection for Sequential Decision-Making. Throughout this tutorial, we assumed that the "true" GP kernel is known. But this is not the case irl. Working with a wrong or overly complex kernel can be very problematic. Either we can't guarantee convergence, or that the algorithm will be sample inefficient. Ideally, we want to estimate a correct and as simple as possible kernel from the bandit data.



This year, we managed to show that if you can play many similar bandit task, then knowledge of the true kernel is actually not required!
You can meta-learn the kernel on the side, while solving the bandit tasks with oracle-optimal regret guarantee.

Questions?