Lecture 8: A Deep Dive Into Linear Bandits

ID 6002W: Online and Reinforcement Learning

Web-enabled M.Tech. program, IIT Madras

Suryanarayana Sankagiri

3rd June, 2026

Outline of Today's Class

- Understanding the mathematics of the **linear bandits model**
- Understanding the UCB algorithm in this model
- Understanding the real-world interpretation and flexibility of the model

Philosophy for today:

let us understand **what** changes we are making to basic bandit model,

then we will revisit **why** we are doing so

Recap: Basic Bandit Formalism

- We have a fixed set of K arms
- At each time t, we can pull any one arm A_t
- Pulling an arm gives us a random reward X_t
- Assumption: the rewards from any fixed arm are i.i.d.
- Reward distribution of arm i is ν_i (fixed, but unknown)
- Mean reward of arm i is μ_i (fixed, but unknown)

Recap: Basic Bandit Formalism

- The best arm is $i^* = \arg\max_{i \in [K]} \mu_i$
- The goal is to minimise the cumulative regret over T rounds:

$$R(T) = \sum_{t=1}^{T} \left(\mu^* - \mathbb{E}[\mu_{A_t}] \right)$$

- Here, we take expectation over A_t, because actions can depend on previous rewards, which are random

Recap: Basic Bandit Formalism

- A good algorithm, like UCB or Thompson Sampling, gets sublinear regret:

$$R(T) \sim \sum_{i \in [K]} \frac{\log T}{\Delta_i}$$

- Getting sublinear regret involves a careful trade-off of exploration and exploitation
- Regret scales linearly with number of arms
  - We must try each arm a few arms to know whether it is good or not

Adding Structure to Multi-Armed Bandits

We now make a couple of new assumptions to the basic MAB framework

In the **linear bandits model**, we assume:

- each arm i has a *context vector* $x_i \in \mathbb{R}^d$ (fixed and known)
- the reward distribution has the following structure:

$$\nu_i = \mathcal{N}(\mu_i, 1)$$

$$\mu_i = \langle \theta, x_i \rangle$$

Where $\theta \in \mathbb{R}^d$ is fixed, but unknown

Adding Structure to Multi-Armed Bandits

Understanding the assumptions

The reward distributions across different arms are related through θ: $\mu_i = \langle \theta, x_i \rangle$

- Suppose θ was known, in addition to x_i
    - Then we know μ_i for each arm $i \Rightarrow$ no problem left to solve!
- Suppose θ was known, but x_i was unknown
    - Then we go back to basic multi-armed bandits, where μ_i have no relation

So what change does the new model (θ unknown, x_i known) bring?

Notation

- x_i : context vector for arm i; $i \in 1,\ldots,K$
- $x(t)$: context vector of arm played at time t; $t \in \{1,2,\ldots,T\}$
  - $x(t) = x_{A_t}$
- x^k : k-th component of a context vector x; $k \in \{1,2,\ldots,d\}$
  - $x_i \in \mathbb{R}^d$, $x(t) \in \mathbb{R}^d$, $x^k \in \mathbb{R}$

We will also introduce a new notation for reward:

- $y(t)$: reward after pulling an arm in round t (earlier, this was X_t)

Advantage of Context Information

- Consider a setting with $K = 5,\ T = 5,\ d = 2$
- Suppose the goal is best arm identification (only for this example)
- Simple strategy: play each arm once, report the arm with best reward
- With context vectors, can we do better?
    - We can try to *estimate* $\hat{\theta}$ from data:

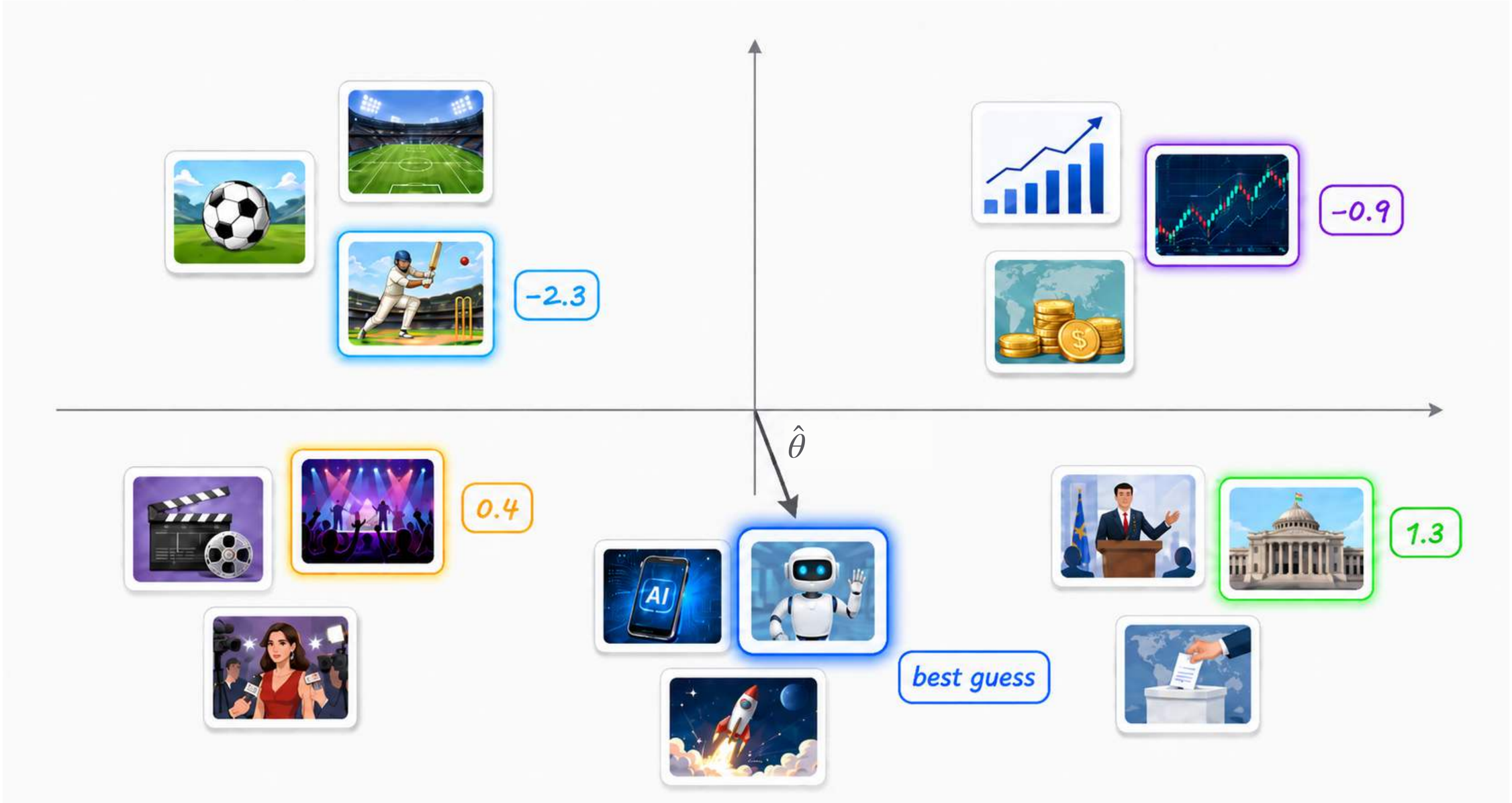
$$\big(x(1), y(1)\big), \big(x(2), y(2)\big), \ldots \big(x(10), y(10)\big)$$

    - Using the estimate, we can calculate $\hat{i} = \arg\max_{i \in [K]} \langle \hat{\theta}, x_i \rangle$

1 data point for each unknown quantity $\mu_i \in \mathbb{R}$

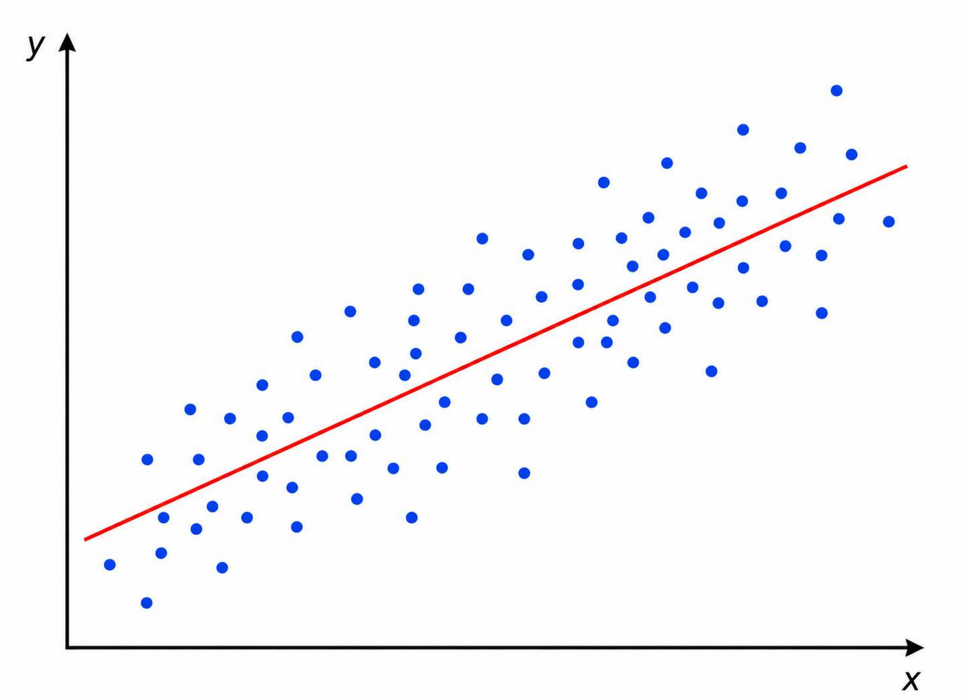
10 data points for a single unknown quantity $\theta \in \mathbb{R}^2$

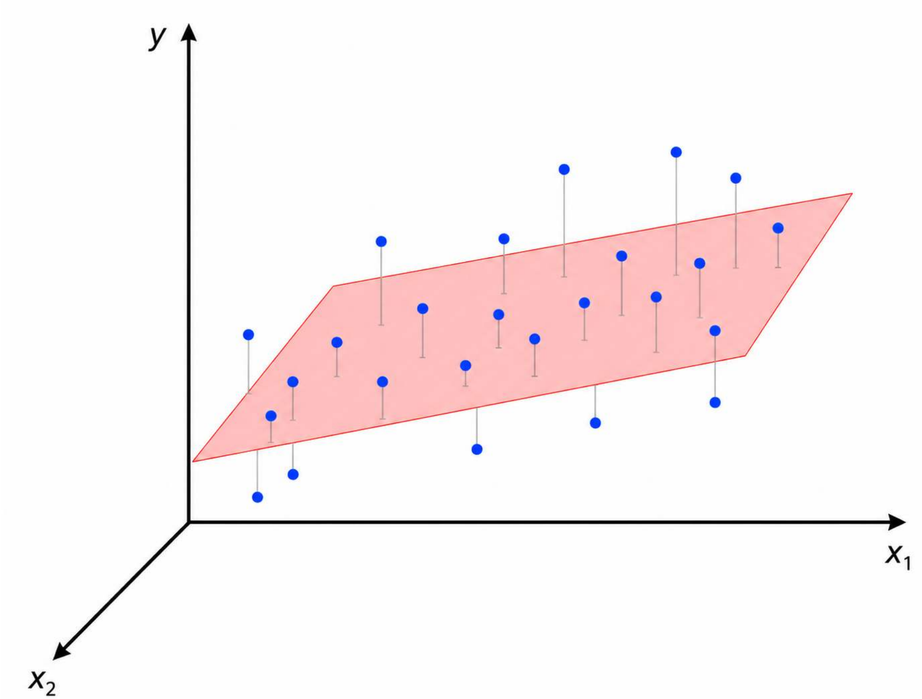
Revisiting Previous Example



Estimating θ

- How should one calculate $\hat{\theta}$? The problem is identical to linear regression
- Data: $\big(x(1), y(1)\big), \big(x(2), y(2)\big), \ldots \big(x(T), y(T)\big)$ where $y(t) = \langle \theta, x(t) \rangle + \eta_t$





Estimating θ

- How should one calculate $\hat{\theta}$? The problem is identical to linear regression
- Data: $\big(x(1), y(1)\big), \big(x(2), y(2)\big), \dots \big(x(T), y(T)\big)$ where $y(t) = \langle \theta, x(t) \rangle + \eta_t$
- Estimate θ by solving the least-squares problem

$$\hat{\theta} = \arg\min_{\theta} \sum_{t=1}^{T} \big(y(t) - \langle \theta, x(t) \rangle\big)^2$$

- For regret minimisation, we need to make a smart decision at each time t
  - We need to calculate this estimate repeatedly

A Closed Form Solution

At any time t, based on the data available, the estimate can be calculated as

$$\hat{\theta}_t = V_t^{-1} b_t$$

$$\hat{\theta}_t = \arg\min_{\theta} \sum_{s=1}^{t} \left(y(s) - \langle \theta, x(s) \rangle \right)^2$$

$$V_t = \sum_{s=1}^{t} x(s) x(s)^\top \qquad b_t = \sum_{s=1}^{t} x(s) y(s)$$

- $V_t \in \mathbb{R}^{d \times d}$, $b_t \in \mathbb{R}^d$
- Both can be easily incremented with every new data point
- All operations are matrix-vector operations—easy. Computing inverse?

Introducing a Regulariser

- When number of datapoints t is less than d, the matrix V_t will not be invertible!
  - For $V_t \in \mathbb{R}^{d \times d}$ to be invertible, the dimension of its column space should be d
  - For any matrix of the form $\sum_{s=1}^{T} x(s)x(s)^\top$, the column space is the span of $\{x(1), \ldots, x(t)\}$
  - t vectors cannot span a d-dimensional space (if $t < d$)
- Solution: we add a regulariser to the least-squares problem

$$\hat{\theta}_t = \arg\min_{\theta} \sum_{t=1}^{T} \Big(y(s) - \langle \theta, x(s) \rangle\Big)^2 + \lambda \|\theta\|^2$$

- $V_t = \lambda I + \sum_{s=1}^{t} x(s)x(s)^\top$, which is always invertible

Towards a Bandit Algorithm

- We have K arms, each arm i with a known context vector x_i
- Mean rewards follow the structure $\mu_i = \langle \theta, x_i \rangle$ for some unknown $\theta \in \mathbb{R}^d$
- We can calculate an estimate $\hat{\theta}_t$ at each step t
- $\hat{\theta}_t$ gives us an estimate of the arm rewards, and hence a guess for the best arm
- Using this estimate, we can act greedily at each step t: $A_{t+1} = \arg\max_{i \in [K]} \langle \hat{\theta}_t, x_i \rangle$
- But greedy strategy, or even ε-greedy, is not good for regret minimisation!

Towards UCB for Linear Bandits

- UCB algorithm requires an uncertainty measure
- For basic bandits, we were estimating μ_i
  - We maintained confidence intervals around $\hat{\mu}_i$
  - with the guarantee: $\mu_i \in [\hat{\mu}_i - \epsilon, \hat{\mu}_i + \epsilon]$ with high probability
- In linear bandits, we are estimating $\theta \in \mathbb{R}^d$
- We need a set $\mathscr{C}$, centred around $\hat{\theta}$, such that $\theta \in \mathscr{C}$ with high probability

Confidence Ellipsoids

- In high dimensions, we represent confidence sets using ellipsoids

$$\mathscr{C}_t = \left\{ \theta \in \mathbb{R}^d : \|\theta - \hat{\theta}_t\|_{V_t}^2 \leq \beta \right\}$$

Why is this an ellipsoid?

- $\|\theta\|_2^2 = \theta^\top \theta = \theta^\top I \theta$
- $\|\theta\|_2^2 \leq \beta$ denotes a ball around the origin
- $\|\theta - \hat{\theta}\|_2^2 \leq \beta$ denotes a ball around $\hat{\theta}$

Confidence Ellipsoids

- In high dimensions, we represent confidence sets using ellipsoids

$$\mathscr{C}_t = \left\{ \theta \in \mathbb{R}^d : \|\theta - \hat{\theta}_t\|_{V_t}^2 \leq \beta \right\}$$

Why is this an ellipsoid?

- $\|\theta\|_V^2 = \theta^\top V \theta$
- If V is a diagonal matrix, $\|\theta\|_V^2 = V_{1,1}(\theta^1)^2 + \ldots + V_{d,d}(\theta^d)^2$
- So $\|\theta\|_V^2 \leq \beta$ is an ellipsoid around origin

radius along axis i : $\sqrt{\beta / V_{i,i}}$

Confidence Ellipsoids

- In high dimensions, we represent confidence sets using ellipsoids

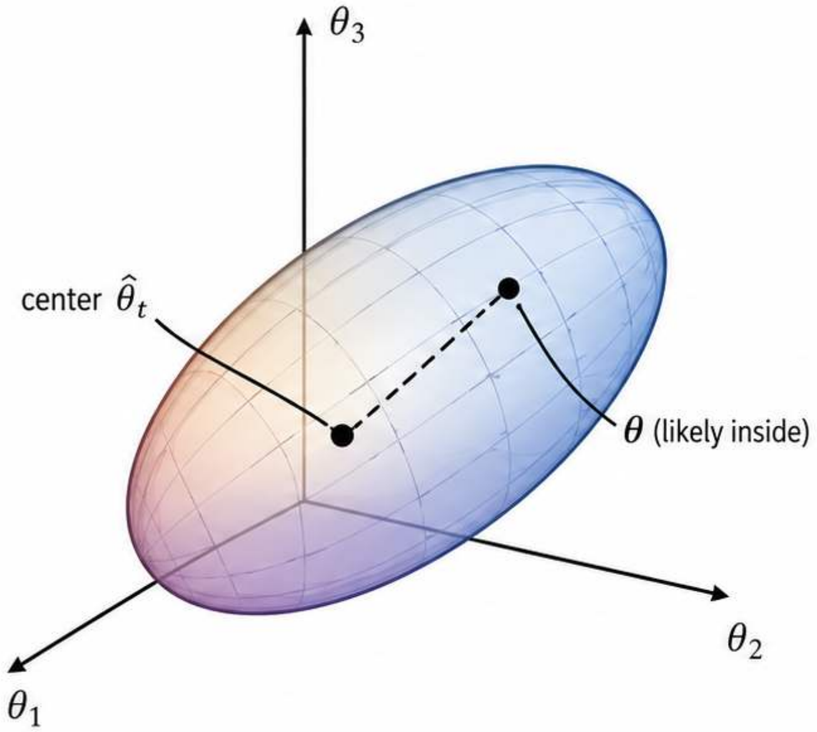
$$\mathscr{C}_t = \left\{ \theta \in \mathbb{R}^d : \|\theta - \hat{\theta}_t\|_{V_t}^2 \leq \beta \right\}$$

Why is this an ellipsoid?

- In general, $V_t = \sum_{s=1}^{t} x(s)x(s)^\top = \sum_{k=1}^{d} \lambda_k v_k v_k^\top$ ← Eigenvalue decomposition for symmetric matrices
- $\|\theta\|_V^2 \leq \beta$ is an ellipsoid with axes aligned along $(v_1, \ldots, v_d)$ and radius

Visual Representation

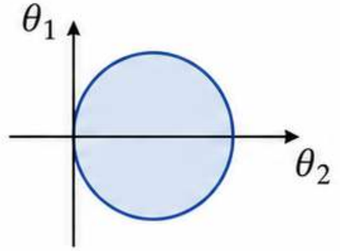
Confidence ellipsoid



$$(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_t)^T \boldsymbol{V}_t (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_t) \leq \beta_t$$

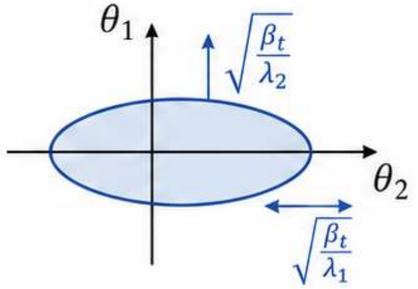
$$\boldsymbol{V}_t \succ 0, \quad \beta_t > 0$$

How V_t shapes the set



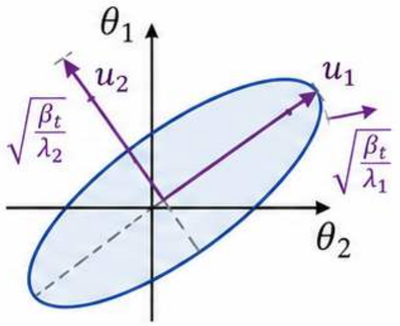
① $\lambda_1 = \lambda_2$

equal uncertainty
in all directions



② $\boldsymbol{V}_t = \mathrm{diag}(\lambda_1, \lambda_2)$

larger λ_i → shorter axis
smaller λ_i → longer axis



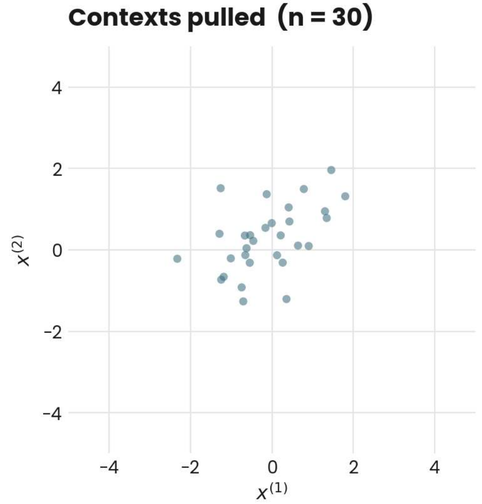
③ $\boldsymbol{V}_t = \boldsymbol{U}\,\mathrm{diag}(\lambda_1, \lambda_2)\,\boldsymbol{U}^T$

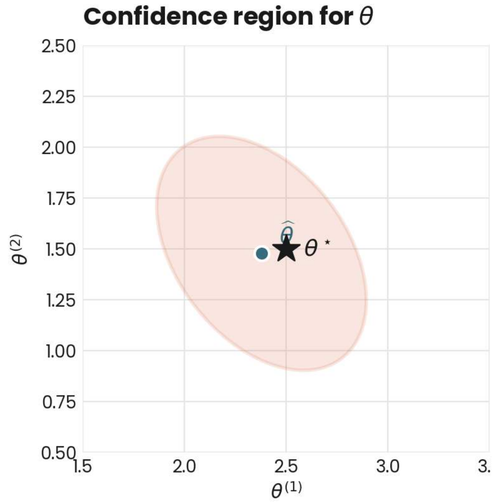
eigenvectors set orientation

Larger eigenvalue → more information → shorter axis.

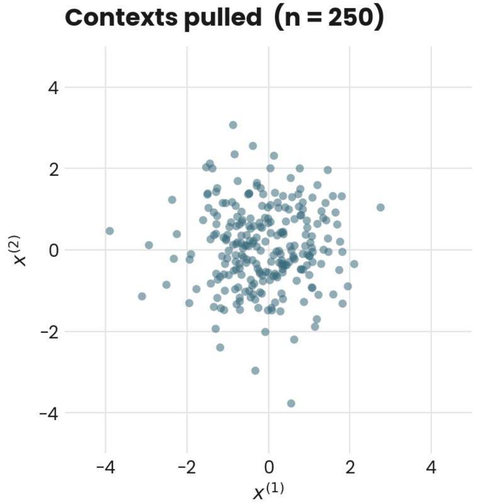
Confidence Ellipsoids — Different Settings

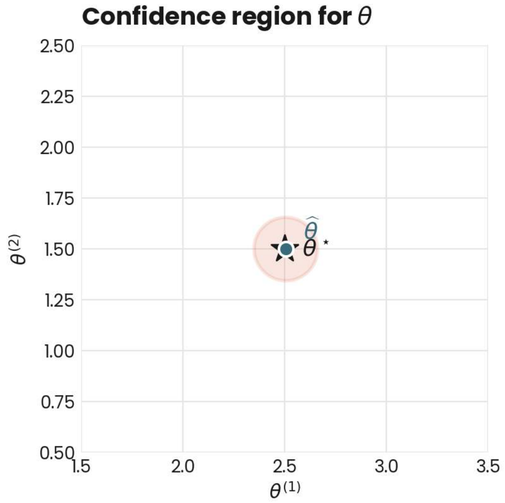
Isotropic exploration — few samples



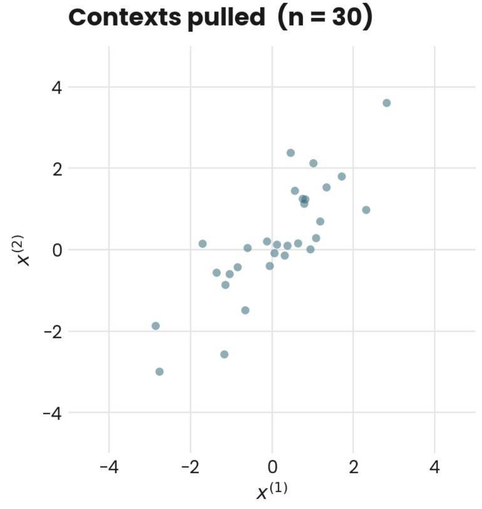


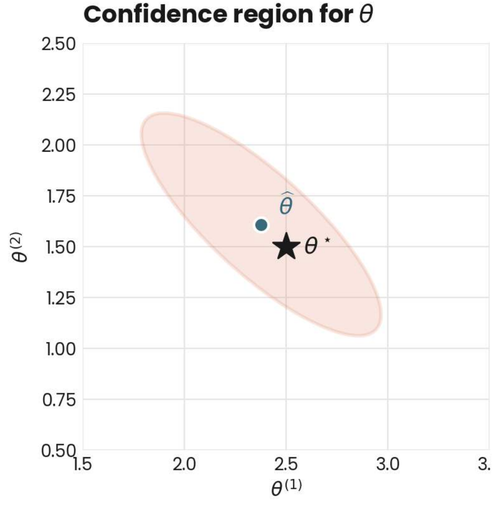
Isotropic exploration — many samples



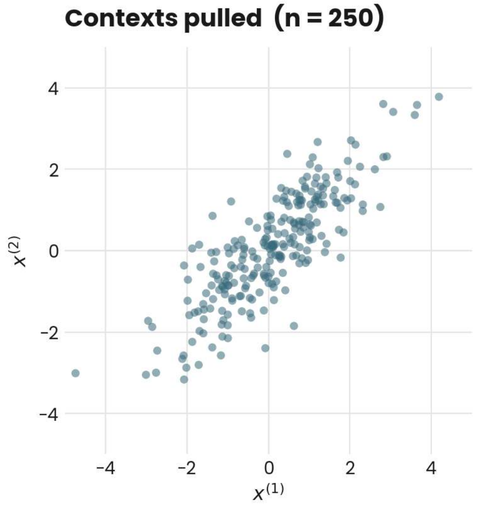


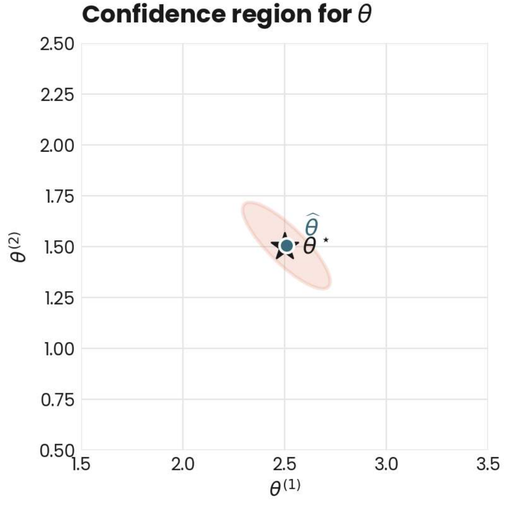
Anisotropic exploration — few samples





Anisotropic exploration — many samples





The LinUCB Algorithm

- $\mathscr{C}_t = \left\{ \theta \in \mathbb{R}^d : \|\theta - \hat{\theta}_{t-1}\|^2_{V_{t-1}} \leq \beta \right\}$
- $V_t = \lambda I + \sum_{s=1}^{t} x(s)x(s)^\top$
- $\hat{\theta}_t = V_t^{-1} b_t$
- $b_t = \sum_{s=1}^{t} x(s)y(s)$

LinUCB Algorithm

For $t = 1,2,\ldots,T$:

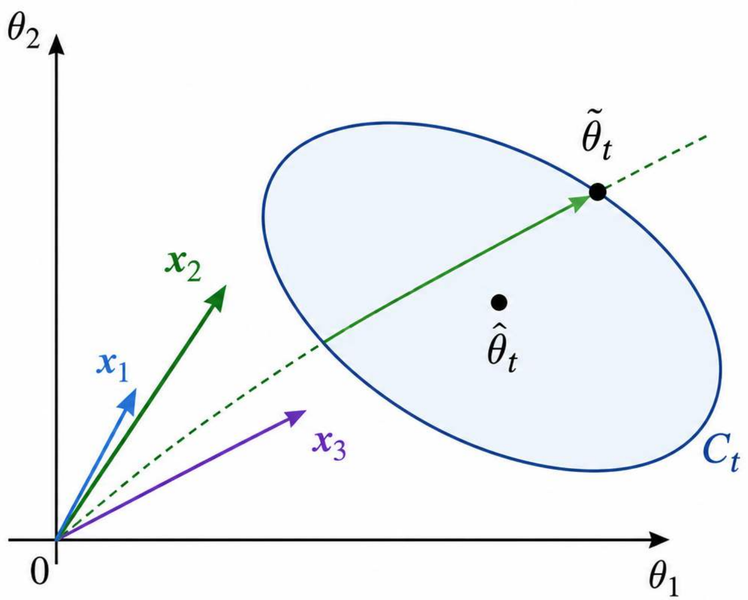
For all arms $i \in [K]$:

Compute $UCB_t(i) = \max_{\theta \in \mathscr{C}_t} \langle \theta, x_i \rangle$

Play $A_t = \arg\max_{i \in [K]} UCB_{t-1}(i)$

Observe $y(t)$, update $\hat{\theta}_t$, V_t, $\mathscr{C}_t$

The LinUCB Algorithm



$$UCB_t(\mathbf{x}_2) = \max_{\theta \in C_t} \langle \theta, \mathbf{x}_2 \rangle$$

Optimistic value of arm $\mathbf{x}_2$: best plausible reward over C_t

$$A_t = \arg\max_a \max_{\theta \in C_t} \langle \theta, a \rangle$$

LinUCB Algorithm

For $t = 1,2,\ldots,T$:

For all arms $i \in [K]$:

Compute $UCB_t(i) = \max_{\theta \in \mathscr{C}_t} \langle \theta, x_i \rangle$

Play $A_t = \arg\max_{i \in [K]} UCB_{t-1}(i)$

Observe $y(t)$, update $\hat{\theta}_t$, V_t, $\mathscr{C}_t$

Coming Up Next:

How LinUCB works in practice