Gaussian Process Optimization in the Bandit Setting

Pedram Daee

Outline

- Multi-armed Bandit problem
  - Exploration exploitation trade-off
  - Examples
- Contextual Bandits
  - Dependent arms
- Gaussian Process for bandits
  - GP-UCB
  - Thompson Sampling
- Summary

Multi-armed Bandit problem

…

R_1 R_2 R_n

$r_1 \backsim R_2$

$Total\ reward = r_1$

Play #2

Multi-armed Bandit problem

…

R_1 R_2 R_n

$r_2 \backsim R_1$

$$Total\ reward = r_1 + r_2$$

Play #1

Multi-armed Bandit problem

…

R_1 R_2 R_n

After T step: $Total\ reward = r_1 + r_2 + \cdots + r_T$

Goal : maximize the cumulative reward = Minimize expected regret

$$R_T = T\mu^* - \sum_{t=1}^{T} E[r_t]$$

Exploration vs Exploitation

- Search for a balance between exploring the environment to find profitable actions while taking the empirically best action as often as possible.
- Balance between staying with the option that gave highest payoffs in the past and exploring new options that might give higher payoffs in the future.

Multi-Armed Bandits: Examples

- Pure bandit problems arise in many applications
- Applicable whenever:
  - We have a set options with unknown utilities
  - There is a cost for sampling options or a limit on total samples
  - Want to find the best option or maximize utility of our samples
- Examples:
  - Mining for valuable resources (such as gold or oil): exploit good wells, or start digging at a new location.
  - Marketing (e.g. send catalogues to good customers or random people).
- In most practical applications Arms are not independent

Multi-Armed Bandits: Examples (2/2)

- Goal: optimize the beer you drink before you get drunk...

Contextual Bandit

- When there is a large number of arms
  - Idea: Define arms in a feature (context) space where arms that are close to each other have similar expected rewards

At each time step:

1. Algorithm observes:
   - a set of arms A
   - Feature vector x_a for each a ∈ A
2. Algorithm chooses $a_t \in A$, and receives r_{a_t}, where:
$$E[r_a] = f(x_a, \theta)$$
3. Improve strategy based on observed:
$$(x_{a_t}, r_{a_t})$$

T-trial payoff: $\sum_{t=1}^{T} r_{a_t}$

Expected T-trial regret

$$\mathrm{R(T)} = \mathrm{E}\left[\sum_{t=1}^{T} r_{a_t^*}\right] - \mathrm{E}\left[\sum_{t=1}^{T} r_{a_t}\right]$$

Gaussian Process Bandit (1/5)

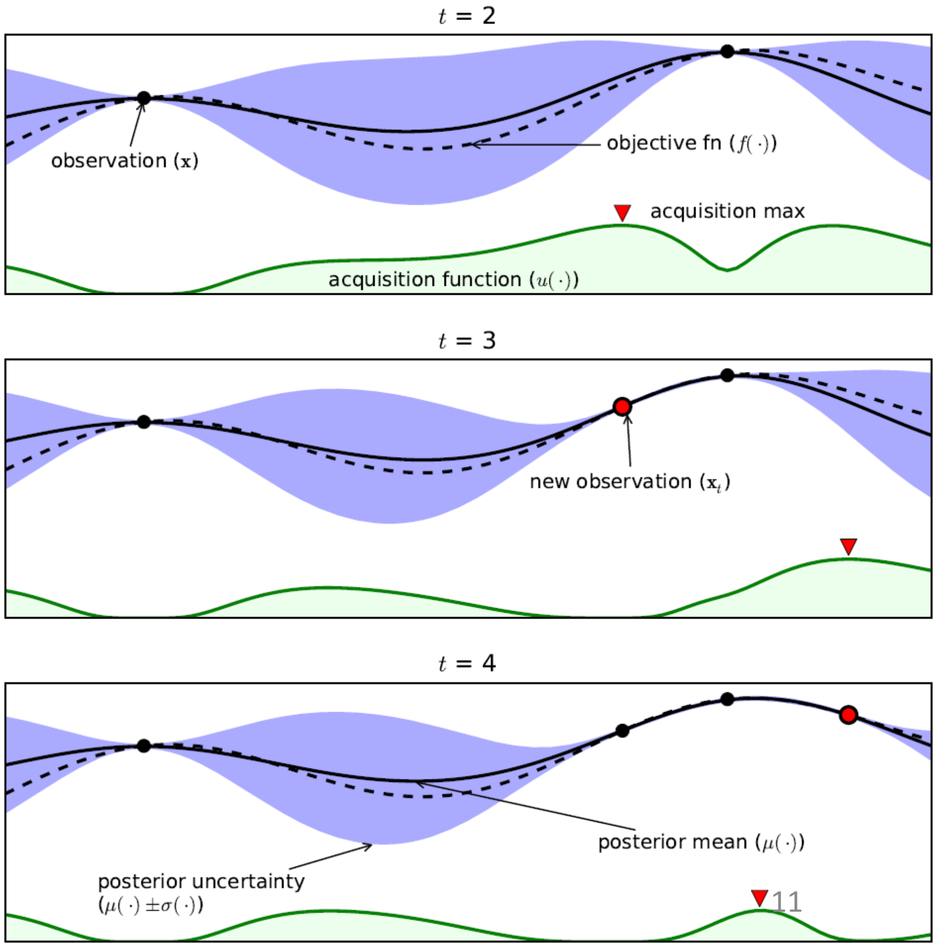
- Optimizing an unknown, noisy function that is expensive to evaluate
  - Minimizing sampling
  - Exploitation vs exploration

Assumptions:

- Playing arm $x \in R^d$, reward value $r = f(x) + \varepsilon$ is observed
  - $\varepsilon \sim N(0, \sigma^2)$, $f \sim GP(0, k(x, x'))$
- Goal: minimize the expected regret: $Tf(x^*) - \sum_{t=1}^{T} f(x_t)$
  - $x^* = \arg\max_{x \in X} f(x)$
  - Perform essentially as well as x^*
  - Regret: the loss in reward due to not knowing fs maximum points beforehand

Gaussian Process Bandit (2/5)

- After playing a set of arms X_{obs} and observing their corresponding reward values R_{obs}
- Posterior:
  - $f|X_{obs}, R_{obs}, X \sim GP(\mu_t(x), k_t(x, x'))$
- $\mu_t(x) = k(X_{obs}, x)^T (k(X_{obs}, X_{obs}) + \sigma^2 I)^{-1} R_{obs}$
- $k_t(x, x') = k(x, x')$
  $- k(X_{obs}, x)^T (k(X_{obs}, X_{obs}) + \sigma^2 I)^{-1} k(X_{obs}, x')$
- Whcih arm to play (which point to sample)?
  - GP-UCB
  - Thompson sampling

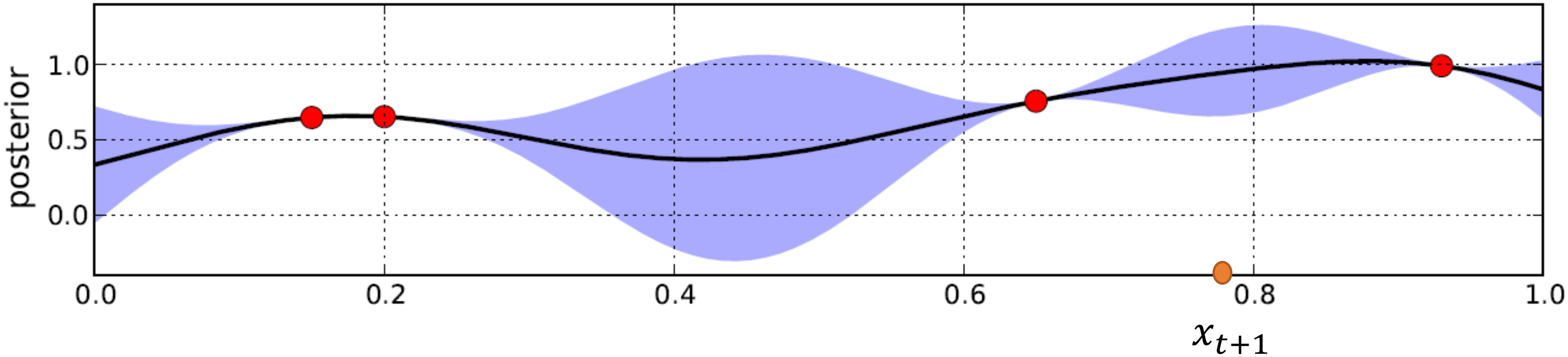


A tutorial on Bayesian optimization …(2010)

Gaussian Process Bandit (3/5) GP-UCB

- $f|X_{obs}, R_{obs}, X \sim GP(\mu_t(x), k_t(x, x'))$
- GP-UCB: select arm with greatest upper bound

$$x_{t+1} = \arg max_{x \in X}\, \mu_t(x) + \beta_t^{\frac{1}{2}}\, k_t(x, x)$$



Brochu, Eric, Vlad M. Cora, and Nando De Freitas. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning." (2010)

Gaussian Process Bandit (4/5) GP-UCB

- GP-UCB regret bounds (up to polylog factors) for linear, radial basis, and Matern kernels
  - d is the dimension, T is the time horizon, and ν is a Matern parameter.

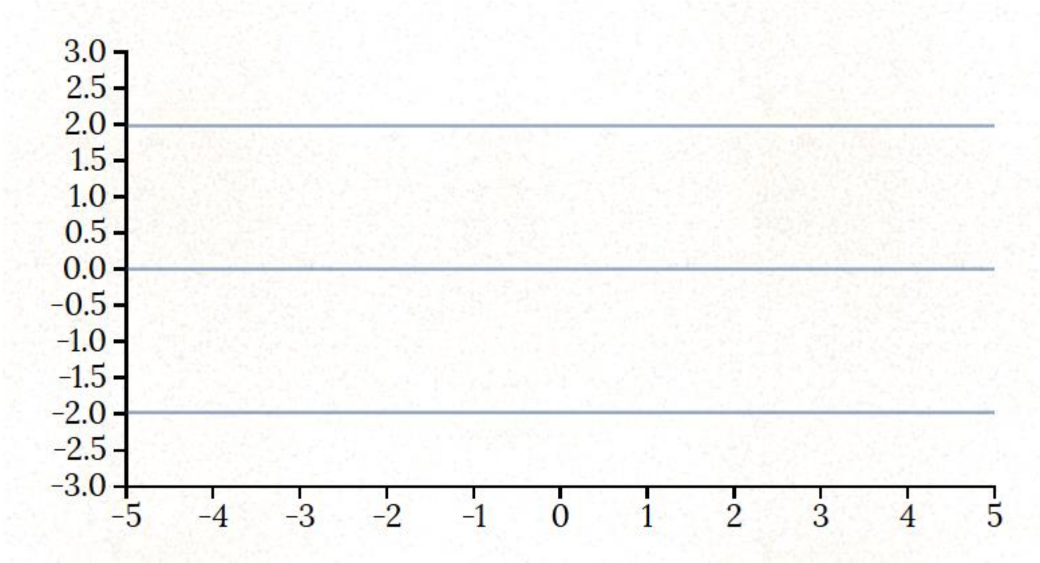
Kernel	Linear	RBF	Matérn
Regret R_T	$d\sqrt{T}$	$\sqrt{T(\log T)^{d+1}}$	$T^{\frac{\nu+d(d+1)}{2\nu+d(d+1)}}$

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

Srinivas, Niranjan, et al. "Gaussian process optimization in the bandit setting: No regret and experimental design." ICML (2010)

Gaussian Process Bandit (5/5)
Thompson Sampling

- Thompson sampling:
  - Draw $f \sim GP(\mu_t(x), k_t(x, x'))$
  - Play $x_{t+1} = \arg max_{x \in X} f(x)$

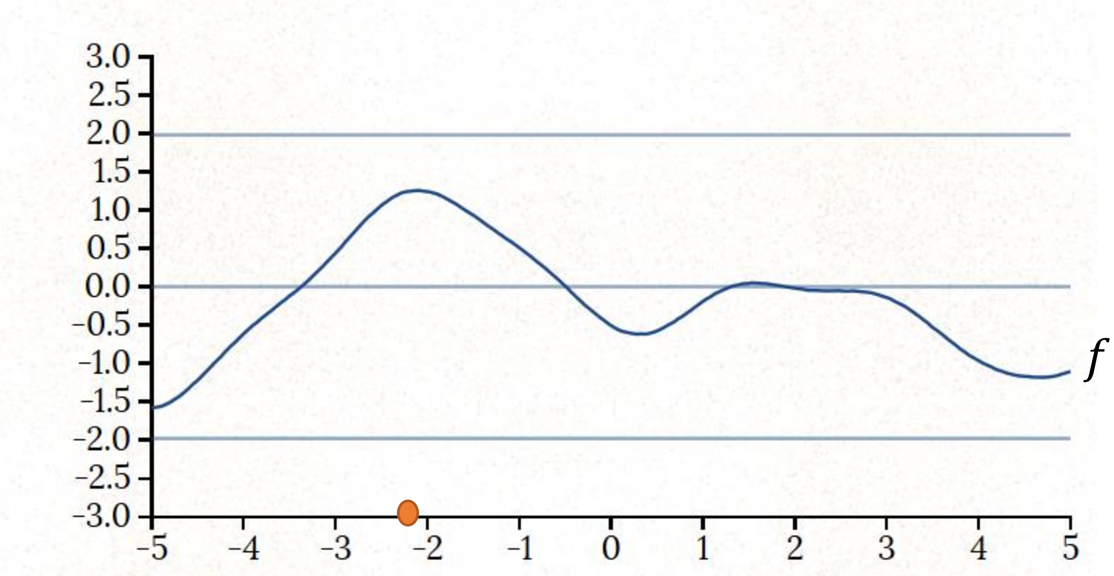


- This will work because our functions will be inside the CIs of the GP

GP Demo: http://www.tmpl.fi/gp/

Gaussian Process Bandit (5/5) Thompson Sampling

- Thompson sampling:
  - Draw $f \sim GP(\mu_t(x), k_t(x, x'))$
  - Play $x_{t+1} = \arg max_{x \in X} f(x)$

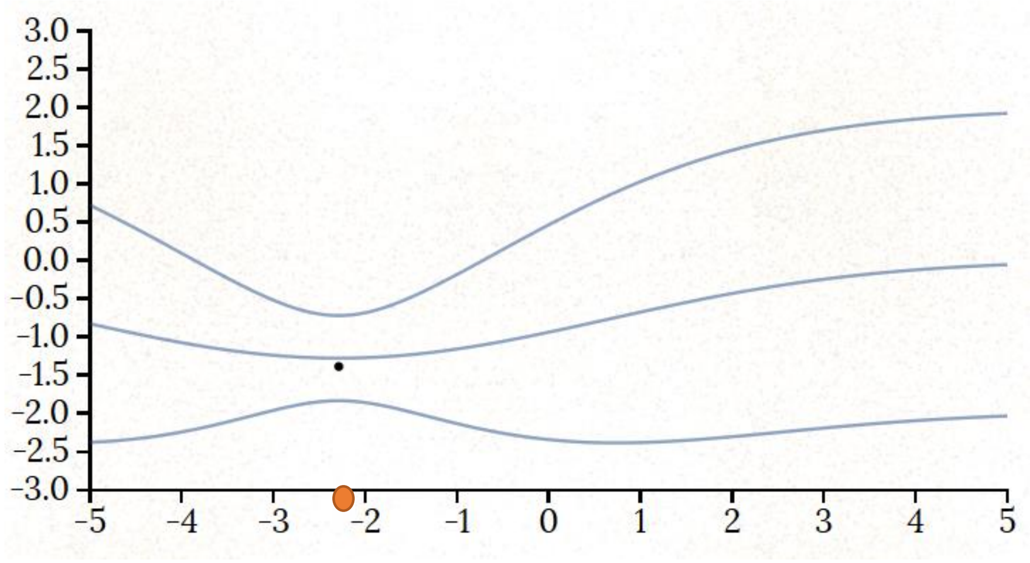


- This will work because our functions will be inside the CIs of the GP

GP Demo: http://www.tmpl.fi/gp/

Gaussian Process Bandit (5/5) Thompson Sampling

- Thompson sampling:
  - Draw $f \sim GP(\mu_t(x), k_t(x, x'))$
  - Play $x_{t+1} = \arg max_{x \in X} f(x)$

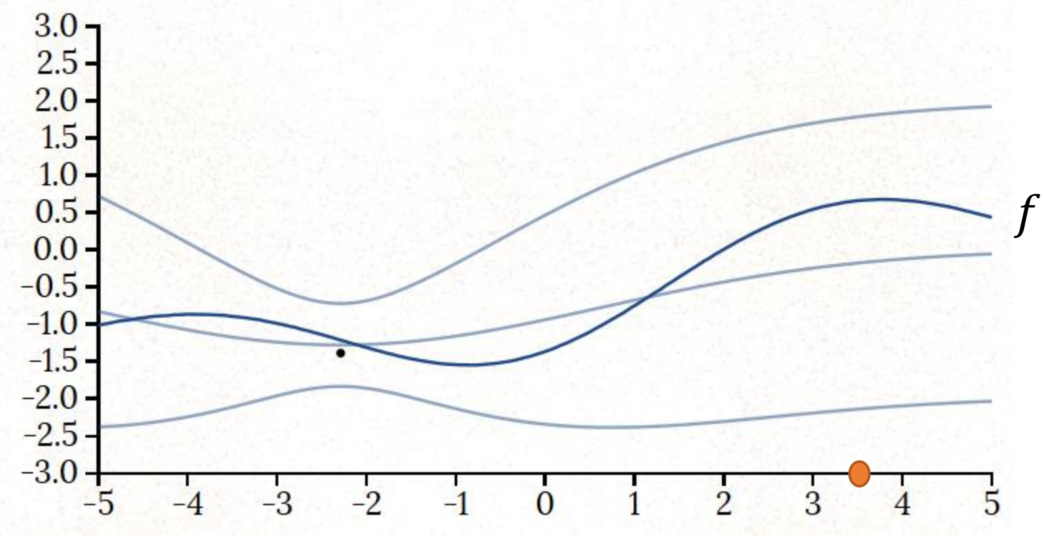


- This will work because our functions will be inside the CIs of the GP

GP Demo: http://www.tmpl.fi/gp/

Gaussian Process Bandit (5/5) Thompson Sampling

- Thompson sampling:
  - Draw $f \sim GP(\mu_t(x), k_t(x, x'))$
  - Play $x_{t+1} = \arg max_{x \in X} f(x)$

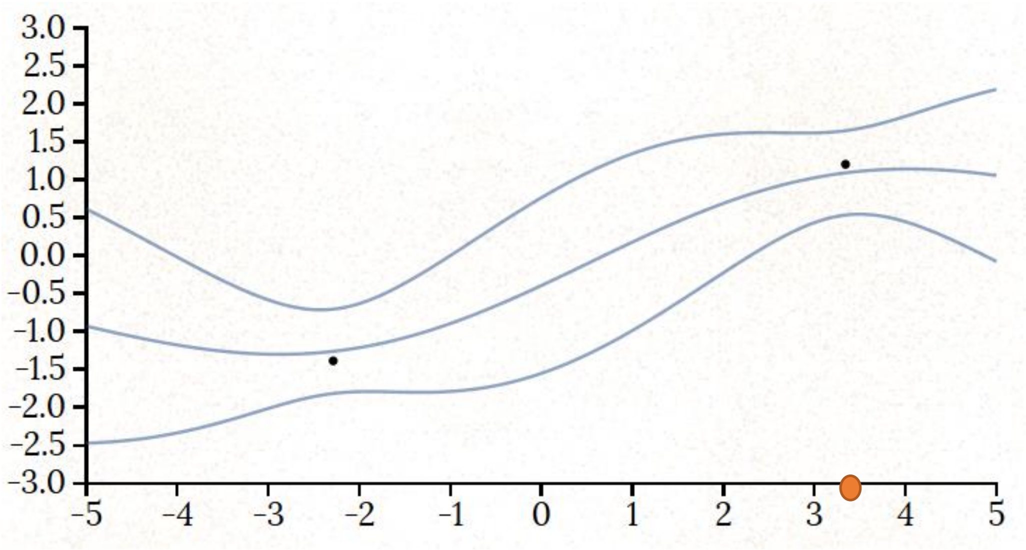


- This will work because our functions will be inside the CIs of the GP

GP Demo: http://www.tmpl.fi/gp/

Gaussian Process Bandit (5/5) Thompson Sampling

- Thompson sampling:
  - Draw $f \sim GP(\mu_t(x), k_t(x, x'))$
  - Play $x_{t+1} = \arg max_{x \in X} f(x)$



- This will work because our functions will be inside the CIs of the GP

GP Demo: http://www.tmpl.fi/gp/

Summary

- Multi-armed Bandit problem
  - Exploration exploitation trade-off
  - Examples
- Contextual Bandits
  - Dependent arms
- GP for bandits
  - GP-UCB
  - Thompson Sampling
- Summary

References

- Srinivas, Niranjan, et al. "Gaussian process optimization in the bandit setting: No regret and experimental design." *arXiv preprint arXiv:0912.3995* (2009).
- Brochu, Eric, Vlad M. Cora, and Nando De Freitas. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning." *arXiv preprint arXiv:1012.2599* (2010).