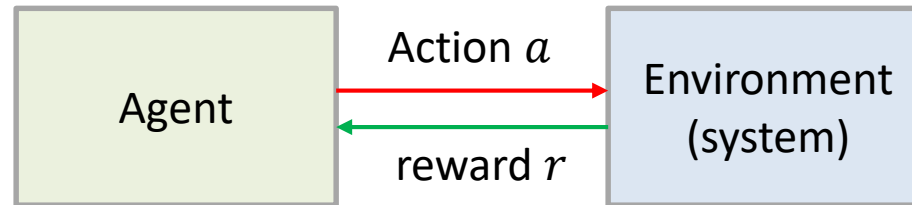


L12. Bandit Problem

Introduction

Learning by taking action

- Take action, observe reward, infer environment(system) behavior and take a better action



Two types of feedbacks can be used to infer environment (system)

- **Evaluative feedback** – how good the action taken is
 - ✓ Basis of reinforcement learning
 - ✓ Require extensive search among the alternative actions
- **Instructive feedback** – indicates the correct action to take
 - ✓ Basis of supervised learning
 - ✓ Independent of the action taken -> whatever action taken, the right one should have been taken will be revealed.
 - ✓ Search over a parameter space

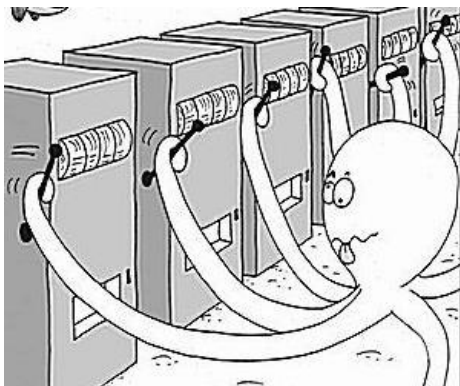
Before discussing Reinforcement learning, we first study the evaluative aspect of reinforcement learning in a simplified setting

From Bandit to Reinforcement Learning

		Model is unknown	
		No state dynamic	State dynamic (action affect state transition and future reward)
Optimum action	Single state (non-associative)	Bandit	-
Optimum policy	Multiple state (associative)	Contextual Bandit	Reinforcement learning

- Contextual Bandit (associative search task) is intermediate between the n -armed bandit problem and full reinforcement learning problem

An n -Armed Bandit Problem



- There are n machines
- Each machine i returns a reward $r \sim P(\theta_i)$: θ_i is unknown
- $a_t \in \{1, \dots, n\}$: the choice of machine at time t
- r_t : the reward at time t
- Policy π maps all the history to new action:

$$\pi: [(a_1, r_1), (a_2, r_2), \dots, (a_{t-1}, r_{t-1})] \rightarrow a_t$$

Find the optimal policy π^* that maximizes $E[\sum_{t=1}^T r_t]$ or $E[r_T]$

Acquiring new information
(exploration)

trade-off

capitalizing on the information
available so far
(exploitation)

Internet add : Advertising showing strategy for users

Finance : Portfolio optimization under unknown return profiles (risk vs mean profit)

Health care : Choosing the best treatment among alternatives

Internet shopping : Choosing the optimum price (sales v.s. profits)

Experiment design : Sequential experimental design (or sequential simulation parameter)

An n -Armed Bandit Problem

The value of the action $Q^*(a)$:

The true value of action a is defined as the mean reward received when that action is selected

$$Q^*(a) = E[r|a]$$

At any time there will be at least one action whose **value of the action** is the largest

- **Select greedy action (Exploiting)**: the action with the largest value of action
- **Select non-greedy action (Exploring)**: the action enabling you to improve your estimate of the non-greedy actions' value

Exploiting

Go to a company to make money right after undergraduate



Exploring

Go to a graduate school to explore my intellectual capability?



How to balance between exploitation and exploration ?

Action-Value Methods

- Action-value method estimates the value of an action and uses this to select the action
- Natural way to estimate the *action-value* is averaging the rewards actually received when the action was selected
- If at time t , **action a has been chosen k_a times prior to t** , yielding rewards, r_1, r_2, \dots, r_{k_a} , the estimated action value $Q_t(a)$ for a at time t is defined as

$$Q_t(a) = \frac{r_1 + r_2 + \dots + r_{k_a}}{k_a}$$

If $k_a = 0$, $Q_t(a) = 0$
If $k_a = \infty$, $Q_t(a) \rightarrow Q^*(a)$

- Greedy action selection rule is

$$a_t = \operatorname{argmax}_a Q_t(a)$$

- ✓ Always exploits current knowledge to maximize immediate reward, that is, it spends no time at all sampling apparently inferior actions to see if they might really be better

Action-Value Methods

- If at t th play, action a has been chosen k_a times prior to t , yielding rewards, r_1, r_2, \dots, r_{k_a} , the estimated action value for a is defined as

$$Q_t(a) = \frac{r_1 + r_2 + \dots + r_{k_a}}{k_a}$$

If $k_a = 0$, $Q_t(a) = 0$
If $k_a = \infty$, $Q_t(a) \rightarrow Q^*(a)$

greedy action selection rule

$$a_t = \underset{a}{\operatorname{argmax}} Q_t(a)$$

- ✓ Always exploits current knowledge to maximize immediate reward
- ✓ spends no time at all sampling apparently inferior actions to see if they might really be better

ϵ - *greedy* action selection rule

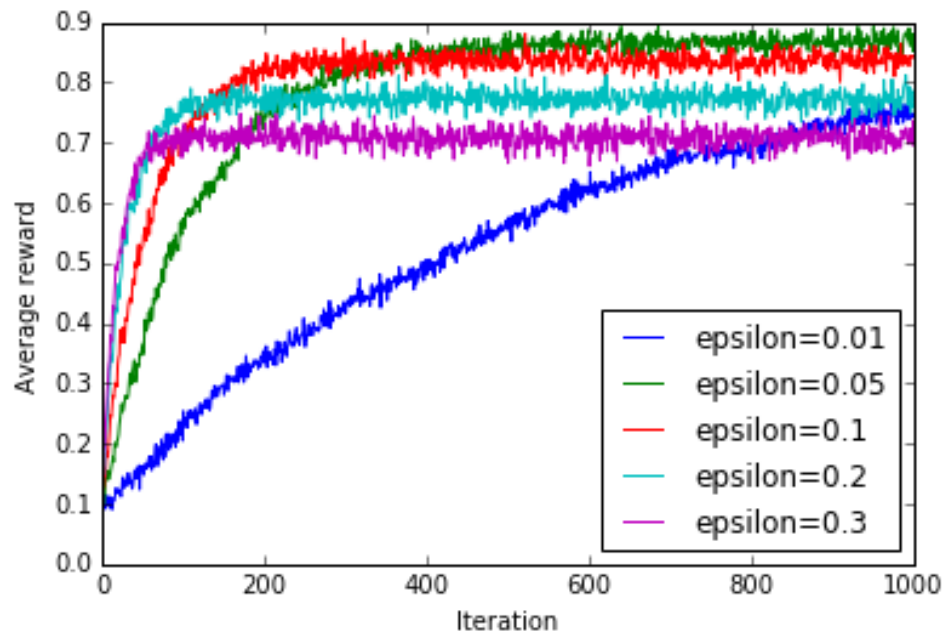
$$\pi(a) = \begin{cases} 1 - \epsilon + \epsilon/|A(s)| & \text{If } a = a^* \\ \epsilon/|A(s)| & \text{If } a \neq a^* \end{cases}$$

$$\left(1 - \epsilon + \frac{\epsilon}{|A(s)|}\right) \times 1 + \frac{\epsilon}{|A(s)|} (|A(s)| - 1) = 1$$

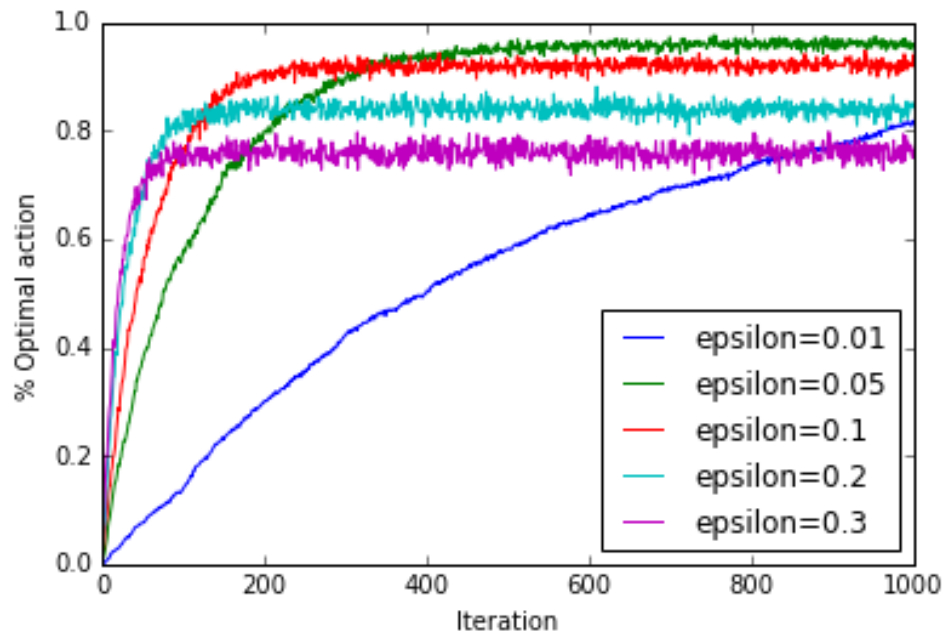
- ✓ In the limit as the number of plays increases, every a will be sampled an infinite number of times, guaranteeing $k_a \rightarrow \infty$ thus $Q_t(a) \rightarrow Q^*(a)$
- ✓ The probability of selecting the optimum action converges to a value greater than $1 - \epsilon$

Action-Value Methods

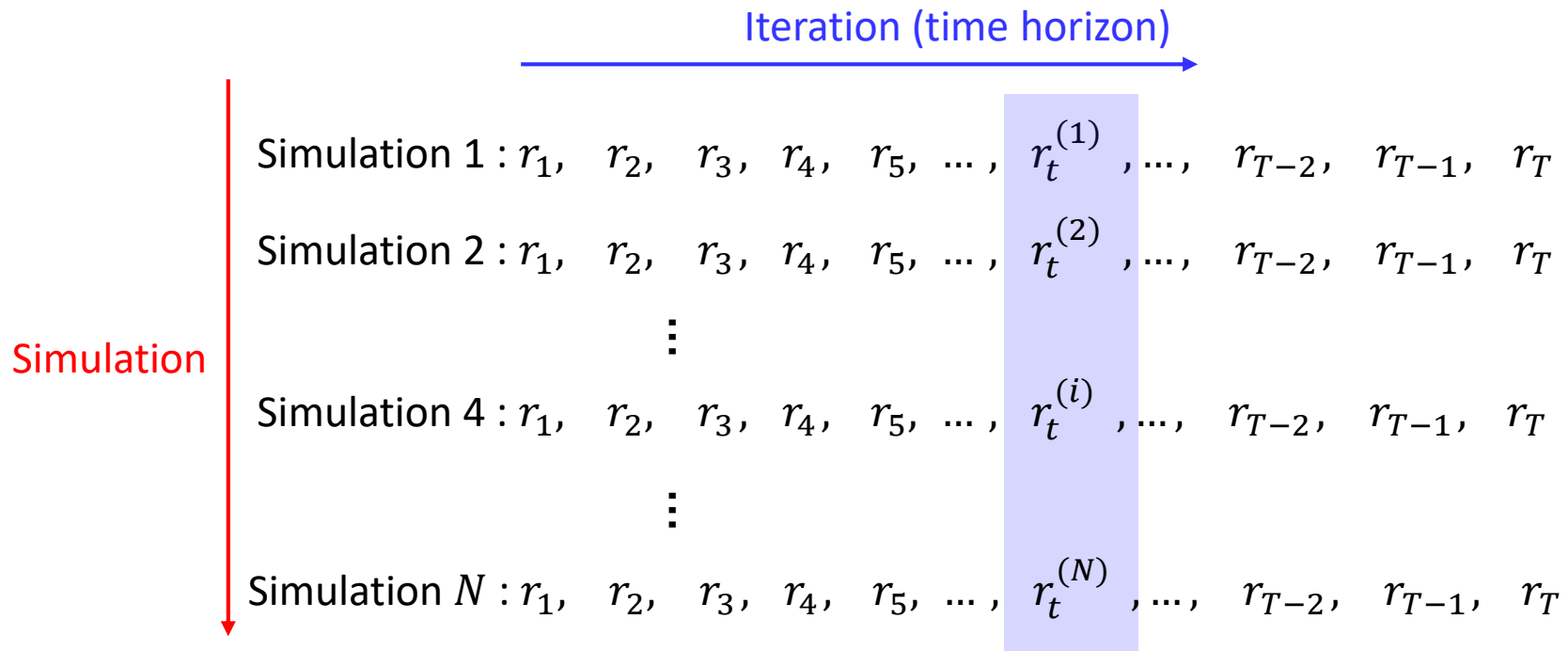
Average reward $E[r_t]$



% of selecting a^*



Obtaining the performance curve



$$E[r_t] = \frac{1}{N} \sum_{i=1}^N r_t^{(i)}$$

$$\% \text{ of } a^* = \frac{\sum_{i=1}^N \mathbf{1}\{a_t^i = a^*\}}{N}$$

Effectiveness of $\epsilon - greedy$ method depends on the task

- If the reward variance is large, it would take more exploration to find the optimal action
- If the reward variance is 0 (deterministic), the $\epsilon - greedy$ method would know the true value of each action after trying it once. After finding the true reward, it is best to seek to exploit
- If the bandit task is non-stationary, exploration needs to continue (even in the deterministic case) to find a better action than to choose the current best action greedily

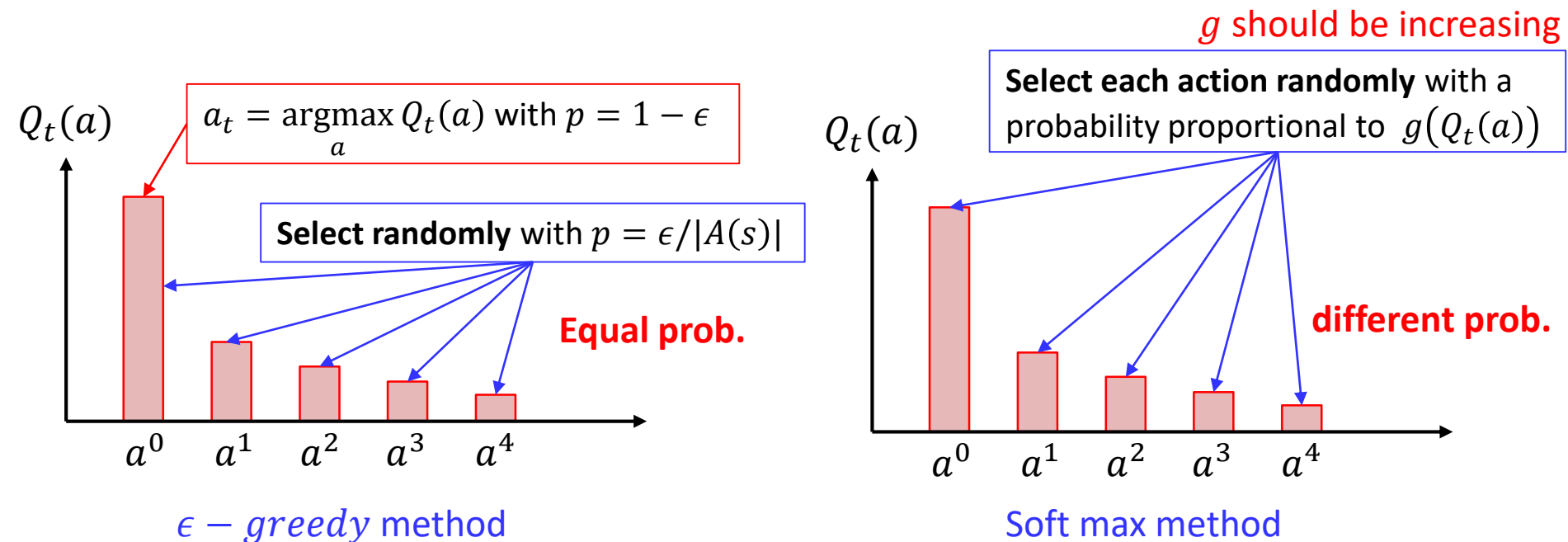
Softmax Action Selection

Disadvantage in ϵ – *greedy* action selection:

- When it explores, it chooses equally among all actions. That is, it is as likely to choose the worst-appearing action as it is to choose the next-to best action

Solution:

- Vary the action probabilities as a graded function of estimated value
- The greedy action is still given the highest selection probability, but all the others are ranked and weighted according to their values estimates



Softmax Action Selection

Disadvantage in ϵ – *greedy* action selection:

- When it explores, it chooses equally among all actions. That is, it is as likely to choose the worst-appearing action as it is to choose the next-to best action

Solution:

- Vary the action probabilities as a graded function of estimated value
- The greedy action is still given the highest selection probability, but all the others are ranked and weighted according to their values estimates

Softmax action selection rule

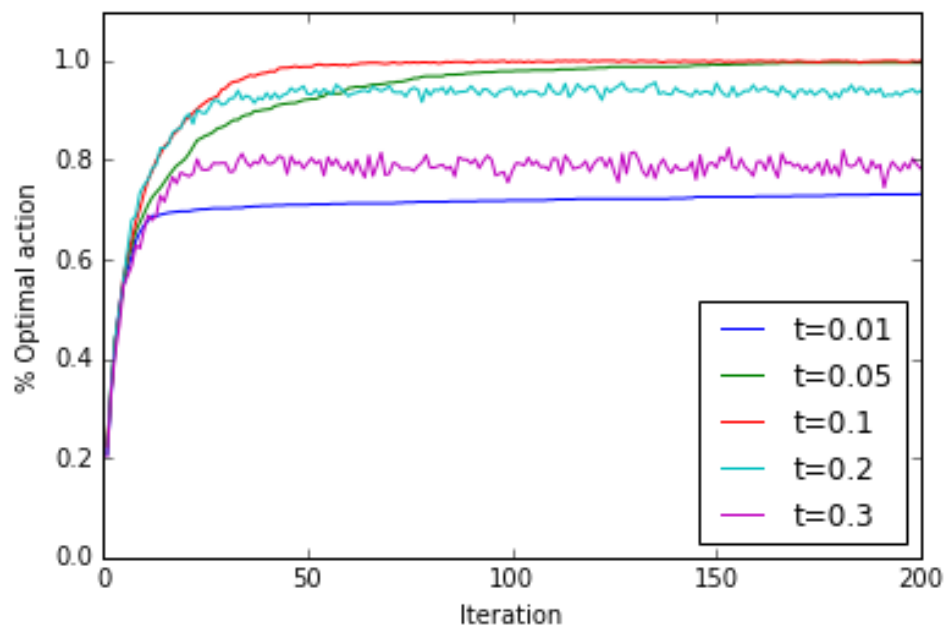
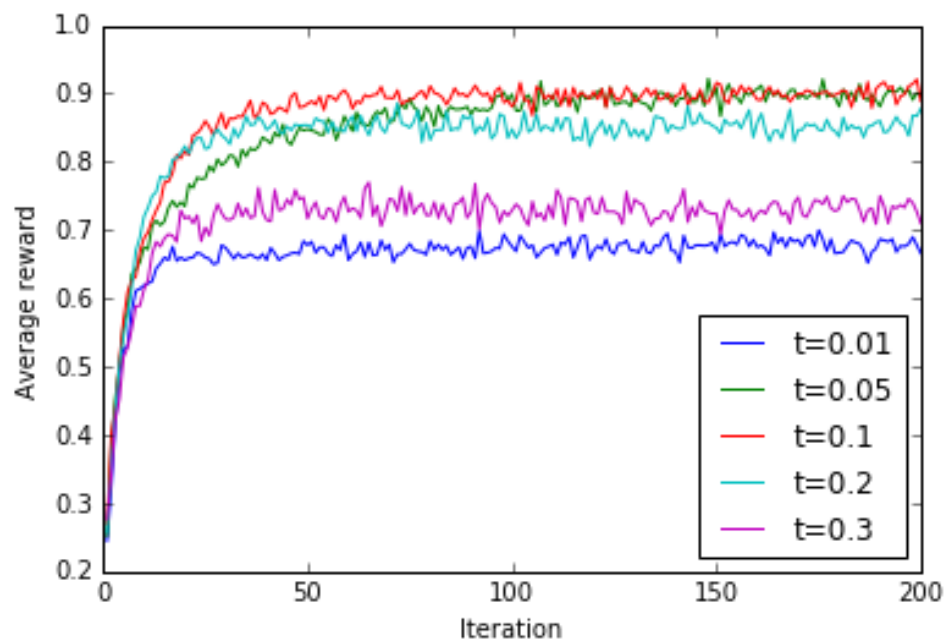
It chooses action a on the t th play with probability

$$\frac{e^{Q_t(a)/\tau}}{\sum_{b=1}^n e^{Q_t(b)/\tau}}$$

- ✓ τ is a positive parameter called the temperature:
 - high $\tau \rightarrow$ all actions are equiprobable
 - low $\tau \rightarrow$ greater difference in selection probability for action
 - When $\tau = 0$, softmax selection rule become greedy one

Whether softmax action selection or greedy action selection rule is better is unclear and may depend on the task and on human factors (i.e., setting τ and ϵ)

Softmax Action Selection Example



Evaluative vs. Instructive feedback



When I pulled No. 3,

- **Evaluative feedback** says “you earn 1\$”
 - ✓ Correctness is a **relative property** of actions that can be determined only by trying them all and comparing their rewards
- **Instructive feedback** says “The best machine is No.2”
 - ✓ No need to try variety of actions
 - ✓ Condensed information
 - ✓ Regardless of the action chosen

Incremental Implementation

At t th play, action a has been chosen k_a times prior to t , yielding rewards, r_1, r_2, \dots, r_{k_a} , the **estimated action value** for a is defined as

$$Q_t(a) = \frac{r_1 + r_2 + \dots + r_{k_a}}{k_a}$$

Issues in estimating the action value:

- It needs to store the reward history $r_1 + r_2 + \dots + r_{k_a}$ for all different actions
 - ✓ Memory and computational requirements grow over time without bound

Incremental Update formula

$$\begin{aligned} Q_{k+1} &= \frac{1}{k+1} \sum_{i=1}^{k+1} r_i \\ &= \frac{1}{k+1} \left(r_{k+1} + \sum_{i=1}^k r_i \right) \\ &= \frac{1}{k+1} (r_{k+1} + kQ_k + Q_k - Q_k) \\ &= \frac{1}{k+1} (r_{k+1} + (k+1)Q_k - Q_k) \\ &= Q_k + \frac{1}{k+1} [r_{k+1} - Q_k] \end{aligned}$$

- This implementation requires memory only for Q_k and k , and only the small computation

Step-size parameter and non-stationary

Incremental Update formula for action value

$$Q_{k+1} = Q_k + \frac{1}{k+1} [r_{k+1} - Q_k]$$

$$NewEstimate \leftarrow OldEstimate + \underset{\alpha_k}{StepSize} [Target - OldEstimate]$$

- When the distribution of the reward is non-stationary, it is natural to weight recent rewards more heavily than long-past ones
- One of the most popular ways of doing this is to use a constant step-size parameter

$$Q_{k+1} = Q_k + \alpha [r_{k+1} - Q_k]$$

$$\begin{aligned} Q_k &= Q_{k-1} + \alpha [r_k - Q_{k-1}] \\ &= \alpha r_k + (1 - \alpha) Q_{k-1} \\ &= \alpha r_k + (1 - \alpha) [\alpha r_{k-1} + (1 - \alpha) Q_{k-2}] \\ &= \alpha r_k + (1 - \alpha) \alpha r_{k-1} + (1 - \alpha)^2 Q_{k-2} \\ &= \alpha r_k + (1 - \alpha) \alpha r_{k-1} + (1 - \alpha)^2 \alpha r_{k-2} + \dots + (1 - \alpha)^{k-1} \alpha r_1 + (1 - \alpha)^k Q_0 \\ &= (1 - \alpha)^k Q_0 + \sum_{i=1}^k \alpha (1 - \alpha)^{k-i} r_i \end{aligned}$$

Reward $\alpha(1 - \alpha)^{k-i}$ given to i th reward r_i is smaller when $k - i$ is larger (i.e., older reward is weighted less)

The estimate of Q_k varies more in response to the most recently receive rewards!

Optimistic Initial Values

- In Estimating the action value $Q_{k+1} = Q_k + \alpha[r_{k+1} - Q_k]$, the initial value Q_0 should be specified by user
- the initial value Q_0 can be used as **prior knowledge** about what level of rewards can be expected
- Optimistic initial value encourages exploration
 - ✓ When the reward corresponding the current action is lower than the initial, the method will explore (change another action) to find a better reward

$$Q_k = (1 - \alpha)^k Q_0 + \sum_{i=1}^k \alpha(1 - \alpha)^{k-i} r_i$$

Reinforcement Comparison

- Actions followed by large rewards should be made more likely to recur
 - Actions followed by small rewards should be made less likely to recur
 - If an action is taken and the environment returns a reward of 5, Is it good or bad?
- To make such a judgment, one must compare the reward with some standard or reference level, called the *reference reward*.

Reinforcement Comparison algorithm

- $p_t(a)$: The preference for each action at time t (not an actual action value)
- Action determination rule (soft max) :

$$\pi_t(a) = \frac{e^{p_t(a)/\tau}}{\sum_{b=1}^n e^{p_t(b)/\tau}}$$

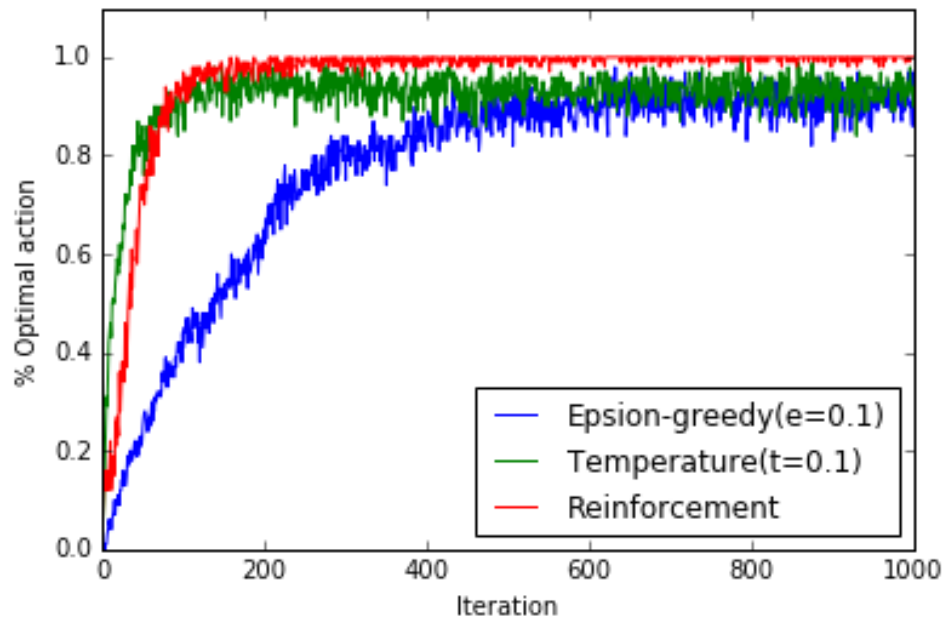
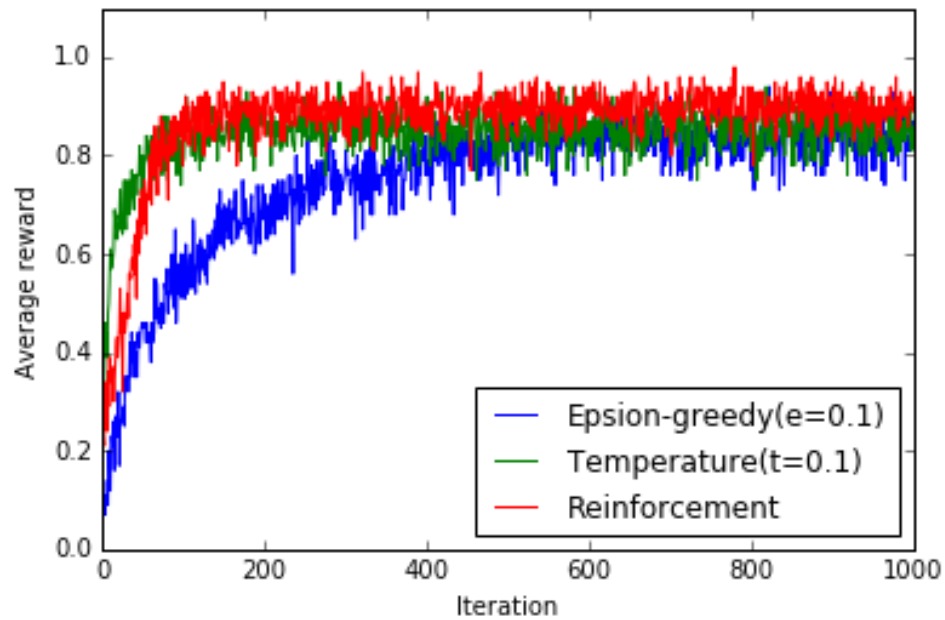
- After selecting action and observing reward, the preference $p_t(a)$ is updated :

$$p_{t+1}(a_t) = p_t(a_t) + \beta[r_t - \bar{r}_t]$$

- ✓ High rewards should increase the probability of reselecting the action taken
- ✓ β is a positive step-size parameter
- ✓ The reference reward \bar{r}_t (i.e., average reward) is updated using all recently received rewards whichever actions were taken:

$$\bar{r}_{t+1} = \bar{r}_t + \alpha[r_t - \bar{r}_t]$$

Reinforcement Comparison



Pursuit method maintain

- $Q_t(a)$: action–value estimate
 - $\pi_t(a)$: action preferences, the probability of selecting action a
- The estimation of the action value $Q_t(a)$ is **used to select the greedy action**

$$a_{t+1}^* = \operatorname{argmax}_a Q_{t+1}(a)$$

- **Action preferences continually pursuit the action that is greedy.** After each play, the probabilities are incremented toward 1 so as to make the greedy action more likely to be selected

$$\pi_{t+1}(a_{t+1}^*) = \pi_t(a_{t+1}^*) + \beta[1 - \pi_t(a_{t+1}^*)]$$

- Probability of selecting other actions are decremented toward zero:

$$\pi_{t+1}(a) = \pi_t(a) + \beta[0 - \pi_t(a)]$$

- The next action is selected according to the policy

$$a_{t+1} \sim \pi_{t+1}(a)$$

Pursuit Methods

Pursuit method maintain

- $Q_t(a)$: action–value estimate
- $\pi_t(a)$: action preferences, the probability of selecting action a
- The estimation of the action value $Q_t(a)$ is **used to select the greedy action**

$$a_{t+1}^* = \operatorname{argmax}_a Q_{t+1}(a)$$

- **Action preferences continually pursuit the action that is greedy.** After each play, the probabilities are incremented toward 1 so as to make the greedy action more likely to be selected

$$\pi_{t+1}(a_{t+1}^*) = \pi_t(a_{t+1}^*) + \beta[1 - \pi_t(a_{t+1}^*)]$$

- Probability of selecting other actions are decremented toward zero:

$$\pi_{t+1}(a) = \pi_t(a) + \beta[0 - \pi_t(a)]$$

- The next action is selected according to the policy

$$a_{t+1} \sim \pi_{t+1}(a)$$

$$Q_t(a) \rightarrow \pi_t(a) \rightarrow a_{t+1} + r_{t+1} \rightarrow Q_{t+1}(a)$$

Upper Confidence Bound (UBC method)

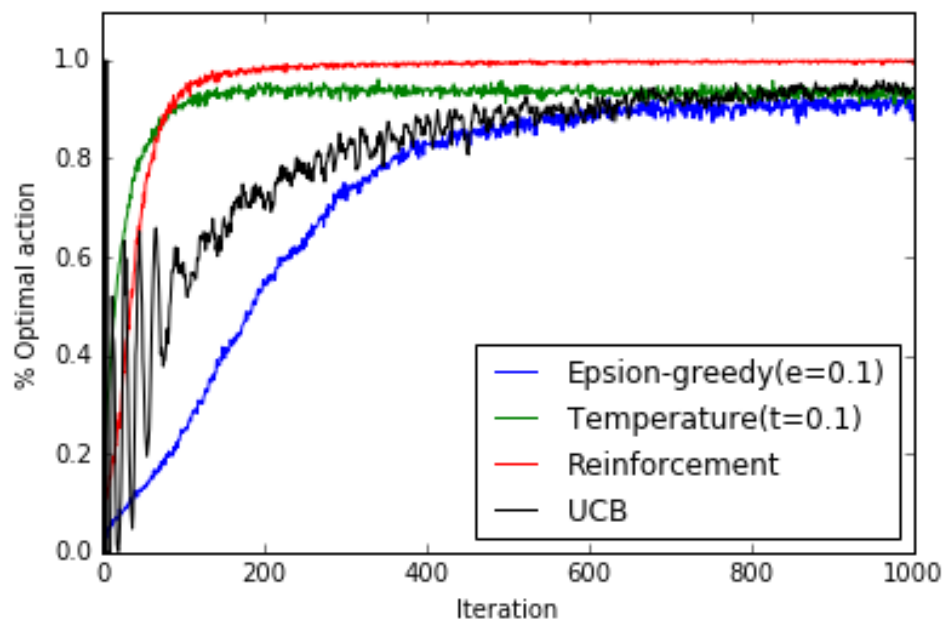
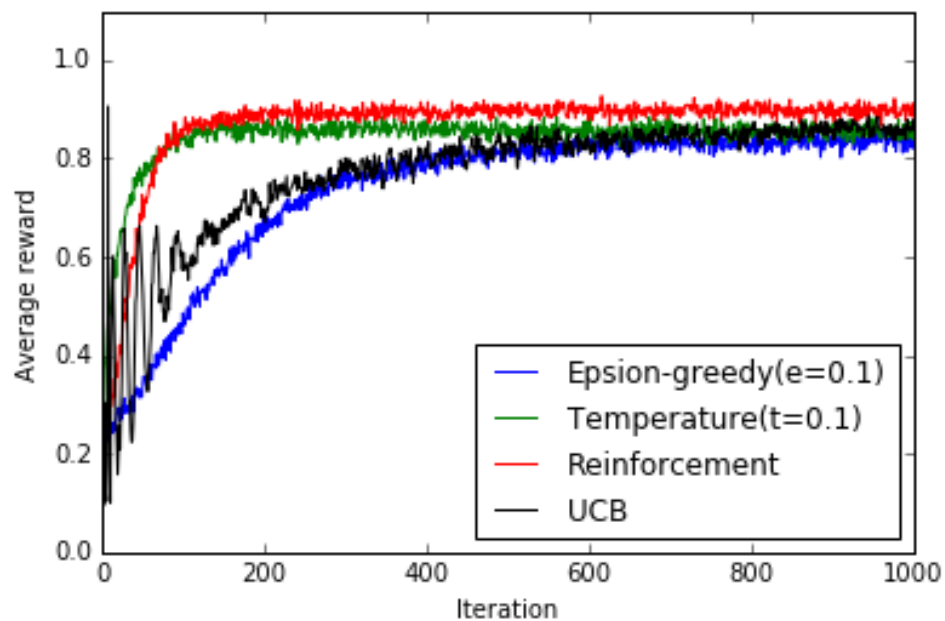
Use confidence intervals, or some other measure of uncertainty, to encourage exploration of particular actions is sound and appealing

- **Upper Confidence Bound (Cesa-Bianchi and Fisher, 2002)**

$$a_t = \operatorname{argmax}_i \left(\mu_i + \sqrt{\frac{2 \ln t}{n_i}} \right)$$

t : the total counts
 n_i : number of pulling for arm i

Upper Confidence Bound (UCB method)



From Bandit to Reinforcement Learning

		Model is unknown	
		No state dynamic	State dynamic (action affect state transition and future reward)
Optimum action	Single state (non-associative)	Bandit	-
Optimum policy	Multiple state (associative)	Contextual Bandit	Reinforcement learning

- Associative task involves both
 - ✓ trial-and-error learning in the form of searching for the best actions
 - ✓ association of these actions with the situations in which they are best.
- Contextual Bandit (associative search task) is intermediate between the n -armed bandit problem and full reinforcement learning problem

Relationship with model based approach

		Only Exploitation	Exploration vs Exploitation
		Model is known	Model is unknown
Optimum action	Single state	Optimization	Bandit
Optimum policy	Multiple state	Dynamic Programing	Contextual Bandit
			Reinforcement learning

- We're going to learn Dynamic Programming approach first, and move to Reinforcement learning

Bayesian View on Bandit Problem (MDP formulation)

The posterior of the success probability given w_t winnings and l_t loss :

$$\theta_t | w_t, l_t \sim \text{Beta}(\theta | \alpha = 1 + w_t, \beta = 1 + l_t)$$

The mean probability of success :

$$\rho_i = \int_0^1 \theta \text{Beta}(\theta | \alpha = 1 + w_i, \beta = 1 + l_i) d\theta = \frac{w_i + 1}{w_i + l_i + 2}$$



data

w

MLE

$$\theta = \frac{1}{1}$$

Bayesian

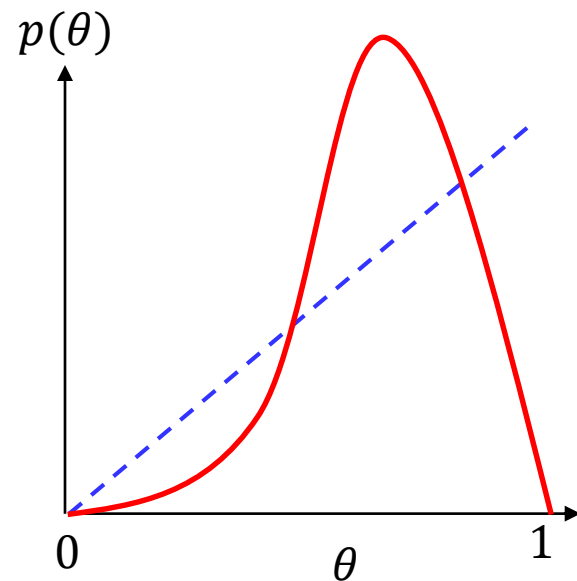
$$\theta | \text{data} \sim \text{Beta}(2, 1)$$



w, w, w, l, w

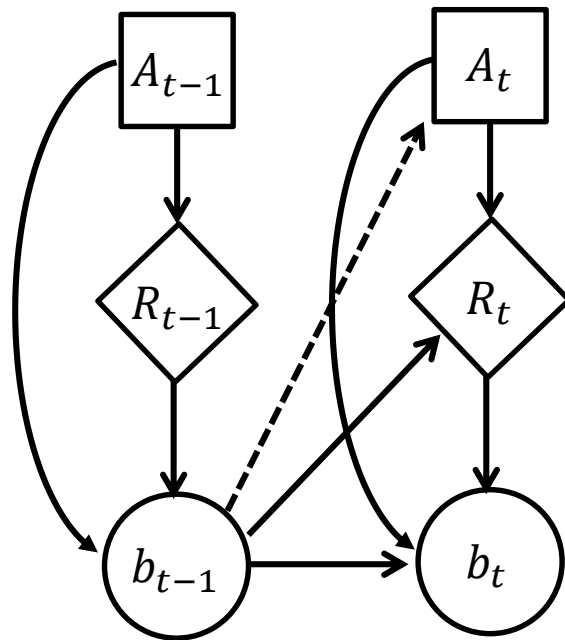
$$\theta = \frac{4}{5}$$

$$\theta | \text{data} \sim \text{Beta}(5, 2)$$



Bayesian View on Bandit Problem (MDP formulation)

MDP over belief state and finding optimal policy using Dynamic Programming



- $h_t = [(a_1, r_1), (a_1, r_1), \dots, (a_t, r_t)]$
- $\theta = (\theta_1, \dots, \theta_n)$: Unknown machine parameters
- Belief state $b_t(\theta) = P(\theta|h_t)$: probability dist. on para.
- Updating **belief state** $b_t(\theta)$ for Binary bandit with prior $\text{Beta}(\theta_i|\alpha, \beta)$: **Deterministic**

$$b_t(\theta_i) = b_{t-1}[a_t, r_t] = \text{Beta}(\theta_i|\alpha + w_{t,i}, \beta + l_{t,i})$$

$w_{t,i}$: Accumulated wins with arm i up to time t

$l_{t,i}$: Accumulated loses with arm i up to time t

Dynamic programming on the value function

$$V_{t-1}(b_{t-1}) = \max_{\pi} E \left[\sum_{t=t}^T r_t \right]$$

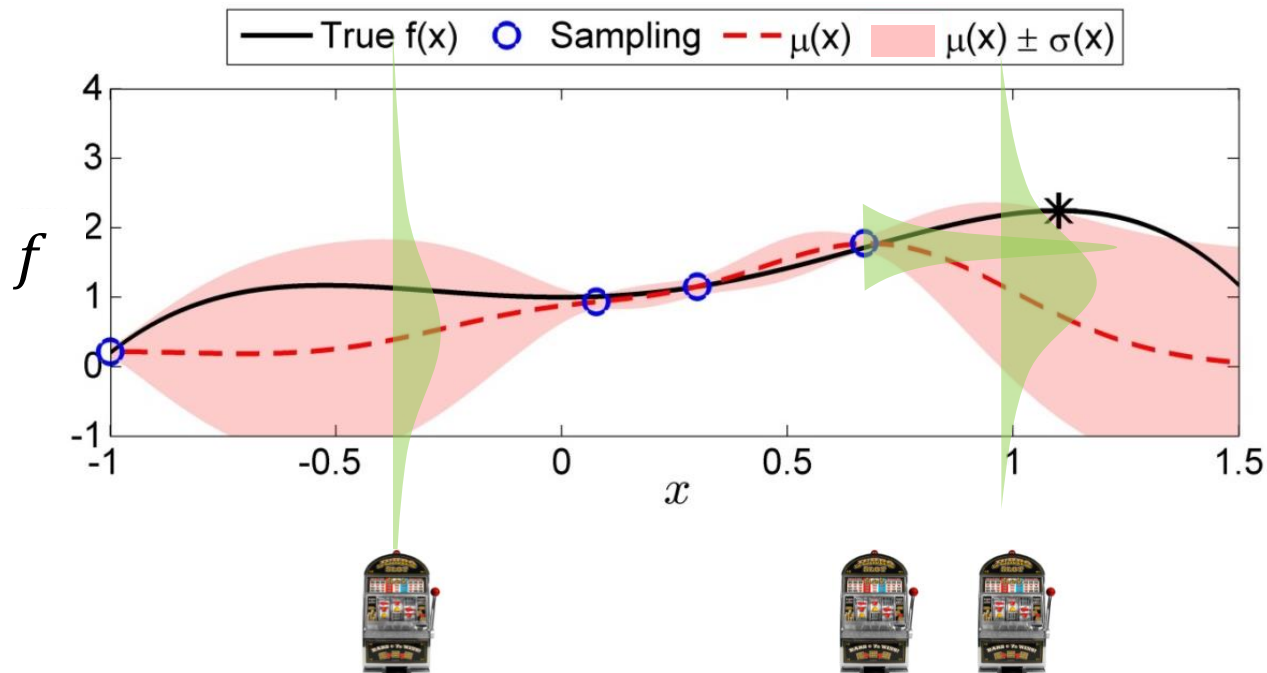
$$= \max_{a_t} \sum_{r_t} P(r_t|a_t, b_{t-1}) [r_t + V_t(b_{t-1}[a_t, r_t])]]$$

$$P(r|a, b) = \int_{\theta_a} b(\theta_a) P(r|\theta_a) d\theta_a$$

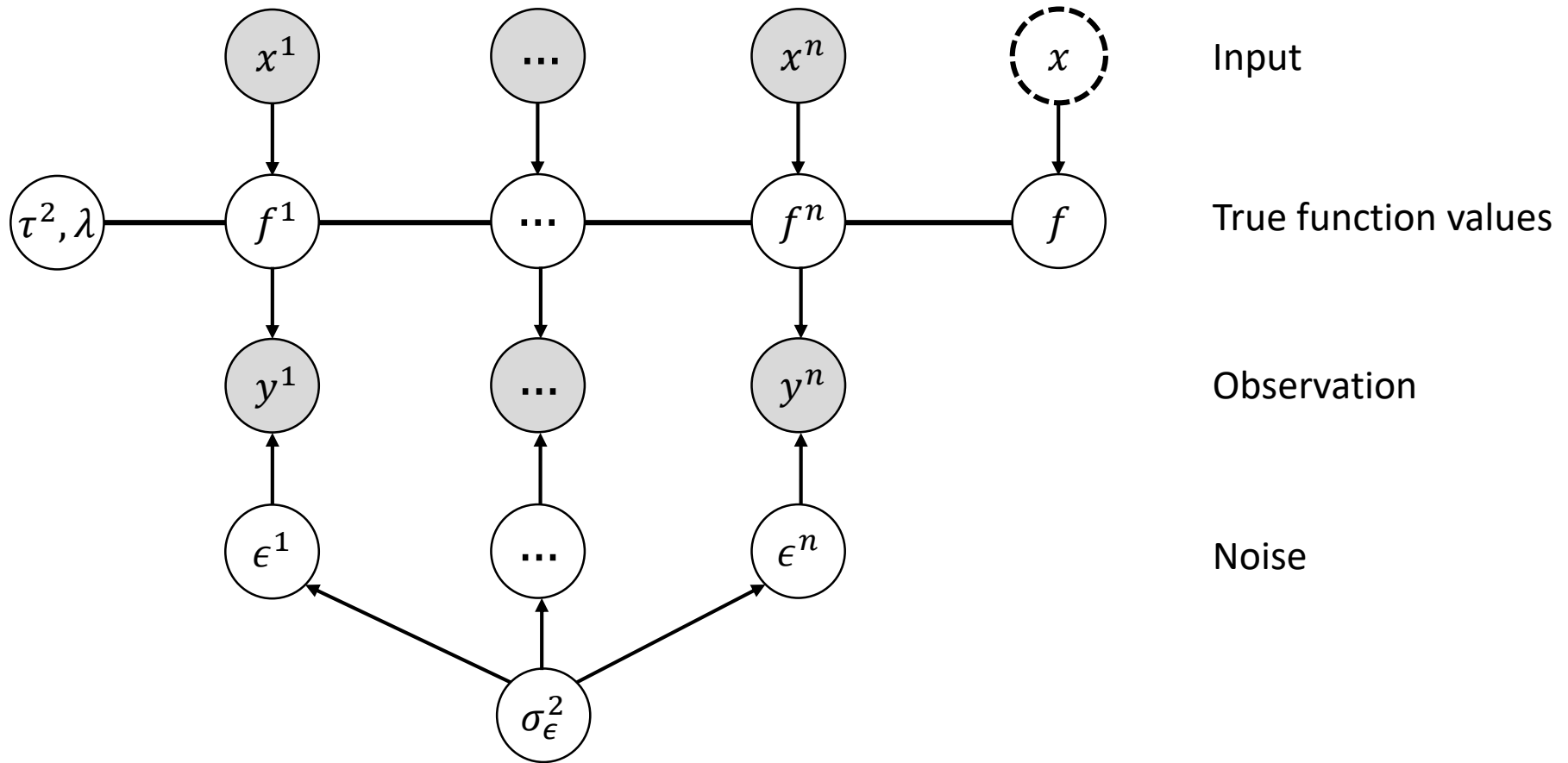
∞ – armed Bandit Problem

How to solve?

- Learn target function (exploration)
- Improve target value (exploitation)

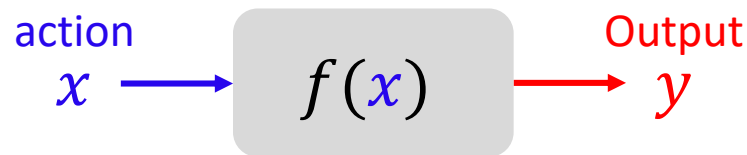


Gaussian Process



Bayesian Optimization

- **Bayesian Optimization (BO)** is a method to maximize (or minimize) a target value using measurement data from the **unknown target system**.
- BO is composed of three iterative steps:
 - (1) Learning : construct a probabilistic model function for a target system
 - (2) Optimization : select the next trial input to improve a target
 - (3) Observation : measure the output of a target system



- Policy π maps all the history to new action:

$$\pi: [(x^1, y^1), (x^2, y^2), \dots, (x^{n-1}, y^{n-1})] \rightarrow x^n$$

- Find the optimal policy π^* that maximizes $E[\sum_{t=1}^T y^t]$ or $E[y^T]$

$$x^* = \operatorname{argmax}_x f(x) \equiv E[y^T]$$

Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value $f = f(x)$ corresponding x

1. Given the data at n th iteration

Inputs

$$\mathbf{x}^{1:n} = \{x^1, \dots, x^i, \dots, x^n\}$$

Latent function values

$$\mathbf{f}^{1:n} = \{f^1, \dots, f^i, \dots, f^n\}$$

$y^i = f^i + \epsilon^i$
Observations

$$\mathbf{y}^{1:n} = \{y^1, \dots, y^i, \dots, y^n\}$$

Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value $f = f(\mathbf{x})$ corresponding \mathbf{x}

1. Given the data at n th iteration

$$\begin{array}{lll} \text{Inputs} & \text{Latent function values} & \text{Observations} \\ \mathbf{x}^{1:n} = \{\mathbf{x}^1, \dots, \mathbf{x}^i, \dots, \mathbf{x}^n\} & \mathbf{f}^{1:n} = \{f^1, \dots, f^i, \dots, f^n\} & \mathbf{y}^{1:n} = \{y^1, \dots, y^i, \dots, y^n\} \end{array}$$

$y^i = f^i + \epsilon^i$

2. Prior on the function values $\mathbf{f}^{1:n}$ is represented as Gaussian Process (GP)

$$p(\mathbf{f}^{1:n}) = GP(m(\cdot), k(\cdot, \cdot)) : \begin{bmatrix} f^1 \\ \vdots \\ f^n \end{bmatrix} \sim N \left(\begin{bmatrix} m(\mathbf{x}^1) \\ \vdots \\ m(\mathbf{x}^n) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}^1, \mathbf{x}^1) & \dots & k(\mathbf{x}^1, \mathbf{x}^n) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}^n, \mathbf{x}^1) & \dots & k(\mathbf{x}^n, \mathbf{x}^n) \end{bmatrix} \right)$$

$m(\cdot)$: mean function
 $k(\cdot, \cdot)$: kernel function

$$m(\mathbf{x}) = \mathbf{0} \quad k(\mathbf{x}, \mathbf{x}') = \gamma \exp \left(-\frac{1}{2} (\mathbf{x} - \mathbf{x}')^T \text{diag}(\boldsymbol{\lambda})^{-2} (\mathbf{x} - \mathbf{x}') \right)$$

$$\text{cov}(f^i, f^j) = E[(f^i - m(\mathbf{x}^i))(f^j - m(\mathbf{x}^j))] \approx k(\mathbf{x}^i, \mathbf{x}^j)$$

$$\text{Exponential Square kernel function: } k(\mathbf{x}^i, \mathbf{x}^j) = \gamma \exp \left(-\frac{1}{2} (\mathbf{x}^i - \mathbf{x}^j)^T \text{diag}(\boldsymbol{\lambda})^{-2} (\mathbf{x}^i - \mathbf{x}^j) \right)$$

Kernel function?

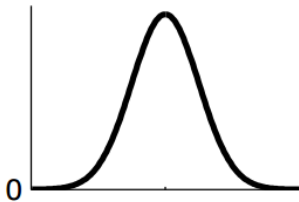
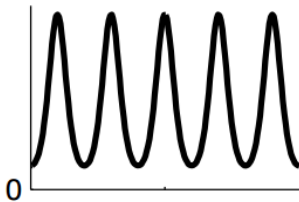
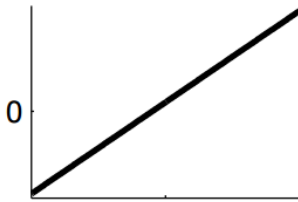
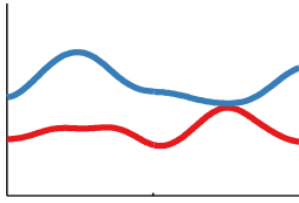
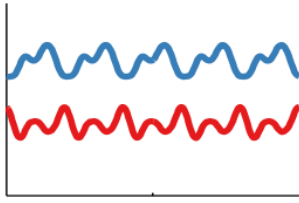
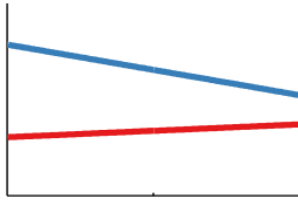
Kernel name:	Squared-exp (SE)	Periodic (Per)	Linear (Lin)
$k(x, x') =$	$\sigma_f^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$	$\sigma_f^2 \exp\left(-\frac{2}{\ell^2} \sin^2\left(\pi \frac{x-x'}{p}\right)\right)$	$\sigma_f^2 (x-c)(x'-c)$
Plot of $k(x, x')$:			
	$x - x'$ ↓	$x - x'$ ↓	x (with $x' = 1$) ↓
Functions $f(x)$ sampled from GP prior:			
	x	x	x
Type of structure:	local variation	repeating structure	linear functions

Figure from <http://www.cs.toronto.edu/~duvenaud/>

Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value $f = f(\mathbf{x})$ corresponding \mathbf{x}

1. Given the data at n th iteration

$$\begin{array}{lll} \text{Inputs} & \text{Latent function values} & \text{Observations} \\ \mathbf{x}^{1:n} = \{\mathbf{x}^1, \dots, \mathbf{x}^i, \dots, \mathbf{x}^n\} & \mathbf{f}^{1:n} = \{f^1, \dots, f^i, \dots, f^n\} & \mathbf{y}^{1:n} = \{y^1, \dots, y^i, \dots, y^n\} \end{array}$$

$y^i = f^i + \epsilon^i$

2. Prior on the function values $\mathbf{f}^{1:n}$ is represented as Gaussian Process (GP)

$$p(\mathbf{f}^{1:n}) = GP(m(\cdot), k(\cdot, \cdot)) : \begin{bmatrix} f^1 \\ \vdots \\ f^n \end{bmatrix} \sim N \left(\begin{bmatrix} m(\mathbf{x}^1) \\ \vdots \\ m(\mathbf{x}^n) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}^1, \mathbf{x}^1) & \dots & k(\mathbf{x}^1, \mathbf{x}^n) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}^n, \mathbf{x}^1) & \dots & k(\mathbf{x}^n, \mathbf{x}^n) \end{bmatrix} \right)$$

$m(\cdot)$: mean function
 $k(\cdot, \cdot)$: kernel function

$$m(\mathbf{x}) = \mathbf{0} \quad k(\mathbf{x}, \mathbf{x}') = \gamma \exp \left(-\frac{1}{2} (\mathbf{x} - \mathbf{x}')^T \text{diag}(\boldsymbol{\lambda})^{-2} (\mathbf{x} - \mathbf{x}') \right)$$

3. Likelihood is constructed base the assumption on the noise, i.e., i.i.d. Gaussian noise

$$p(\mathbf{y}^{1:n} | \mathbf{f}^{1:n}) = N(\mathbf{f}^{1:n}, \sigma_\epsilon^2 \mathbf{I})$$

The hyper-parameters $\boldsymbol{\theta} = (\sigma_\epsilon, \sigma_s, \boldsymbol{\lambda})$ for the noise model and the kernel function are determined as ones maximizing the marginal log-likelihood of the training data $\mathbf{D}^n = \{(\mathbf{x}^i, y^i) | i = 1, \dots, n\}$ as

$$\begin{aligned} \boldsymbol{\theta}^* &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p(\mathbf{y}^{1:n} | \boldsymbol{\theta}) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left(-\frac{1}{2} (\mathbf{y}^{1:n})^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n} - \frac{1}{2} \log |\mathbf{K} + \sigma_\epsilon^2 \mathbf{I}| - \frac{n}{2} \log 2\pi \right) \end{aligned}$$

Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value $f = f(\mathbf{x})$ corresponding \mathbf{x}

1. Given the data at n th iteration

$$\begin{array}{lll} \text{Inputs} & \text{Latent function values} & \text{Observations} \\ \mathbf{x}^{1:n} = \{\mathbf{x}^1, \dots, \mathbf{x}^i, \dots, \mathbf{x}^n\} & \mathbf{f}^{1:n} = \{f^1, \dots, f^i, \dots, f^n\} & \mathbf{y}^{1:n} = \{y^1, \dots, y^i, \dots, y^n\} \end{array}$$

$y^i = f^i + \epsilon^i$

2. Prior on the function values $\mathbf{f}^{1:n}$ is represented as Gaussian Process (GP)

$$p(\mathbf{f}^{1:n}) = GP(m(\cdot), k(\cdot, \cdot)) : \begin{bmatrix} f^1 \\ \vdots \\ f^n \end{bmatrix} \sim N \left(\begin{bmatrix} m(\mathbf{x}^1) \\ \vdots \\ m(\mathbf{x}^n) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}^1, \mathbf{x}^1) & \dots & k(\mathbf{x}^1, \mathbf{x}^n) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}^n, \mathbf{x}^1) & \dots & k(\mathbf{x}^n, \mathbf{x}^n) \end{bmatrix} \right)$$

$m(\cdot)$: mean function
 $k(\cdot, \cdot)$: kernel function

$$m(\mathbf{x}) = \mathbf{0} \quad k(\mathbf{x}, \mathbf{x}') = \gamma \exp \left(-\frac{1}{2} (\mathbf{x} - \mathbf{x}')^T \text{diag}(\boldsymbol{\lambda})^{-2} (\mathbf{x} - \mathbf{x}') \right)$$

3. Likelihood is constructed base the assumption on the noise, i.e., i.i.d. Gaussian noise

$$p(\mathbf{y}^{1:n} | \mathbf{f}^{1:n}) = N(\mathbf{f}^{1:n}, \sigma_\epsilon^2 \mathbf{I})$$

4. Joint distribution based on Bayes' rule :

$$p(f, \mathbf{y}^{1:n}) = \int p(f, \mathbf{f}^{1:n}) p(\mathbf{y}^{1:n} | \mathbf{f}^{1:n}) d\mathbf{f}^{1:n} \quad \Rightarrow \quad \begin{bmatrix} \mathbf{y}^{1:n} \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(\mathbf{x}, \mathbf{x}) \end{bmatrix} \right)$$

Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value $f = f(\mathbf{x})$ corresponding \mathbf{x}

Property 4 : Conditionals of a GRV are Gaussians, more specifically, if

$$\mathbf{Z} = \begin{bmatrix} Y_1 \\ - \\ Y_2 \end{bmatrix} \sim N \left(\begin{bmatrix} Y_1 \\ - \\ Y_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ - & - \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \right)$$

where Y_1 is k -dim RV and Y_2 is an $n - k$ dim RV, then

$$Y_2 | \{Y_1 = y\} \sim N(\Sigma_{21}\Sigma_{11}^{-1}(y - \mu_1) + \mu_2, \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12})$$

4. Joint distribution based on Bayes' rule :

$$p(f, \mathbf{y}^{1:n}) = \int p(f, \mathbf{f}^{1:n})p(\mathbf{y}^{1:n}|\mathbf{f}^{1:n}) d\mathbf{f}^{1:n} \quad \Rightarrow \quad \begin{bmatrix} \mathbf{y}^{1:n} \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(\mathbf{x}, \mathbf{x}) \end{bmatrix} \right)$$

5. Conditionalization \rightarrow **Posterior distribution on the function value $f = f(\mathbf{x})$ for \mathbf{x} given $\mathbf{D}^n = \{\mathbf{x}^{1:n}, \mathbf{y}^{1:n}\}$**

$$p(f|\mathbf{D}^n) = N(\mu(\mathbf{x}|\mathbf{D}^n), \sigma^2(\mathbf{x}|\mathbf{D}^n))$$

Mean : $\mu(\mathbf{x}|\mathbf{D}^n) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$

Variance : $\sigma^2(\mathbf{x}|\mathbf{D}^n) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$

Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value $f = f(x)$ corresponding x

1. Given the data at n th iteration

$$\begin{array}{lll} \text{Inputs} & \text{Latent function values} & \text{Observations} \\ \mathbf{x}^{1:n} = \{x^1, \dots, x^i, \dots, x^n\} & \mathbf{f}^{1:n} = \{f^1, \dots, f^i, \dots, f^n\} & \mathbf{y}^{1:n} = \{y^1, \dots, y^i, \dots, y^n\} \end{array}$$

$y^i = f^i + \epsilon^i$

2. Prior on the function values $\mathbf{f}^{1:n}$ is represented as Gaussian Process (GP)

$$p(\mathbf{f}^{1:n}) = GP(m(\cdot), k(\cdot, \cdot)) : \begin{bmatrix} f^1 \\ \vdots \\ f^n \end{bmatrix} \sim N \left(\begin{bmatrix} m(x^1) \\ \vdots \\ m(x^n) \end{bmatrix}, \begin{bmatrix} k(x^1, x^1) & \dots & k(x^1, x^n) \\ \vdots & \ddots & \vdots \\ k(x^n, x^1) & \dots & k(x^n, x^n) \end{bmatrix} \right)$$

$m(\cdot)$: mean function
 $k(\cdot, \cdot)$: kernel function

$$m(x) = \mathbf{0} \quad k(x, x') = \gamma \exp \left(-\frac{1}{2} (x - x')^T \text{diag}(\lambda)^{-2} (x - x') \right)$$

3. Likelihood is constructed base the assumption on the noise, i.e., i.i.d. Gaussian noise

$$p(\mathbf{y}^{1:n} | \mathbf{f}^{1:n}) = N(\mathbf{f}^{1:n}, \sigma_\epsilon^2 \mathbf{I})$$

4. Joint distribution based on Bayes' rule :

$$p(f, \mathbf{y}^{1:n}) = \int p(f, \mathbf{f}^{1:n}) p(\mathbf{y}^{1:n} | \mathbf{f}^{1:n}) d\mathbf{f}^{1:n} \quad \Rightarrow \quad \begin{bmatrix} \mathbf{y}^{1:n} \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$

5. Conditionalization \rightarrow Posterior distribution on the function value $f = f(x)$ for x given $\mathbf{D}^n = \{\mathbf{x}^{1:n}, \mathbf{y}^{1:n}\}$

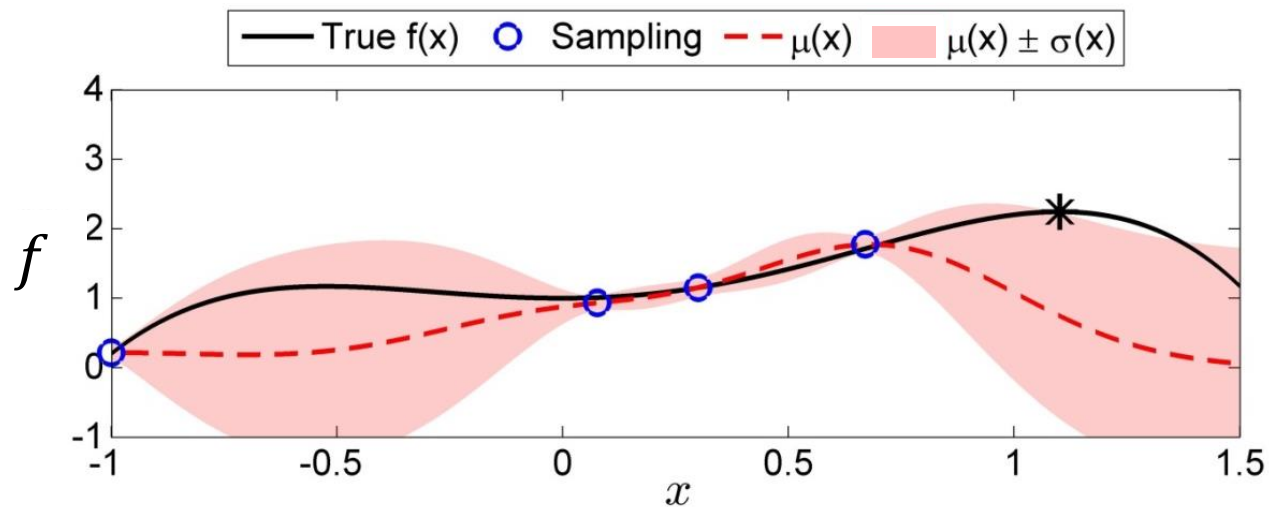
$$p(f | \mathbf{D}^n) = N(\mu(x | \mathbf{D}^n), \sigma^2(x | \mathbf{D}^n))$$

Mean : $\mu(x | \mathbf{D}^n) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$
Variance : $\sigma^2(x | \mathbf{D}^n) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$

Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value $f = f(x)$ corresponding x

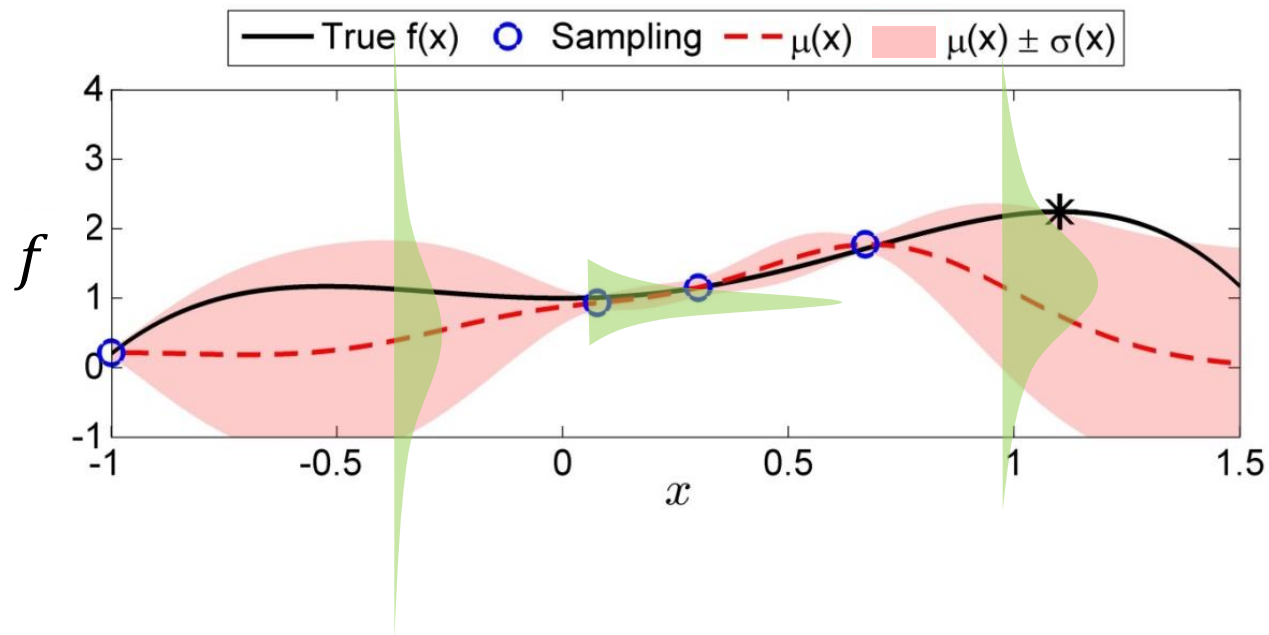
$$p(f|\mathbf{D}^n) = N(\mu(x|\mathbf{D}^n), \sigma^2(x|\mathbf{D}^n))$$



Learning phase : Gaussian Process (GP) regression

Construct the distribution on unknown target value $f = f(x)$ corresponding x

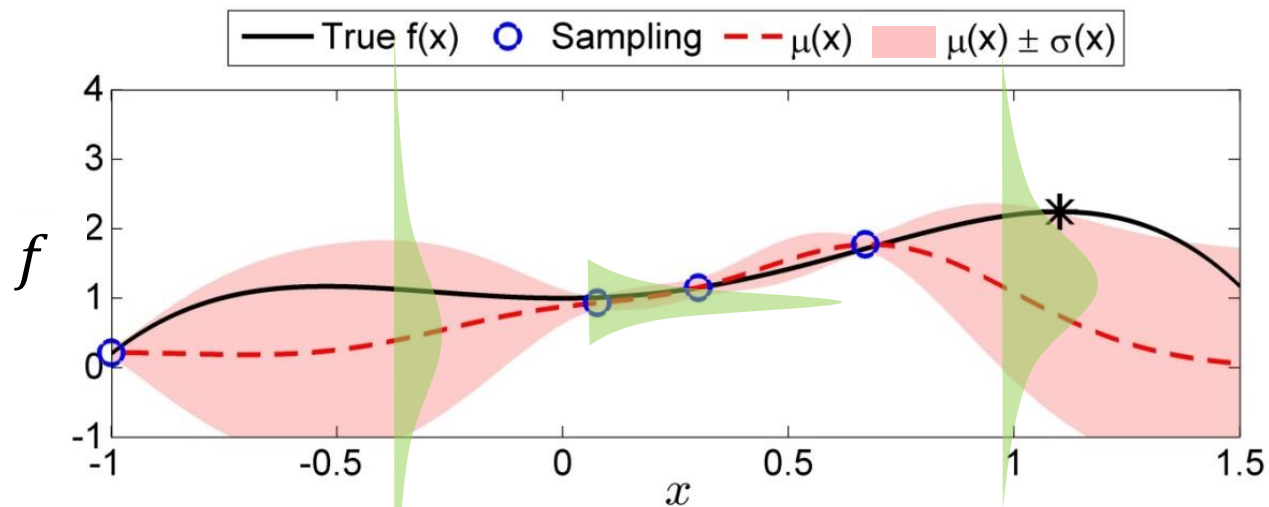
$$p(f|\mathbf{D}^n) = N(\mu(x|\mathbf{D}^n), \sigma^2(x|\mathbf{D}^n))$$



Optimization (Sampling) phase

How to select the next input ?

- Learn target function (exploration)
- Improve target value (exploitation)



Go to a graduate school to explore my intellectual capability?

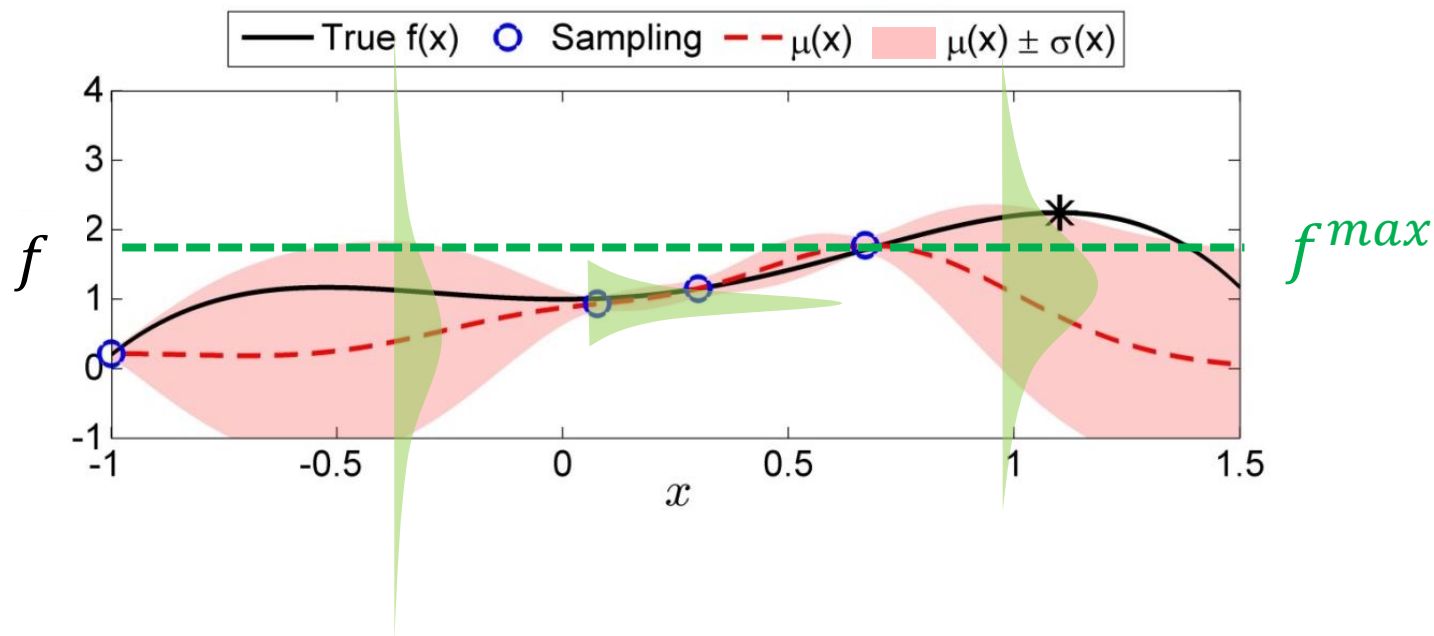


Go to a company to make money?

Optimization (Sampling) phase

How to select the next input ?

- Learn target function (exploration)
- Improve target value (exploitation)



Next sampling point x^{n+1} is determined by solving (Mockus *et al*, 1978)

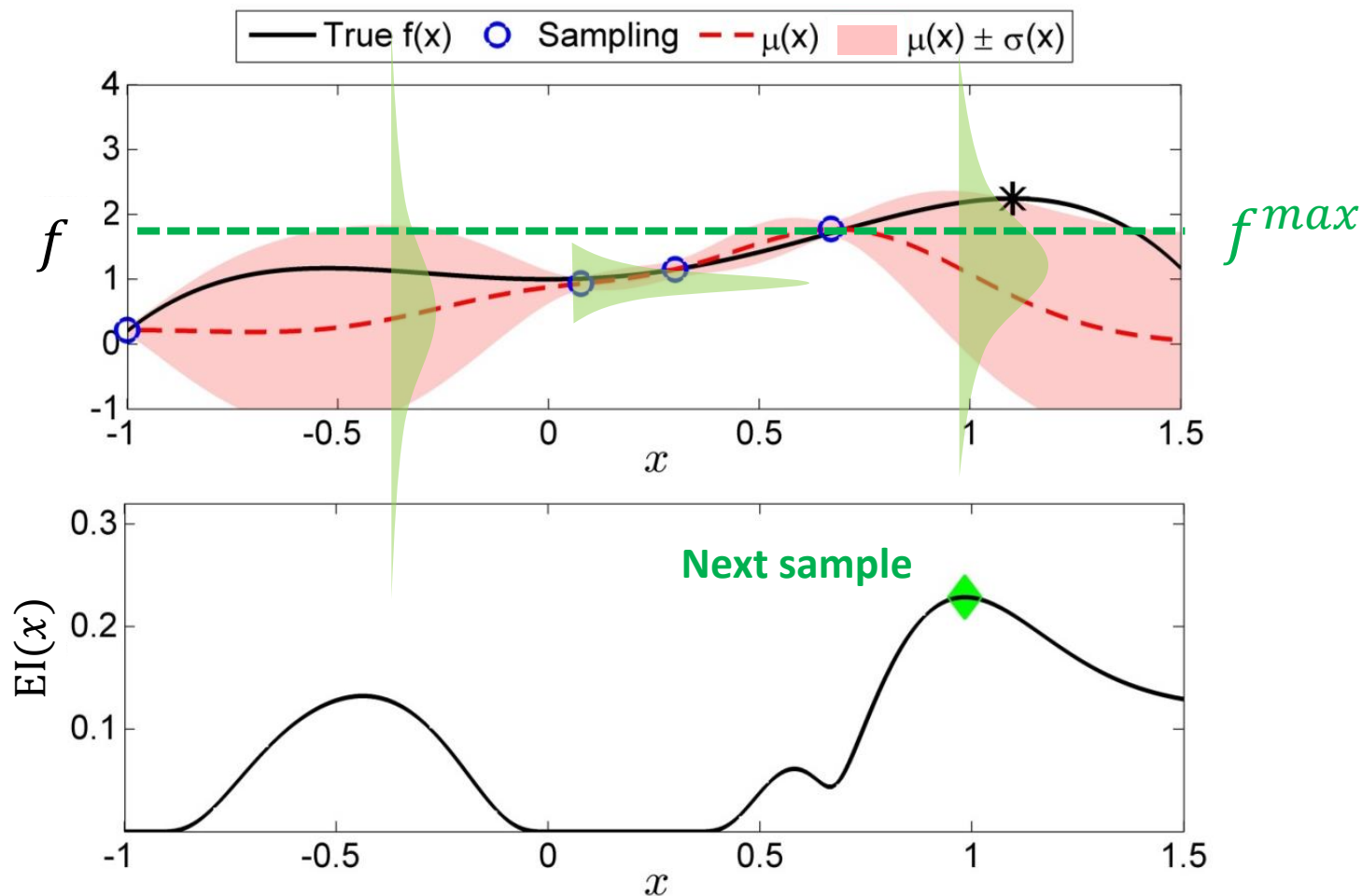
$$x^{n+1} = \arg \max_x \text{EI}(x) \triangleq \text{E}[\max\{0, f - f^{max}\} | \mathbf{D}^n]$$

DIRECT (Finkel, 2003), a gradient free optimization code, is used to solve this problem

Optimization (Sampling) phase

How to select the next input ?

- Learn target function (exploration)
- Improve target value (exploitation)



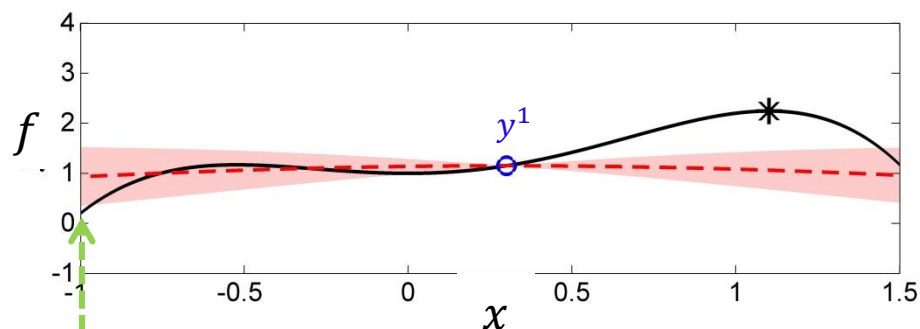
Bayesian Optimization

Illustrative example

$$\underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

$$\text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2)$$

$$y^1 \begin{bmatrix} 1.15 \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$



Construct probability distribution on the unknown function value

$$p(f|\mathbf{D}^{1:n}) = N(\mu(x|\mathbf{D}^{1:n}), \sigma^2(x|\mathbf{D}^{1:n}))$$

$$\mu(x|\mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

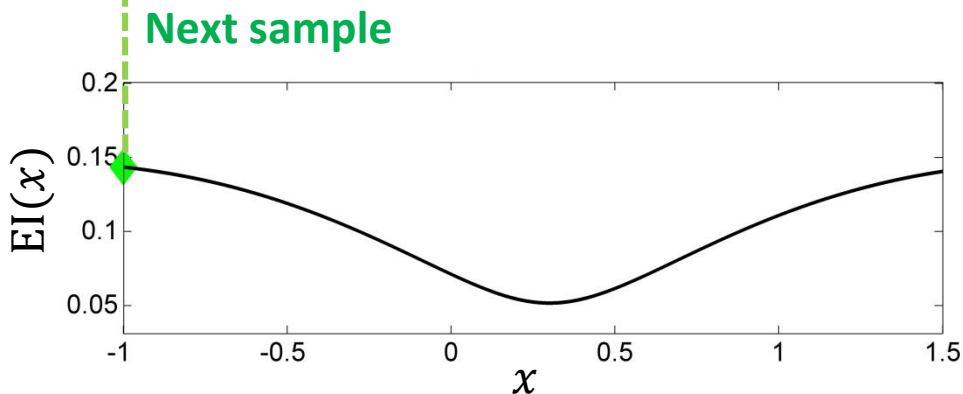
$$\mathbf{x}^2 = \arg \max_x \text{EI}(x) \triangleq \text{E}[\max\{0, f - f^{\max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^2 = -1.00$$

Query the function value

$$y^2 = f(\mathbf{x}^2) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^2 = 0.18$$

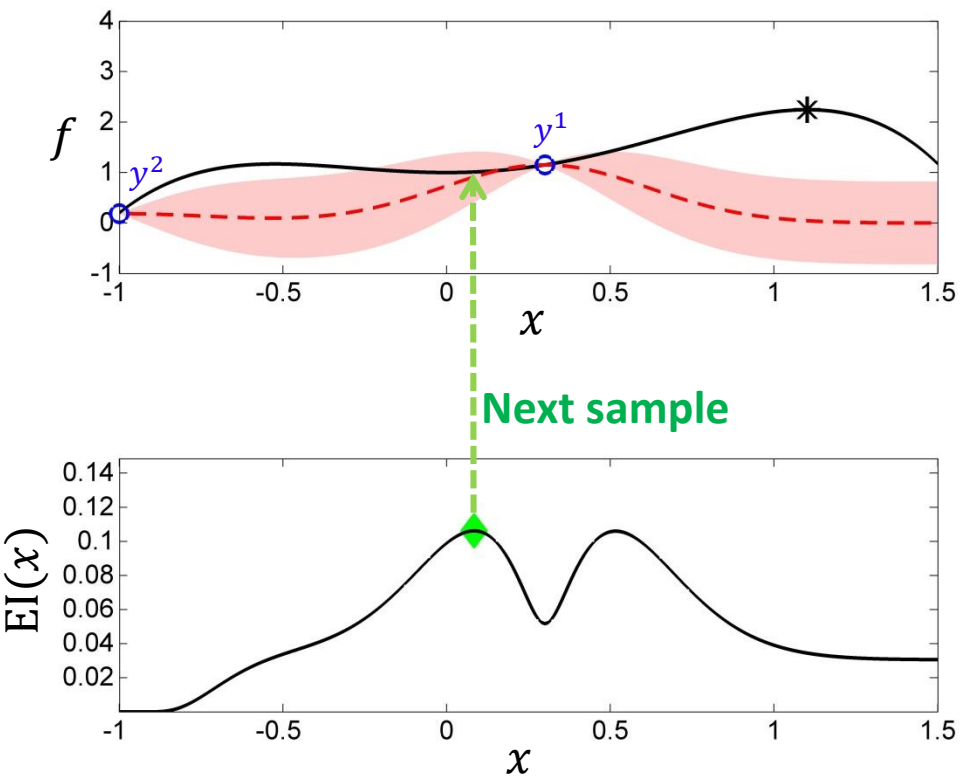


Bayesian Optimization

Illustrative example

$$\begin{aligned} & \underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$

$$\begin{matrix} y^1 \\ y^2 \\ f \end{matrix} \begin{bmatrix} 1.15 \\ 0.18 \\ f \end{bmatrix} \sim N \left(\begin{bmatrix} \mathbf{0} \\ \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} \\ \mathbf{k}^T \end{bmatrix}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$



Construct probability distribution on the unknown function value

$$p(f|\mathbf{D}^{1:n}) = N(\mu(x|\mathbf{D}^{1:n}), \sigma^2(x|\mathbf{D}^{1:n}))$$

$$\mu(x|\mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^3 = \arg \max_x \text{EI}(x) \triangleq \text{E}[\max\{0, f - f^{max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^3 = 0.03$$

Query the function value

$$\mathbf{x}^3 = f(\mathbf{x}^3) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

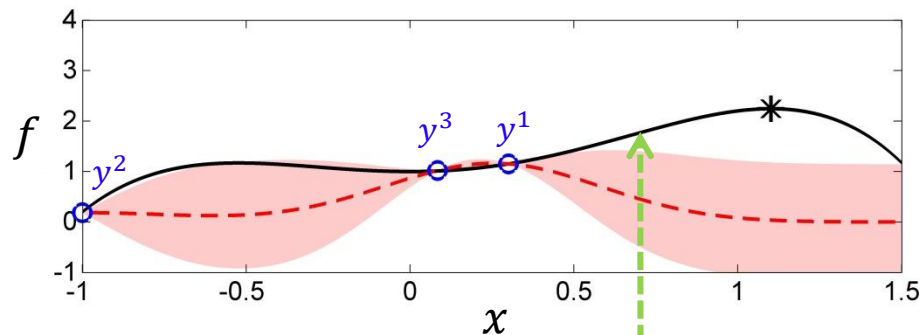
$$\mathbf{x}^3 = 1.02$$

Bayesian Optimization

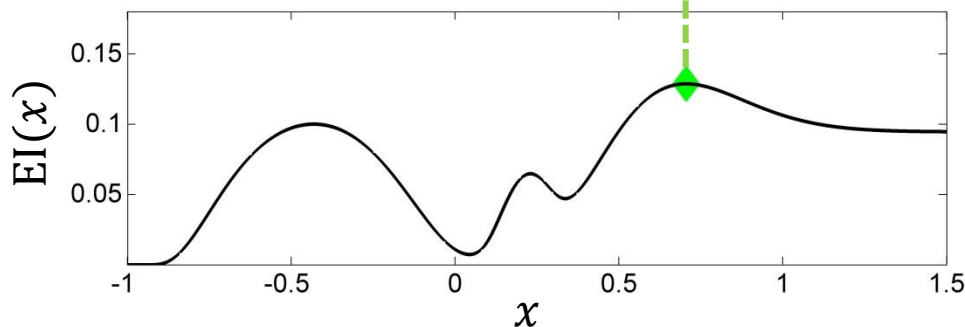
Illustrative example

$$\begin{aligned} & \underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$

$$\begin{matrix} y^1 \\ y^2 \\ y^3 \\ f \end{matrix} \begin{bmatrix} 1.15 \\ 0.18 \\ 1.02 \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$



Next sample



Construct probability distribution on the unknown function value

$$p(f|\mathbf{D}^{1:n}) = N(\mu(x|\mathbf{D}^{1:n}), \sigma^2(x|\mathbf{D}^{1:n}))$$

$$\mu(x|\mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^4 = \arg \max_x EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^4 = 0.70$$

Query the function value

$$y^4 = f(\mathbf{x}^4) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^4 = 1.76$$

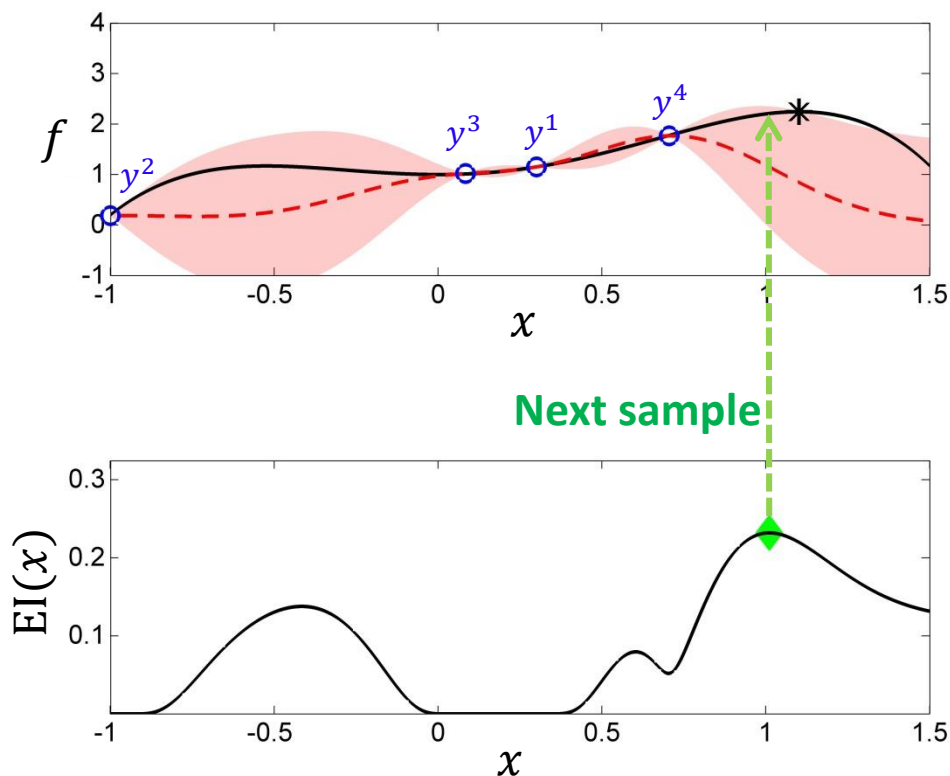
Bayesian Optimization

Illustrative example

$$\begin{aligned} & \underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$

$$\begin{matrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ f \end{matrix} \begin{bmatrix} 1.15 \\ 0.18 \\ 1.02 \\ 1.76 \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$

$n = 4$



Construct probability distribution on the unknown function value

$$p(f|\mathbf{D}^{1:n}) = N(\mu(x|\mathbf{D}^{1:n}), \sigma^2(x|\mathbf{D}^{1:n}))$$

$$\mu(x|\mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^5 = \arg \max_x \text{EI}(x) \triangleq \text{E}[\max\{0, f - f^{\max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^5 = 1.01$$

Query the function value

$$y^5 = f(\mathbf{x}^5) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^5 = 2.19$$

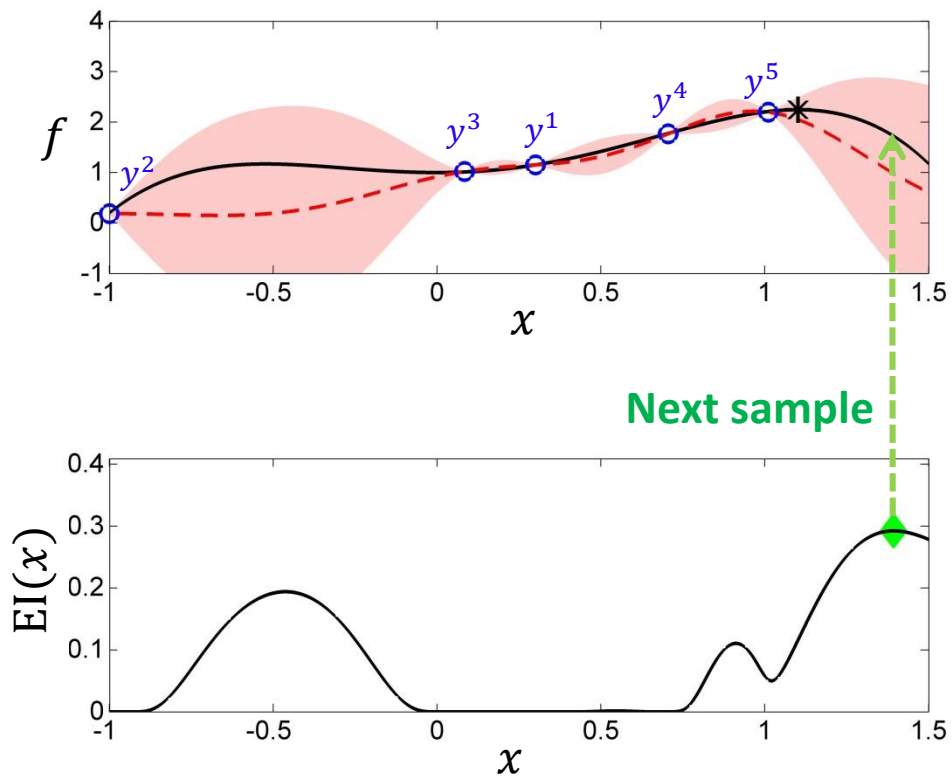
Bayesian Optimization

Illustrative example

$$\underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

$$\text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2)$$

$$\begin{matrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ f \end{matrix} \begin{bmatrix} 1.15 \\ 0.18 \\ 1.02 \\ 1.76 \\ 2.19 \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$



Construct probability distribution on the unknown function value

$$p(f|\mathbf{D}^{1:n}) = N(\mu(x|\mathbf{D}^{1:n}), \sigma^2(x|\mathbf{D}^{1:n}))$$

$$\mu(x|\mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^6 = \arg \max_x \text{EI}(x) \triangleq \text{E}[\max\{0, f - f^{max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^6 = 1.39$$

Query the function value

$$y^6 = f(\mathbf{x}^6) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

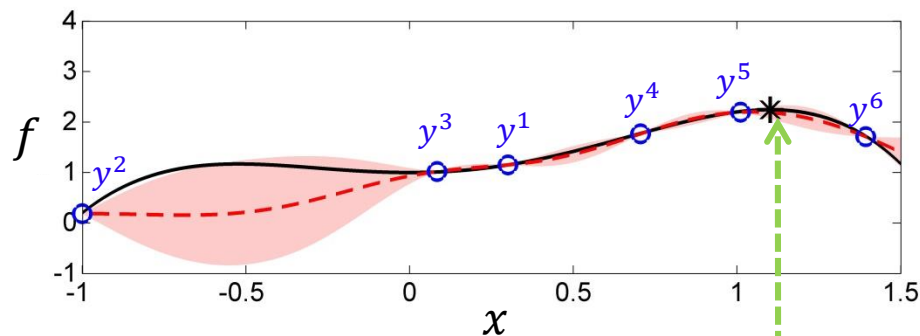
$$y^6 = 1.70$$

Bayesian Optimization

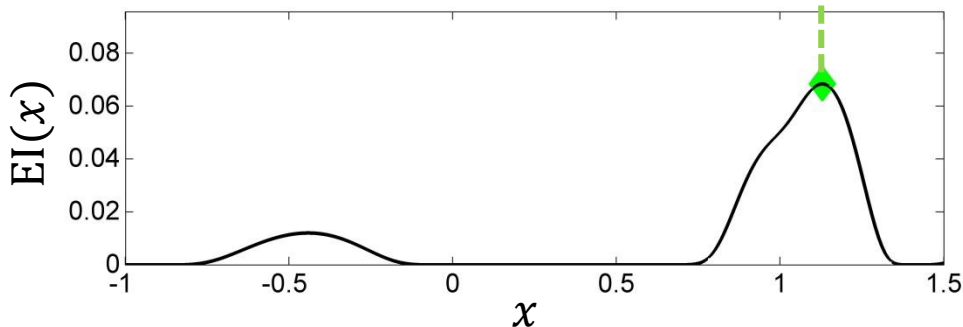
Illustrative example

$$\begin{aligned} & \underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$



Next sample



Construct probability distribution on the unknown function value

$$p(f|\mathbf{D}^{1:n}) = N(\mu(x|\mathbf{D}^{1:n}), \sigma^2(x|\mathbf{D}^{1:n}))$$

$$\mu(x|\mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^7 = \arg \max_x EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^7 = 1.13$$

Query the function value

$$y^7 = f(\mathbf{x}^7) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

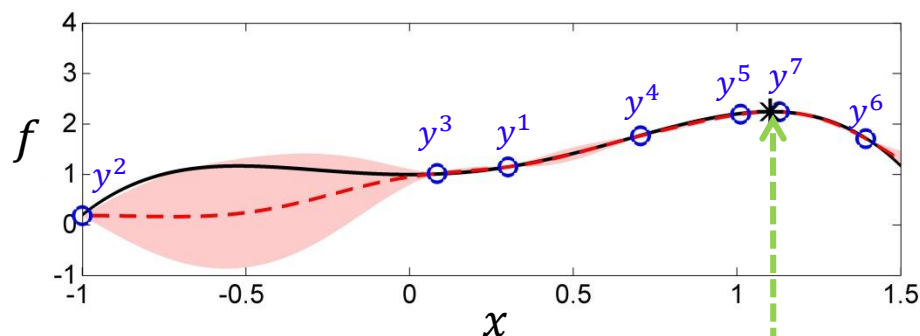
$$y^7 = 2.24$$

Bayesian Optimization

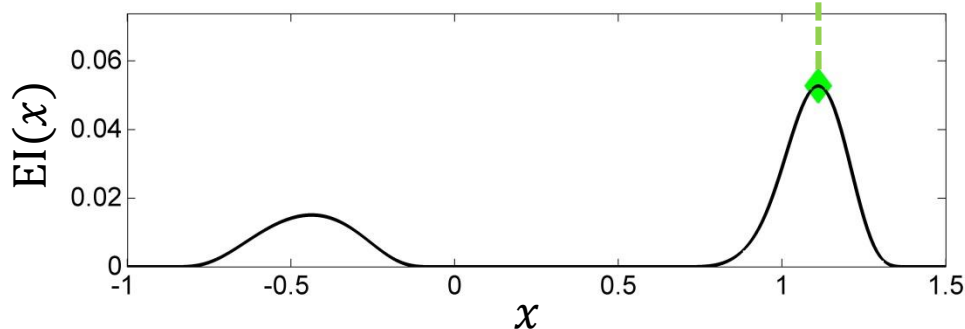
Illustrative example

$$\begin{aligned} & \underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon \\ & \text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2) \end{aligned}$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ y^7 \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$



Next sample



Construct probability distribution on the unknown function value

$$p(f|\mathbf{D}^{1:n}) = N(\mu(x|\mathbf{D}^{1:n}), \sigma^2(x|\mathbf{D}^{1:n}))$$

$$\mu(x|\mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^8 = \arg \max_x \text{EI}(x) \triangleq \text{E}[\max\{0, f - f^{\max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^8 = 1.11$$

Query the function value

$$y^8 = f(\mathbf{x}^8) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^8 = 2.24$$

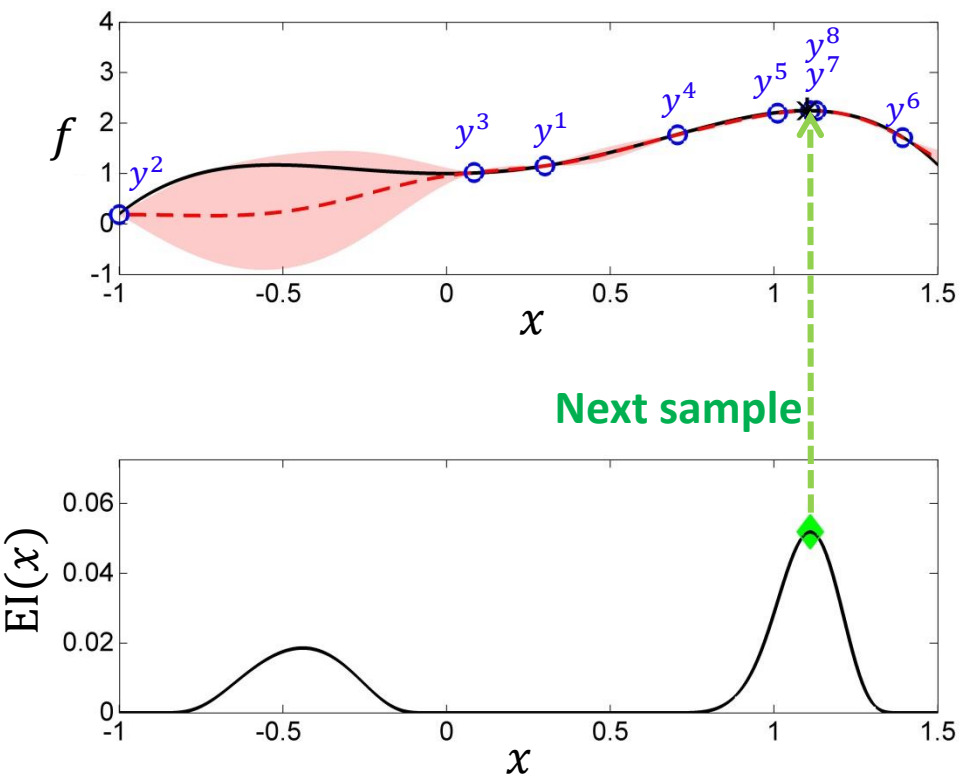
Bayesian Optimization

Illustrative example

$$\underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

$$\text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2)$$

$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ y^7 \\ y^8 \\ f \end{bmatrix} \sim N \left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \right)$$



Construct probability distribution on the unknown function value

$$p(f|\mathbf{D}^{1:n}) = N(\mu(x|\mathbf{D}^{1:n}), \sigma^2(x|\mathbf{D}^{1:n}))$$

$$\mu(x|\mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x|\mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^9 = \arg \max_x EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^9 = 1.11$$

Query the function value

$$y^9 = f(\mathbf{x}^9) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

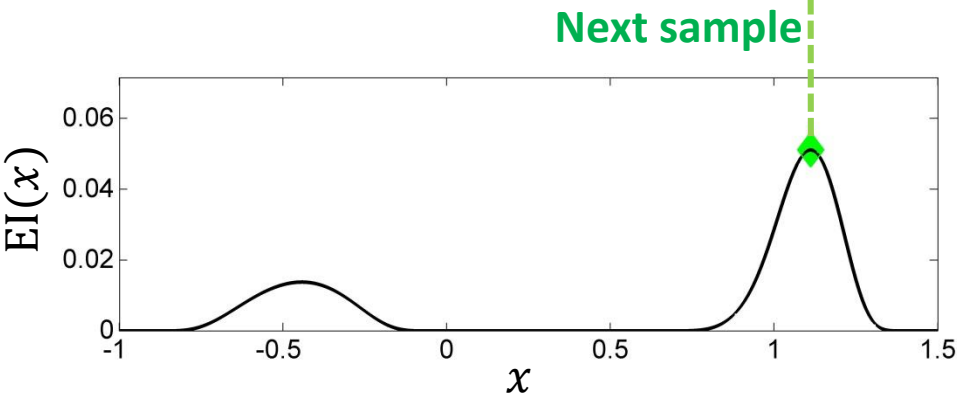
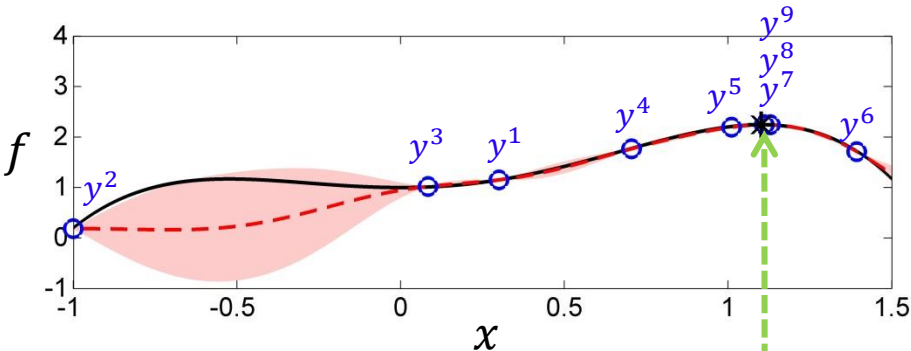
$$y^9 = 2.24$$

Bayesian Optimization

Illustrative example

$$\underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

$$\text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2)$$



$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ y^7 \\ y^8 \\ y^9 \\ f \end{bmatrix} \sim N \left(\begin{bmatrix} \mathbf{0}, \\ \begin{bmatrix} \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{k} \\ \mathbf{k}^T & k(x, x) \end{bmatrix} \end{bmatrix} \right)$$

Construct probability distribution on the unknown function value

$$p(f | \mathbf{D}^{1:n}) = N(\mu(x | \mathbf{D}^{1:n}), \sigma^2(x | \mathbf{D}^{1:n}))$$

$$\mu(x | \mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x | \mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^{10} = \arg \max_z EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathbf{D}^{1:n}]$$

$$\mathbf{x}^{10} = 1.11$$

Query the function value

$$y^{10} = f(\mathbf{x}^{10}) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

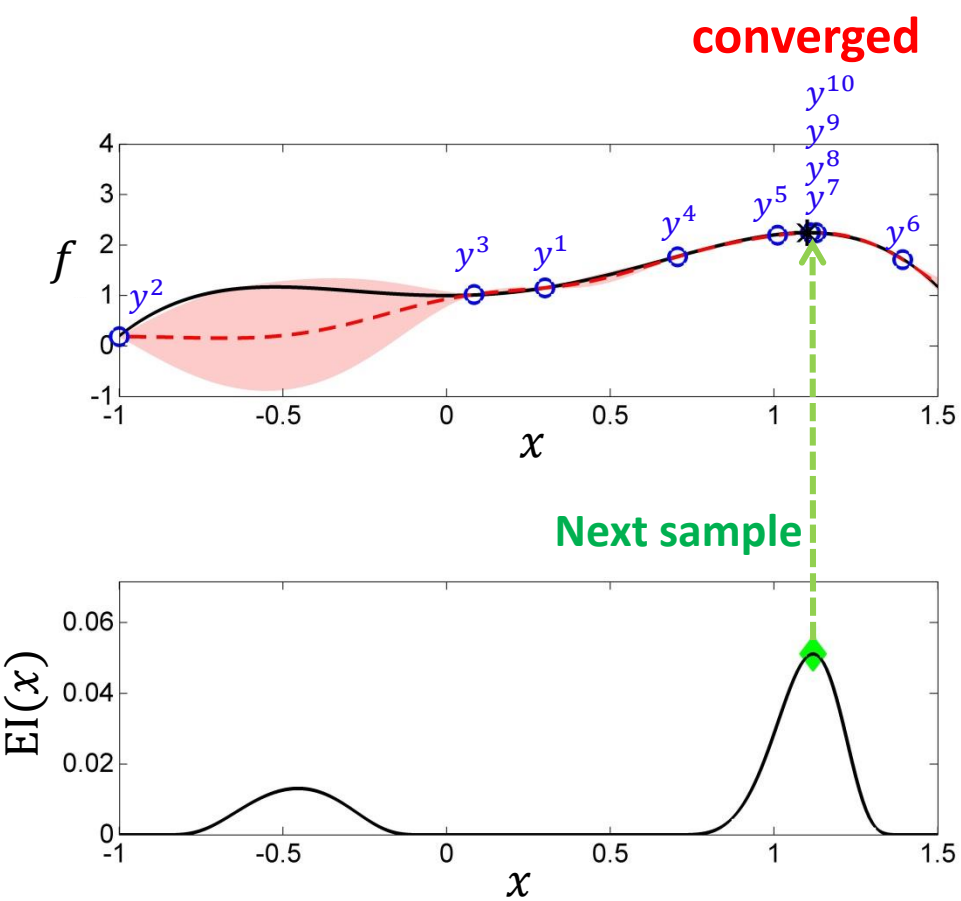
$$y^{10} = 2.24$$

Bayesian Optimization

Illustrative example

$$\underset{x}{\text{maximize}} \quad f(x) = -1.3x^4 + 1x^3 + 1.5x^2 + 1 + \epsilon$$

$$\text{subject to} \quad -1 \leq x \leq 1.5 \quad \epsilon \sim N(0, 0.01^2)$$



$$\begin{bmatrix} y^1 \\ y^2 \\ y^3 \\ y^4 \\ y^5 \\ y^6 \\ y^7 \\ y^8 \\ y^9 \\ y^{10} \\ f \end{bmatrix} \sim N \left(\begin{bmatrix} \mathbf{0} \\ \mathbf{K} + \sigma_\epsilon^2 \mathbf{I} \end{bmatrix}, \begin{bmatrix} \mathbf{k} \\ k(x, x) \end{bmatrix} \right)$$

Construct probability distribution on the unknown function value

$$p(f | \mathbf{D}^{1:n}) = N(\mu(x | \mathbf{D}^{1:n}), \sigma^2(x | \mathbf{D}^{1:n}))$$

$$\mu(x | \mathbf{D}^{1:n}) = \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{y}^{1:n}$$

$$\sigma^2(x | \mathbf{D}^{1:n}) = k(x, x) - \mathbf{k}^T (\mathbf{K} + \sigma_\epsilon^2 \mathbf{I})^{-1} \mathbf{k}$$

Select the next trial action that maximizes

$$\mathbf{x}^{11} = \arg \max_x EI(x) \triangleq E[\max\{0, f - f^{max}\} | \mathbf{D}^{1:n}]$$

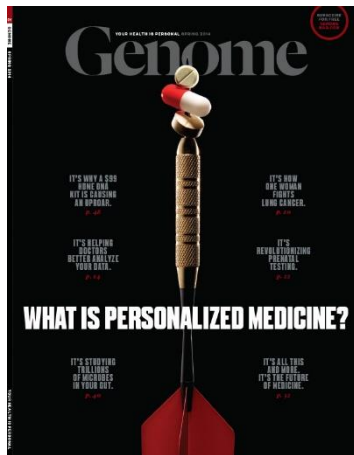
$$\mathbf{x}^{11} = 1.11$$

Query the function value

$$y^{11} = f(\mathbf{x}^{11}) + \epsilon, \epsilon \sim N(0, 0.01^2)$$

$$y^{11} = 2.24$$

Contextual Bayesian Optimization



Suggested for you

Facebook partners with Uber for ride-hailing service via Messenger
Reuters - 1 hour ago
Facebook Inc said on Wednesday it is testing a service that will allow users of its Messenger app, without leaving a conversation or downloading the ride-hailing app.

Eye on safety, California sets rules for self-driving cars
The Seattle Times - 1 hour ago
FILE - In this May 13, 2015, file photo, Google's new self-driving prototype car is presented at Google campus in Mountain View, Calif.

The Apple iPhone 5s got a massive price cut, but is it still worth it?
Firstpost - 1 hour ago
Apple launched the iPhone 5s in India back in November 2013. Two years later, its price is Rs. 21,499 officially for the 16GB version, does the iPhone 5s still make a good deal?

Philips Hue: Just Kidding About Blocking Third-Party Bulbs
PC Magazine - 10 hours ago
You win, internet. Following user backlash, Philips has decided not to block third-party lighting platform after all.

[More Technology stories](#)

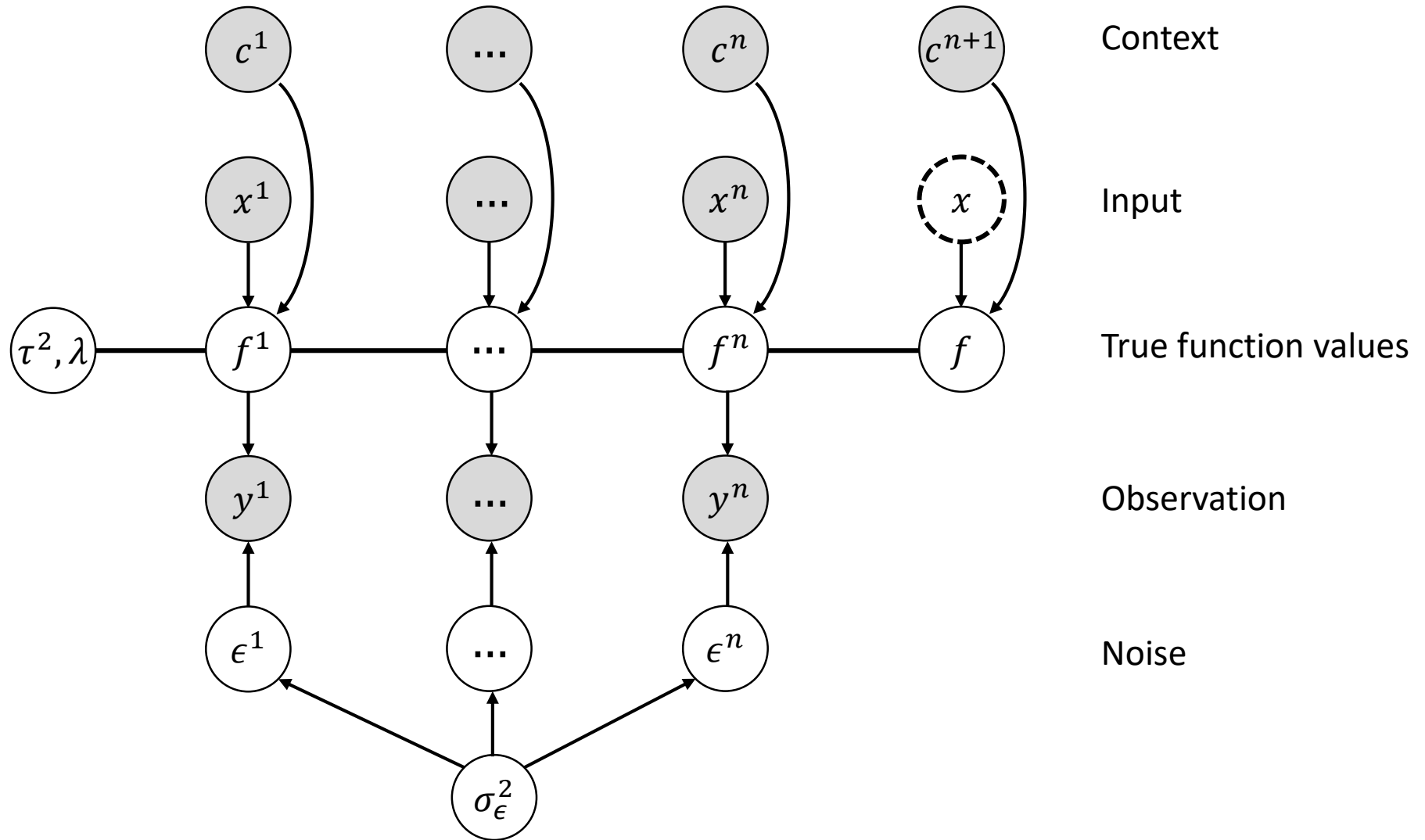


Customers Who Bought This Item Also Bought

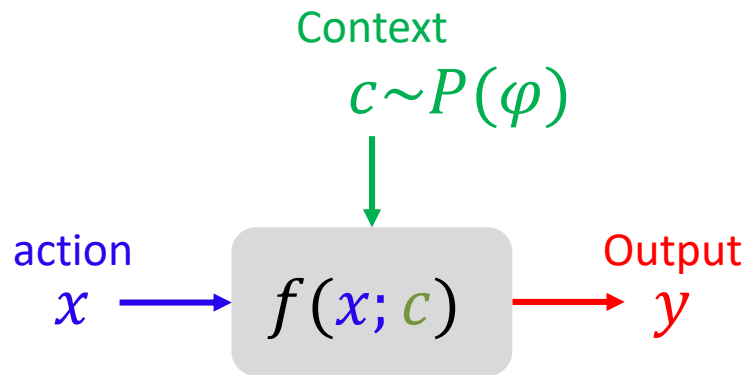
Page 1 of 12

- Finance** : Portfolio optimization under unknown return profiles given varying economic condition
- Health care** : Choosing the best treatment among alternatives given different patient information
- Internet shopping** : Choosing the optimum price (sales v.s. profits) given varying season
- Experiment design** : Sequential experimental design given varying environmental condition

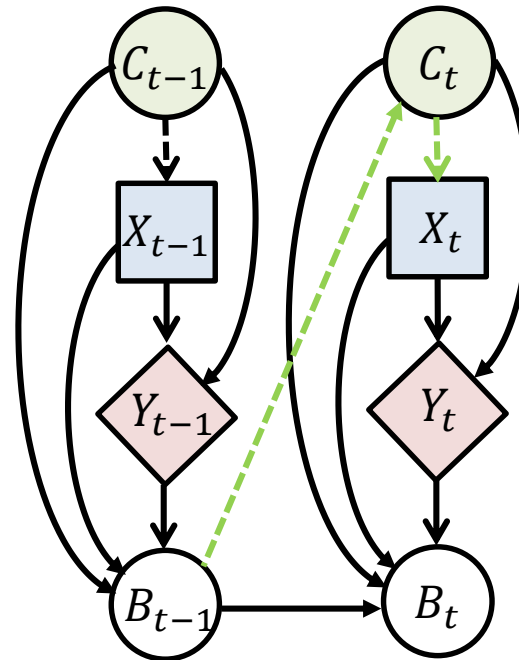
Contextual Bayesian Optimization



Contextual Bayesian Optimization



MDP over belief state



$B_t(f)$: Belief state about unknown function f at t

- Policy π maps all the history to new action:

$$\pi: [(c_1, x_1, y_1), (c_2, x_2, y_2), \dots, (c_{t-1}, x_{t-1}, y_{t-1}), c_t] \rightarrow x_t$$

- Find the optimal policy π^* that maximizes $E[\sum_{t=1}^T y_t]$ or $E[y_T]$

$$x^* = \pi^*(c) = \operatorname{argmax}_x f(x; c)$$