Multi-Armed Bandits

CSE 312 23Su
Lecture 22

https://xkcd.com/1445/

The Problem

There are K slot machines (“bandits” with “arms”).

The Problem

There are K slot machines (“bandits” with “arms”).

You pull arms T times, where the arm you pull at time t $a_t \in \{1, \dots, K\}$ will return a **random reward** to you.

The Problem

Your goal? Win as much money as possible! 🤑 That is, maximize your total expected reward after T pulls of an arm of the K slot machines.

The Problem

Your goal? Win as much money as possible! 🤑 That is, maximize your total expected reward after T pulls of an arm of the K slot machines.

Question: How do you know which arm to pull (based on information you have – the past) to achieve your goal?

The Problem

Your goal? Win as much money as possible! 🤑 That is, maximize your total expected reward after T pulls of an arm of the K slot machines.

Question: How do you know which arm to pull (based on information you have – the past) to achieve your goal?

Assumptions:

1. Rewards from each arm are independent.

2. The reward distribution of an arm does not change over time.

If you knew the reward distribution…

You have the below K=3 slot machines, whose distribution of rewards you know. What strategy are you going to use to maximize your total (expected) reward?

$Poi(\lambda = 1.36)$

$Bin(n = 10, p = 0.4)$

$\mathcal{N}(\mu = -1, \sigma^2 = 4)$

If you knew the reward distribution…

You have the below K=3 slot machines, whose distribution of rewards you know. What strategy are you going to use to maximize your total (expected) reward?

Expected reward per pull: $1.36, 4, -1$. Pull arm 2 T times!

$Poi(\lambda = 1.36)$

$Bin(n = 10, p = 0.4)$

$\mathcal{N}(\mu = -1, \sigma^2 = 4)$

Neat! Problem solved!

Aren't you glad you learnt to calculate expectations so early in this course?

Oh wait. Why would we know the reward distributions of slot machines?

We don't.

We need to… *estimate* all K expectations.

WHILE maximizing the total (expected) reward.

Keys to Problem Setup

Exploitation vs. Exploration

That famous tradeoff…

Exploitation: Pulling arms that we know are “good” based on reward history.

Exploration: Pulling other arms in case they could be “good” or “better”.

Bernoulli Bandits

We will simplify the problem ☺

We have K arms, where the reward from each arm is either 1 or 0.

For arm $a \in \{1, \dots, K\}$, the reward is drawn from $Ber(p_a)$.

Then, the expected reward from arm a is p_a.

Regret

The difference between the best possible total (expected) reward and the actual reward from T pulls.

$$p^* = \arg\max_{i \in \{1,\dots K\}} p_i$$

$$Regret(T) = T \cdot p^* - Reward(T)$$

Regret

The difference between the best possible total (expected) reward and the actual reward from T pulls.

$p^* = \arg\max_{i \in \{1,\dots K\}} p_i$

$Regret(T) = T \cdot p^* - Reward(T)$

$AvgRegret(T) = p^* - \frac{Reward(T)}{T}$

Minimizing average regret is equivalent to maximizing reward!

Framework

Algorithm 1 (Bernoulli) Bandit Framework

1: Have K arms, where pulling arm $i \in \{1, \ldots, K\}$ gives $Ber(p_i)$ reward ▷ p_i's all unknown.
2: **for** $t = 1, \ldots, T$ **do**
3: At time t, pull arm $a_t \in \{1, \ldots, K\}$. ▷ How do we do decide which arm?
4: Receive reward $r_t \sim Ber(p_{a_t})$. ▷ Reward is either 1 or 0.

How do we choose which arm?

Strategy 1: Naïve Greedy

Pull each arm M times.

Then, for the remaining pulls, pull the “best” arm.

Strategy 1: Naïve Greedy

Pull each arm M times.

Then, for the remaining pulls, pull the “best” arm.

Algorithm 2 Greedy (Naive) Strategy for Bernoulli Bandits

1: Choose a number of times M to pull each arm initially, with $KM \leq T$.
2: **for** $i = 1, 2, \ldots, K$ **do**
3: Pull arm i M times, observing iid rewards $r_{i1}, \ldots, r_{iM} \sim Ber(p_i)$.
4: Estimate $\hat{p}_i = \frac{\sum_{j=1}^{M} r_{ij}}{M}$.

Strategy 1: Naïve Greedy

Pull each arm M times.

Then, for the remaining pulls, pull the "best" arm.

Algorithm 2 Greedy (Naive) Strategy for Bernoulli Bandits

1: Choose a number of times M to pull each arm initially, with $KM \leq T$.
2: **for** $i = 1, 2, \ldots, K$ **do**
3: Pull arm i M times, observing iid rewards $r_{i1}, \ldots, r_{iM} \sim Ber(p_i)$.
4: Estimate $\hat{p}_i = \frac{\sum_{j=1}^{M} r_{ij}}{M}$.
5: Determine best (empirical) arm $a^* = \arg\max_{i \in \{1,2,\ldots,K\}} \hat{p}_i$. ▷ We could be wrong...
6: **for** $t = KM + 1, KM + 2, \ldots, T$ **do**:
7: Pull arm $a_t = a^*$. ▷ Pull the same arm for the rest of time.
8: Receive reward $r_t \sim Ber(p_{a_t})$.

Problems?

We may not get an accurate idea from our first M pulls.

What happens when we increase M?

We did all exploration, and then all exploitation.

Why don’t we blend the two a bit more?

Strategy 2: Epsilon-Greedy

Let's mix in some more exploration.

We pull the arms M times.

For the remaining time:

with some small probability ε, we try an arm that may not be the known best

otherwise, we pull our known best arm

Strategy 2: Epsilon-Greedy

Algorithm 3 ε-Greedy Strategy for Bernoulli Bandits

Choose a number of times M to pull each arm initially, with $KM \leq T$.
for $i = 1, 2, \ldots, K$ **do**
    Pull arm i M times, observing iid rewards $r_{i1}, \ldots, r_{iM} \sim Ber(p_i)$.
    Estimate $\hat{p}_i = \frac{\sum_{j=1}^{M} r_{ij}}{M}$.
for $t = KM + 1, KM + 2, \ldots, T$ **do**:
    if $Ber(\varepsilon) == 1$: **then** ▷ With probability ε, explore.
        Pull arm $a_t = Unif(1, K)$ (discrete). ▷ Choose a uniformly random arm.
    else ▷ With probability $1 - \varepsilon$, exploit.
        Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} \hat{p}_i$. ▷ Choose arm with highest estimated reward.
    Receive reward $r_t \sim Ber(p_{a_t})$.
    Update $\hat{p}_{a_t}$ (using newly observed reward r_t).

Problems?

Is uniform exploration the optimal policy?

Let's say, arm 1 has been pulled 100 times and we estimate the expected reward to be 0.01.

Arm 2 has been pulled 5 times, and we estimate the expected reward to be 0.6.

Do we want to explore these equally?

Strategy 3: Upper Confidence Bound (UCB)

Instead of looking at the probability estimate $\widehat{p_a}$, we want to construct confidence intervals.

For e.g.,

$\widehat{\boldsymbol{p_1}} = \mathbf{0.75}$, confidence interval $[0.75 - 0.10, 0.75 + 0.10] = [0.65, 0.85]$

$\widehat{p_2} = 0.33$, confidence interval $[0.33 - 0.25, 0.33 + 0.25] = [0.08, 0.58]$

$\widehat{p_3} = 0.60$, confidence interval $[0.60 - 0.29, 0.60 + 0.29] = [0.31, \mathbf{0.89}]$

Strategy 3: Upper Confidence Bound (UCB)

Algorithm 4 UCB1 Algorithm (Upper Confidence Bound) for Bernoulli Bandits

1: **for** $i = 1, 2, \ldots, K$ **do**
2: Pull arm i once, observing $r_i \sim Ber(p_i)$.
3: Estimate $\hat{p}_i = r_i/1$. ▷ Each estimate $\hat{p}_i$ will initially either be 1 or 0.

Strategy 3: Upper Confidence Bound (UCB)

Algorithm 4 UCB1 Algorithm (Upper Confidence Bound) for Bernoulli Bandits

1: **for** $i = 1, 2, \ldots, K$ **do**

2: Pull arm i once, observing $r_i \sim Ber(p_i)$.

3: Estimate $\hat{p}_i = r_i/1$. ▷ Each estimate $\hat{p}_i$ will initially either be 1 or 0.

4: **for** $t = K+1, K+2, \ldots, T$ **do**:

5: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} \left(\hat{p}_i + \sqrt{\frac{2\ln(t)}{N_t(i)}} \right)$, where $N_t(i)$ is the number of times arm i was pulled before time t.

Strategy 3: Upper Confidence Bound (UCB)

Algorithm 4 UCB1 Algorithm (Upper Confidence Bound) for Bernoulli Bandits

1: **for** $i = 1, 2, \ldots, K$ **do**
2: Pull arm i once, observing $r_i \sim Ber(p_i)$.
3: Estimate $\hat{p}_i = r_i$. ▷ Each estimate $\hat{p}_i$ will initially either be 1 or 0.
4: **for** $t = K+1, K+2, \ldots, T$ **do**:
5: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} \left(\hat{p}_i + \sqrt{\frac{2\ln(t)}{N_t(i)}} \right)$, where $N_t(i)$ is the number of times arm i was pulled before time t.
6: Receive reward $r_t \sim Ber(p_{a_t})$.
7: Update $N_t(a_t)$ and $\hat{p}_{a_t}$ (using newly observed reward r_t).

Hey, hey, hey! Where did that UCB come from?

Derived from Hoeffding's inequality (an upper bound on the probability of deviation from the expected value for bounded independent random variables)

If interested, theoretical details in "Finite-time analysis of the multiarmed bandit problem" by Auer, Cesa-Bianchi and Fischer.

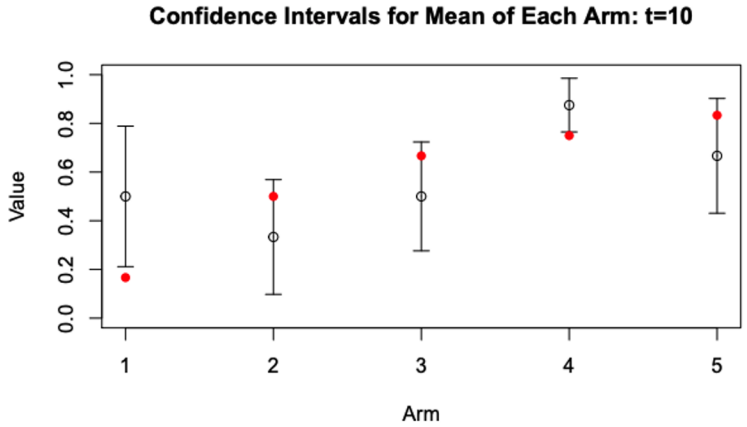
UCB

Exploration engrained into the process.

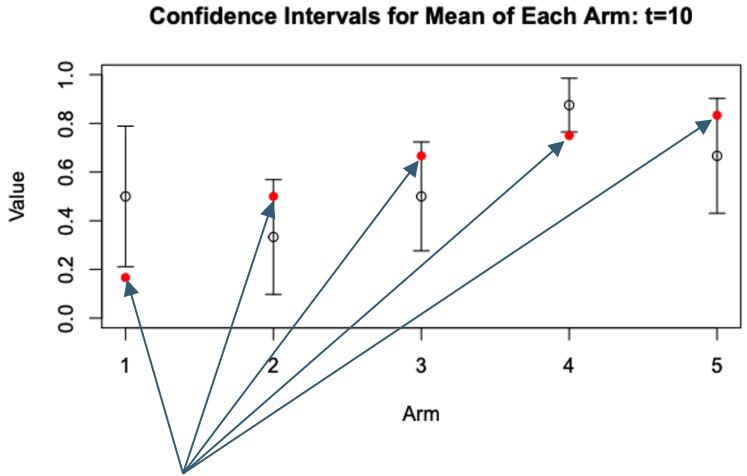
More pulls => narrower confidence interval => Exploit knowledge of “good” (and “bad”) arms

Fewer pulls => larger confidence interval => Opportunity to explore

UCB: Confidence Intervals over Time

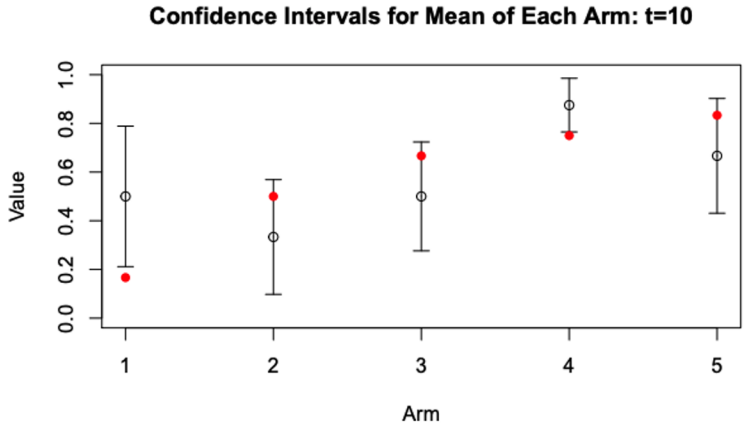


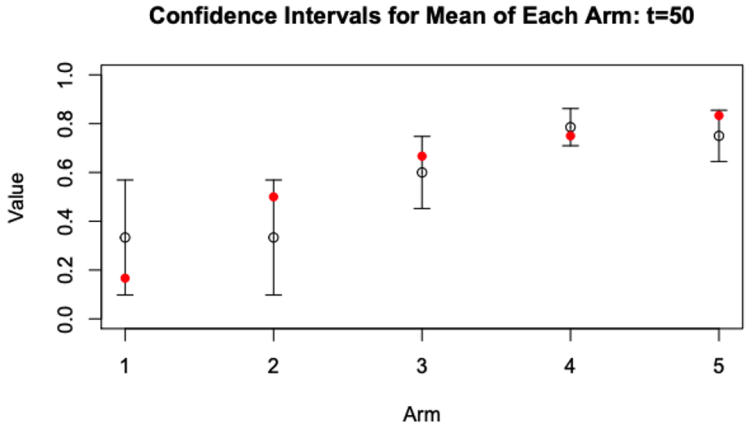
UCB: Confidence Intervals over Time



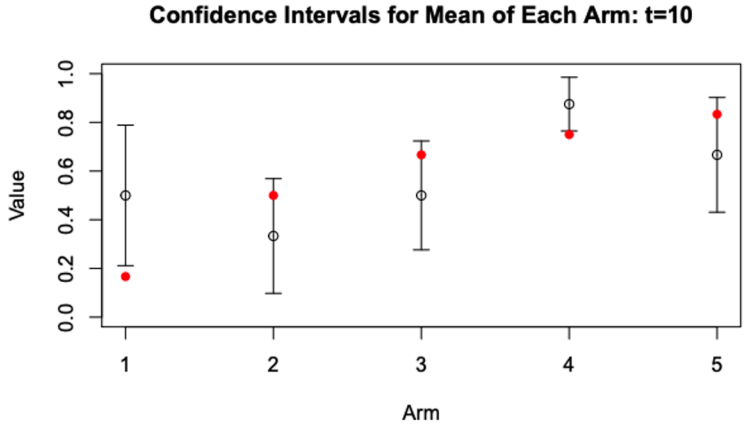
Red Dots: True Means

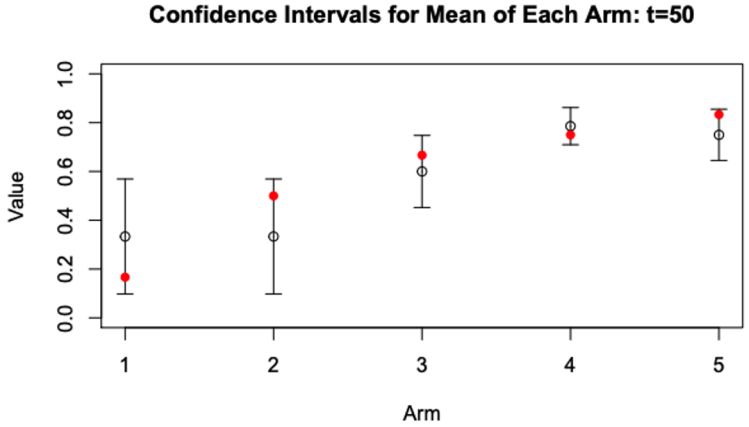
UCB: Confidence Intervals over Time

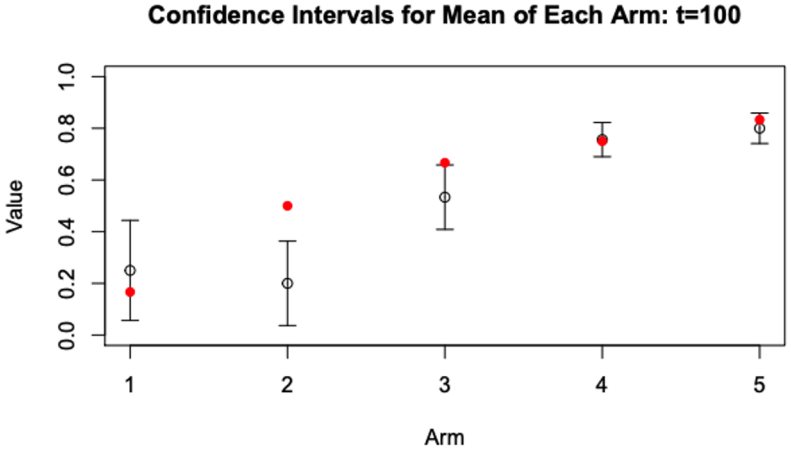




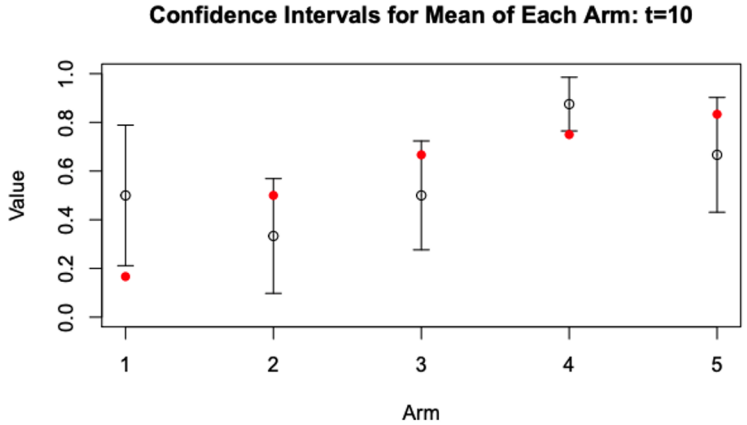
UCB: Confidence Intervals over Time

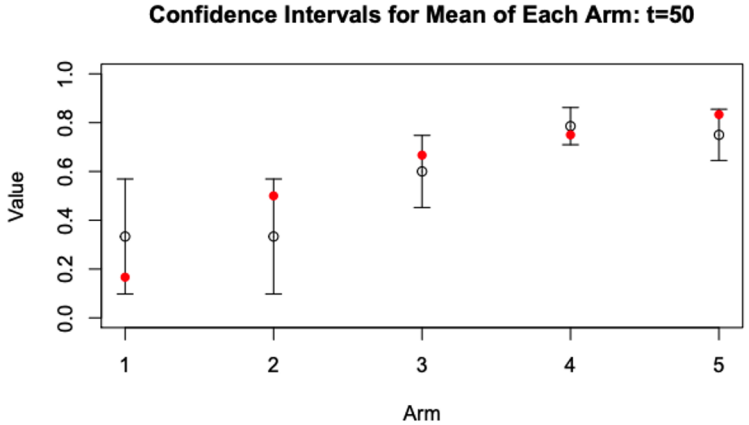


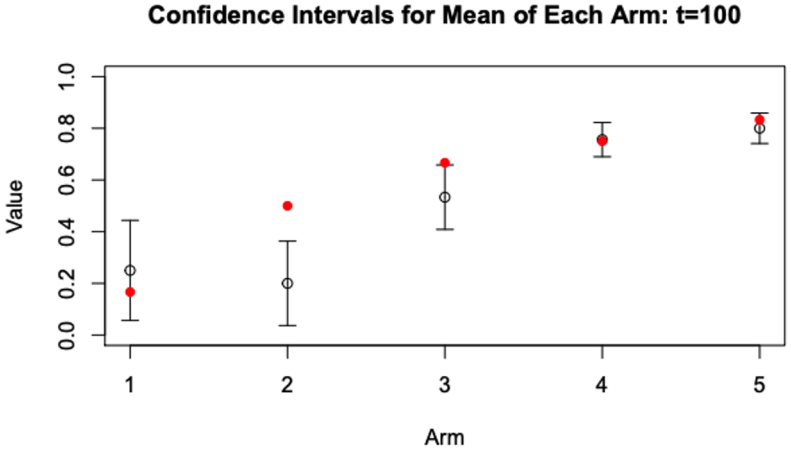


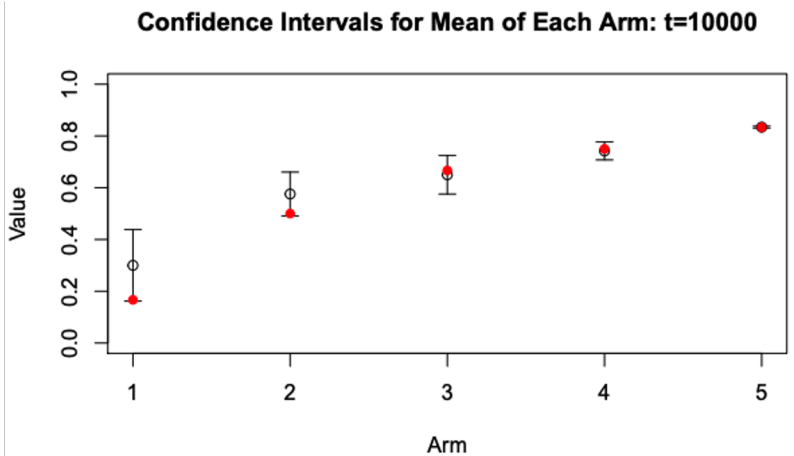


UCB: Confidence Intervals over Time









Strategy 4: Thompson Sampling

Well… we're trying to figure out the parameter of a Bernoulli distribution.

And we've been using MLE to determine that parameter.

But we said earlier we have other estimators.

A Change of Perspective: Let's Get Meta

With MLE, we maximized $\mathbb{P}(\boldsymbol{x}\,;\theta)$.

It makes sense that we might want to maximize $\mathbb{P}(\Theta\,;\boldsymbol{x})$ instead of the opposite.

However, this only works if the parameter is a random variable.

How do we describe the distribution of Θ?

Beta Distribution

Treat the parameter itself as a random variable (uncertainty about its value).

Model the probability distribution for the values that the parameter θ can take.

Beta Distribution

X is the value of p from a Binomial distribution with $\alpha - 1$ successes and $\beta - 1$ failures.

$$\Omega_X = [0, 1]$$

$$X \sim Beta(\alpha, \beta)$$

$$f_X(x) = \begin{cases} \frac{1}{B(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1} & 0 \leq x \leq 1 \\ 0 & otherwise \end{cases}$$

$$Mode[X] = \frac{\alpha-1}{(\alpha-1)+(\beta-1)}$$

Concretely…

If we want to treat the parameter we are estimating as a random variable, we define it as a Beta random variable (for Bernoulli/Binomial priors).

MAB Prior

What prior would you choose for the K slot machines, if you have no information about them?

MAB Prior

What prior would you choose for the K slot machines, if you have no information about them?

We know nothing, so a reasonable prior may be that every parameter value is equally likely.

We saw 0 successes and 0 failures, which describes a $Beta(1,1) = Unif(0,1)$ prior.

So how do we choose which arm to pull?

Earlier, we determined the MLE and pulled the arm with the highest estimated parameter.

We may consider doing something similar and finding the MAP estimator using our Beta prior.

However, it turns out that MAP and MLE produce identical estimators for a uniform distribution.

Sampling!

Instead of finding an estimator for the Bernoulli parameter, we are going to **sample** from the distribution.

This sampling adds exploration to our algorithm.

The Algorithm

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
    For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
    Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
    Receive reward $r_t \sim Ber(p_{a_t})$.
    if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
    else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

E.g.

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

1: For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
2: **for** $t = 1, 2, \ldots, T$ **do**:
3: For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
4: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
5: Receive reward $r_t \sim Ber(p_{a_t})$.
6: **if** $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
7: **else if** $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5			
2	0.2			
3	0.9			

Time (t)	Arm Pulled (a_t)	Reward (r_t)

Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	
2	0.2	1	1	
3	0.9	1	1	

Time (t)	Arm Pulled (a_t)	Reward (r_t)
1		

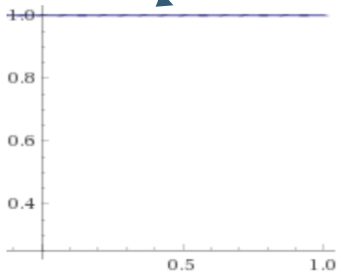
Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.43
2	0.2	1	1	
3	0.9	1	1	

Sample from Beta(1,1) density →



Time (t)	Arm Pulled (a_t)	Reward (r_t)
1		

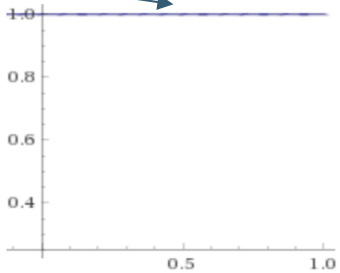
Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

1: For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
2: **for** $t = 1, 2, \ldots, T$ **do**:
3: For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
4: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
5: Receive reward $r_t \sim Ber(p_{a_t})$.
6: **if** $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
7: **else if** $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.43
2	0.2	1	1	0.75
3	0.9	1	1	

Sample from Beta(1,1) density →



Time (t)	Arm Pulled (a_t)	Reward (r_t)
1		

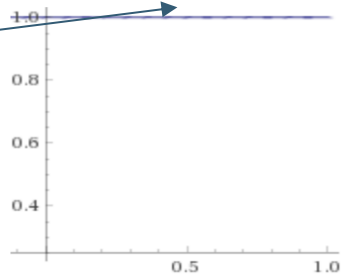
Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.43
2	0.2	1	1	0.75
3	0.9	1	1	0.11

Sample from Beta(1,1) density →



Time (t)	Arm Pulled (a_t)	Reward (r_t)
1		

Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.43
2	0.2	1	1	0.75
3	0.9	1	1	0.11

Time (t)	Arm Pulled (a_t)	Reward (r_t)
1	2	

Choose arm with highest sample!

Thompson Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

1: For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
2: **for** $t = 1, 2, \ldots, T$ **do**:
3: For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
4: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
5: Receive reward $r_t \sim Ber(p_{a_t})$.
6: **if** $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
7: **else if** $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.43
2	0.2	1	1	0.75
3	0.9	1	1	0.11

Time (t)	Arm Pulled (a_t)	Reward (r_t)
1	2	0

Observe reward 1 with probability 0.2 and 0 with probability 0.8.

Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	
2	0.2	1	2	
3	0.9	1	1	

Time (t)	Arm Pulled (a_t)	Reward (r_t)
1	2	0

Add a count of 1 to the failures :(.

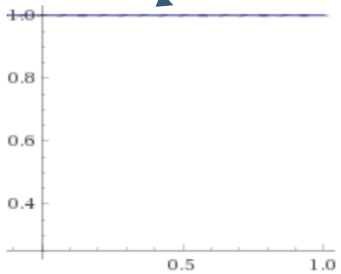
Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.52
2	0.2	1	2	
3	0.9	1	1	

Sample from Beta(1,1) density →



Time (t)	Arm Pulled (a_t)	Reward (r_t)
2		

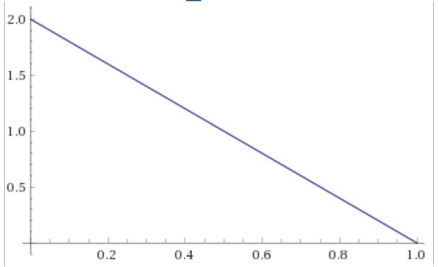
Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.52
2	0.2	1	2	0.05
3	0.9	1	1	

Sample from Beta(1,2) density →



Time (t)	Arm Pulled (a_t)	Reward (r_t)
2		

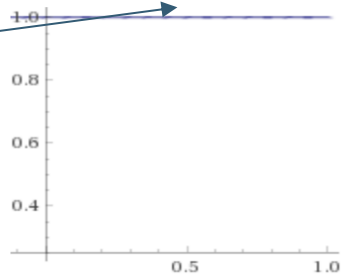
Thompson Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

1: For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
2: **for** $t = 1, 2, \ldots, T$ **do**:
3: For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
4: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
5: Receive reward $r_t \sim Ber(p_{a_t})$.
6: **if** $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
7: **else if** $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.52
2	0.2	1	2	0.05
3	0.9	1	1	0.67

Sample from Beta(1,1) density →



Time (t)	Arm Pulled (a_t)	Reward (r_t)
2		

Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.52
2	0.2	1	2	0.05
3	0.9	1	1	0.67

Time (t)	Arm Pulled (a_t)	Reward (r_t)
2	3	

Choose arm with highest sample!

Thompson Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

1: For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
2: **for** $t = 1, 2, \ldots, T$ **do**:
3: For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
4: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
5: Receive reward $r_t \sim Ber(p_{a_t})$.
6: **if** $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
7: **else if** $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.52
2	0.2	1	2	0.05
3	0.9	1	1	0.67

Time (t)	Arm Pulled (a_t)	Reward (r_t)
2	3	1

Observe reward 1 with probability 0.9 and 0 with probability 0.1.

Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	
2	0.2	1	2	
3	0.9	2	1	

Time (t)	Arm Pulled (a_t)	Reward (r_t)
2	3	1

Add a count of 1 to the successes :).

Thompson Sampling Example

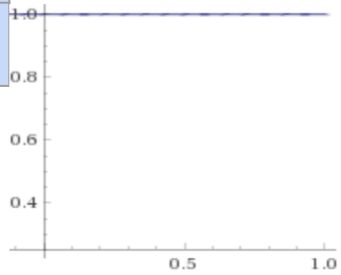
Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

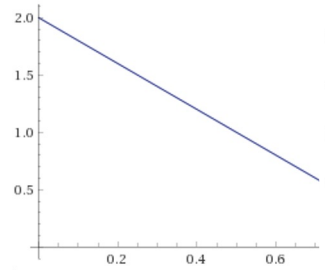
1: For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
2: **for** $t = 1, 2, \ldots, T$ **do**:
3: For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
4: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
5: Receive reward $r_t \sim Ber(p_{a_t})$.
6: **if** $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
7: **else if** $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

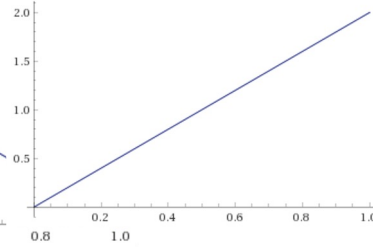
Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.44
2	0.2	1	2	0.27
3	0.9	2	1	0.86

Time (t)	Arm Pulled (a_t)	Reward (r_t)
3		

Sample from each arm's Beta distribution →







Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.44
2	0.2	1	2	0.27
3	0.9	2	1	0.86

Time (t)	Arm Pulled (a_t)	Reward (r_t)
3	3	0

Observe reward 1 with probability 0.9 and 0 with probability 0.1.

Thompson Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.44
2	0.2	1	2	0.27
3	0.9	2	2	0.86

Time (t)	Arm Pulled (a_t)	Reward (r_t)
3	3	0

Add a count of 1 to the failures :(.

Thompson Sampling Example

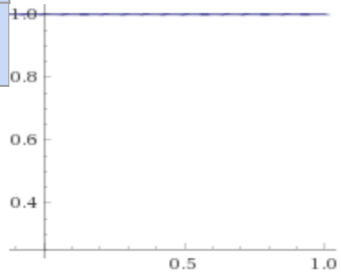
Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

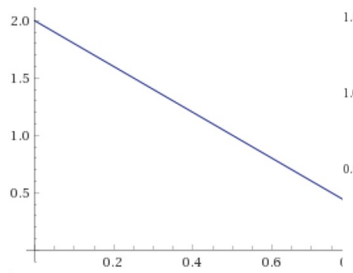
1: For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
2: **for** $t = 1, 2, \ldots, T$ **do**:
3: For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
4: Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
5: Receive reward $r_t \sim Ber(p_{a_t})$.
6: **if** $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
7: **else if** $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

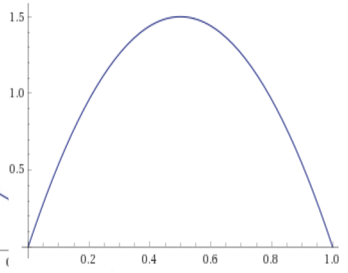
Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.63
2	0.2	1	2	0.15
3	0.9	2	2	0.44

Time (t)	Arm Pulled (a_t)	Reward (r_t)
4		

Sample from each arm's Beta distribution →







Thompson Sampling Example

Algorithm 5 Thompson Sampling Algorithm for Beta-Bernoulli Bandits

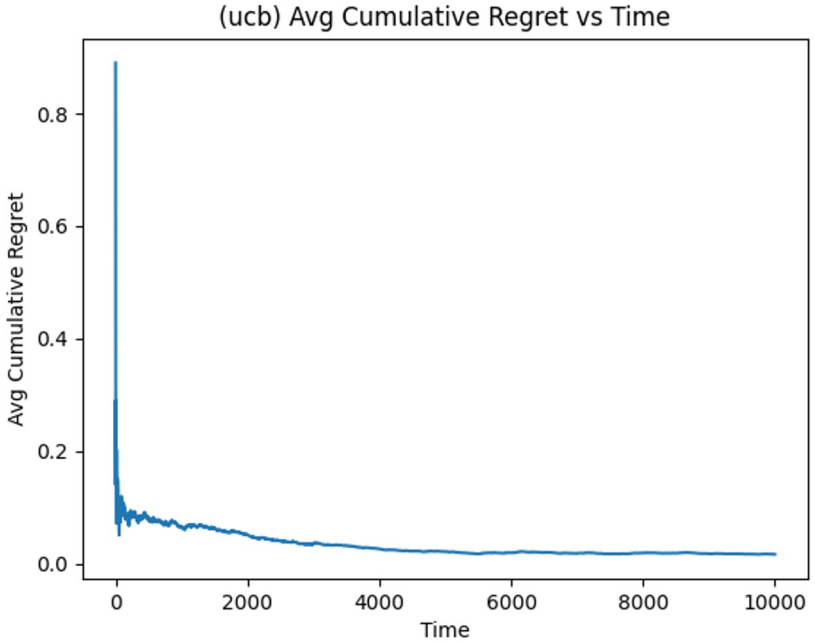
For each arm $i \in \{1, \ldots, K\}$, initialize $\alpha_i = \beta_i = 1$. ▷ Set $Beta(\alpha_i, \beta_i)$ prior for each p_i.
for $t = 1, 2, \ldots, T$ **do**:
  For each arm i, get sample $s_{i,t} \sim Beta(\alpha_i, \beta_i)$. ▷ Each is a float in $[0, 1]$.
  Pull arm $a_t = \arg\max_{i \in \{1,2,\ldots,K\}} s_{i,t}$. ▷ This "bakes in" exploration!
  Receive reward $r_t \sim Ber(p_{a_t})$.
  if $r_t == 1$ **then** $\alpha_{a_t} \leftarrow \alpha_{a_t} + 1$. ▷ Increment number of "successes".
  else if $r_t == 0$ **then** $\beta_{a_t} \leftarrow \beta_{a_t} + 1$. ▷ Increment number of "failures".

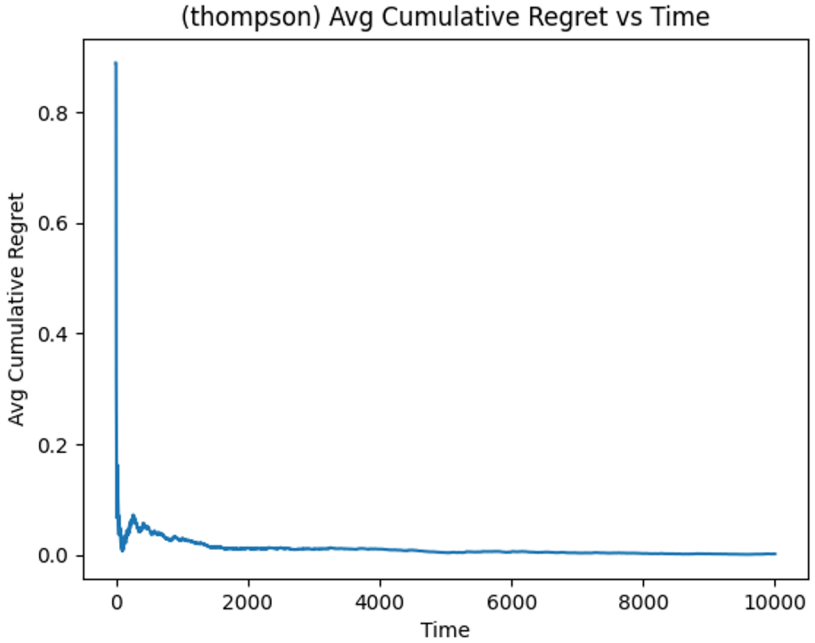
Arm (i)	True p_i	α_i	β_i	$s_{i,t}$
1	0.5	1	1	0.63
2	0.2	1	2	0.15
3	0.9	2	2	0.44

Time (t)	Arm Pulled (a_t)	Reward (r_t)
4		

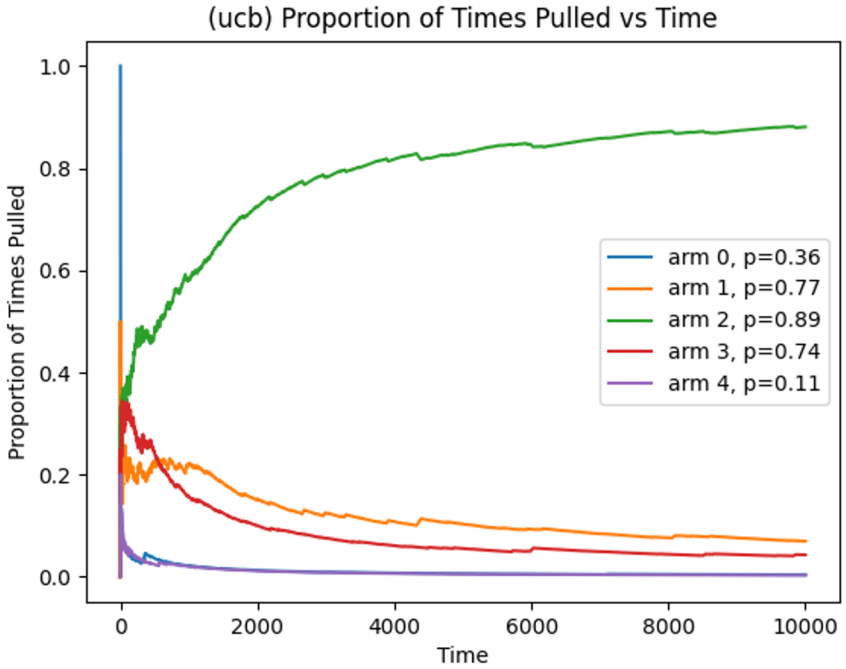
And so on!!! Notice how we explore because there's some chance the "best" arm will have a lower sample occasionally and let other arms win!

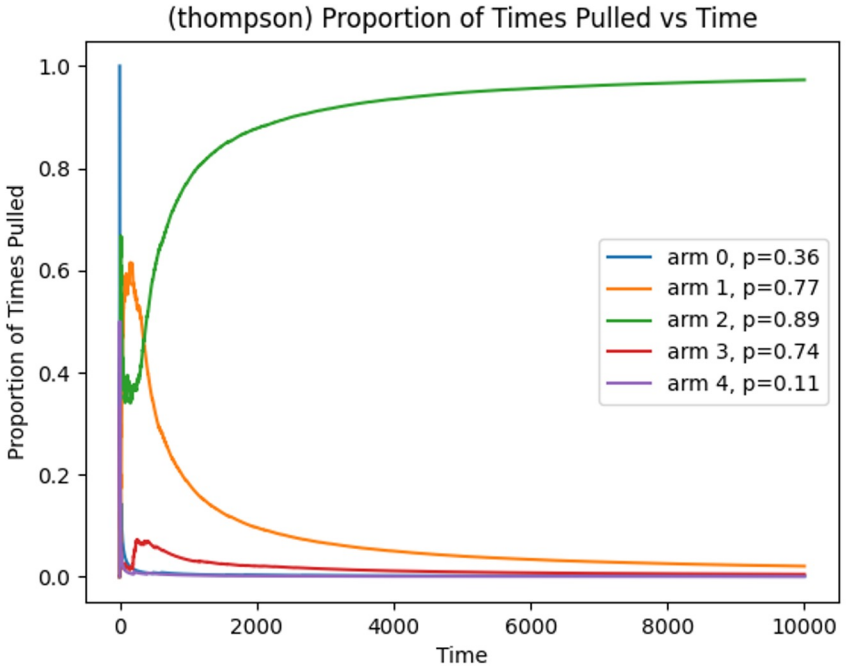
UCB Vs Thompson Sampling: Avg Regret over Time





UCB Vs Thompson Sampling: Proportion of Times Pulled





Why do we care?

Hypothesis Testing

You’re at a tech company. You want to experiment with releasing a new feature. How do we know if our users prefer it or not?

Hypothesis Testing

You're at a tech company. You want to experiment with releasing a new feature. How do we know if our users prefer it or not?

99% of users see the current version; 1% see the new version.

Perform a hypothesis test to see if the new feature performs better for some metric (e.g., click-through rate).

Hypothesis Testing

How about adaptively assigning subjects to each group?

Arm 1: Current feature

Arm 2: New feature

Dynamically increase visitor allocation to better performing versions.

Can have many arms!

Other Uses~

A classic reinforcement learning problem.

In medicine: compare different experimental treatments & minimize harm to patients

In networks: adaptive routing, to minimize delays in the network

(For the curious) MAP Estimation

Typical process:

1. Choose a prior.
2. Observe sample
3. $\pi_{\Theta}(\theta\,;\boldsymbol{x}) = \frac{\mathcal{L}(\boldsymbol{x}\,;\theta)\pi_{\Theta}(\theta)}{\mathbb{P}(\boldsymbol{x})} \propto \mathcal{L}(\boldsymbol{x}\,;\theta)\pi_{\Theta}(\theta)$
4. $\hat{\theta}_{MAP} = \arg\max_{\theta} \mathcal{L}(\boldsymbol{x}\,;\theta)\pi_{\Theta}(\theta)$