CSE 250A. Principles of AI

Probabilistic Reasoning and Decision-Making

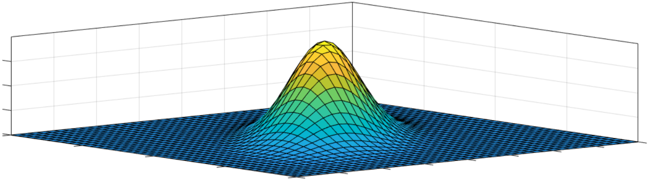
Lecture 16 – latent variable models (wrap-up), reinforcement learning (intro)

Lawrence Saul
Department of Computer Science and Engineering
University of California, San Diego

Fall 2021

Outline

Multivariate Gaussian distribution



- **Probability density function (PDF) over $\Re^d$**

$$P(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d \det(\boldsymbol{\Sigma})}} \exp\left\{ -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) \right\}$$

- **Parameters**

$$\textbf{mean} \quad \boldsymbol{\mu} = \mathrm{E}[\mathbf{x}] = \int_{\Re^d} P(\mathbf{x})\,\mathbf{x}$$

$$\textbf{covariance} \quad \boldsymbol{\Sigma} = \mathrm{E}[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top] = \int_{\Re^d} P(\mathbf{x})\,(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top$$

Maximum likelihood (ML) estimation

- **Learning from data**

  Let $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ be *i.i.d.* examples in $\Re^d$.
  Assume $\mathbf{x}$ is normally distributed: how to estimate $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$?

- **ML estimates**

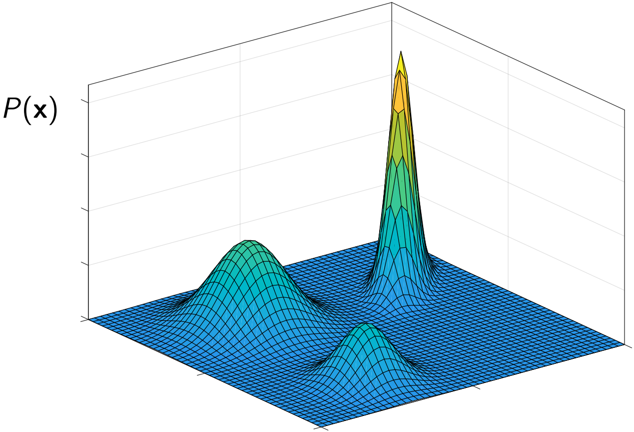
$$\boldsymbol{\mu}_{\mathrm{ML}} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}_t$$

sample mean

$$\boldsymbol{\Sigma}_{\mathrm{ML}} = \frac{1}{T}\sum_{t=1}^{T}(\mathbf{x}_t - \boldsymbol{\mu}_{\mathrm{ML}})(\mathbf{x}_t - \boldsymbol{\mu}_{\mathrm{ML}})^\top$$

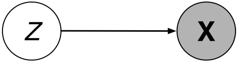
sample covariance matrix

How to model multimodal distributions?



We can model this as a mixture of $k = 3$ Gaussian distributions.

Gaussian mixture model (GMM)



- **Random variables**

$\mathbf{x} \in \Re^d$	real-valued vector	(observed)
$z \in \{1, 2, \ldots, k\}$	cluster label	(hidden)

- **Conditional probability distributions**

$P(Z=i)$	fraction of data in i^{th} cluster
$P(\mathbf{x}\|Z=i)$	multivariate Gaussian $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

$\boxed{\textbf{Each cluster has its own mean } \boldsymbol{\mu}_i \textbf{ and covariance matrix } \boldsymbol{\Sigma}_i.}$

EM algorithm

- **E-step: compute posterior probabilities**

$$P(Z\!=\!i|\mathbf{x}_t) \;=\; \frac{P(\mathbf{x}_t|Z\!=\!i)\,P(Z\!=\!i)}{\sum_j P(\mathbf{x}_t|Z\!=\!j)\,P(Z\!=\!j)} \qquad \boxed{\textbf{Bayes rule}}$$

- **M-step: update model parameters**

$$P(Z\!=\!i) \;\leftarrow\; \frac{1}{T}\sum_t P(Z\!=\!i|\mathbf{x}_t)$$

$$\boldsymbol{\mu}_i \;\leftarrow\; \frac{\sum_{t=1}^{T} P(Z\!=\!i|\mathbf{x}_t)\,\mathbf{x}_t}{\sum_{t=1}^{T} P(Z\!=\!i|\mathbf{x}_t)}$$

$$\boldsymbol{\Sigma}_i \;\leftarrow\; \frac{\sum_{t=1}^{T} P(Z\!=\!i|\mathbf{x}_t)\,(\mathbf{x}_t-\boldsymbol{\mu}_i)(\mathbf{x}_t-\boldsymbol{\mu}_i)^\top}{\sum_{t=1}^{T} P(Z\!=\!i|\mathbf{x}_t)}$$

Outline

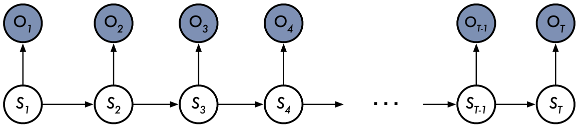
1. **Review**
2. **Linear dynamical systems**
3. **Reinforcement learning**

Motivating example

From sensor measurements, how to determine the location and bearing of a missile?

All these variables are fundamentally **continuous**.

Discrete versus continuous



- **Discrete dynamical system** (e.g., HMM)

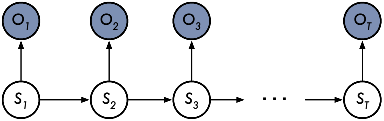
$$\begin{array}{rcll} \text{observations} & o_t & \in & \{1, 2, \ldots, m\} \\ \text{hidden states} & s_t & \in & \{1, 2, \ldots, n\} \end{array}$$

- **Continuous dynamical system**

$$\begin{array}{rcll} \text{observations} & \mathbf{o_t} & \in & \Re^m \\ \text{hidden states} & \mathbf{s_t} & \in & \Re^n \end{array}$$

Linear dynamical system

- **Belief network**



observations $o_t \in \Re^m$

states $s_t \in \Re^n$

- **Conditional PDFs**

$$\begin{array}{rrcllll} & \mathbf{s_1} & \sim & \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) & \text{with} & \boldsymbol{\mu}_0 \in \Re^n, & \boldsymbol{\Sigma}_0 \in \Re^{n\times n} \\ (t>1) & \mathbf{s_t} & \sim & \mathcal{N}(\mathbf{A}\mathbf{s_{t-1}}, \boldsymbol{\Sigma}_\mathrm{H}) & \text{with} & \mathbf{A} \in \Re^{n\times n}, & \boldsymbol{\Sigma}_\mathrm{H} \in \Re^{n\times n} \\ & \mathbf{o_t} & \sim & \mathcal{N}(\mathbf{B}\mathbf{s_t}, \boldsymbol{\Sigma}_\mathrm{O}) & \text{with} & \mathbf{B} \in \Re^{m\times n}, & \boldsymbol{\Sigma}_\mathrm{O} \in \Re^{m\times m} \end{array}$$

Why **tractable**? Because $P(\mathbf{s_1}, \mathbf{o_1}, \ldots, \mathbf{s_T}, \mathbf{o_T})$ is jointly Gaussian.

Why **linear**? Because $\mathrm{E}[\mathbf{s_t}|\mathbf{s_{t-1}}] = \mathbf{A}\mathbf{s_{t-1}}$ and $\mathrm{E}[\mathbf{o_t}|\mathbf{s_t}] = \mathbf{B}\mathbf{s_t}$.

Belief updating

How to compute $P(\mathbf{s_t}|\mathbf{o_1}, \mathbf{o_2}, \ldots, \mathbf{o_t})$?
In linear dynamical systems, this is known as **Kalman filtering**.

- **Recall key property:**

  Since $P(\mathbf{s_1}, \mathbf{o_1}, \ldots, \mathbf{s_T}, \mathbf{o_T})$ is multivariate Gaussian,
  so are all of its marginal and conditional distributions.

- **Why this helps:**

  $P(\mathbf{s_t}|\mathbf{o_1}, \ldots, \mathbf{o_t})$ is multivariate Gaussian.
  It is enough to track the mean and covariance:

$$\begin{aligned}
\boldsymbol{\mu}_t &= \mathrm{E}[\mathbf{s_t}|\mathbf{o_1}, \mathbf{o_2}, \ldots, \mathbf{o_t}] \\
\boldsymbol{\Sigma}_t &= \mathrm{E}[(\mathbf{s_t} - \boldsymbol{\mu}_t)(\mathbf{s_t} - \boldsymbol{\mu}_t)^\top|\mathbf{o_1}, \mathbf{o_2}, \ldots, \mathbf{o_t}].
\end{aligned}$$

  If we know these statistics, we know the entire distribution.

Kalman filtering

Update (stated without proof):

$$\boldsymbol{\mu}_{t+1} = \underbrace{\mathbf{A}\boldsymbol{\mu}_t}_{\substack{\text{extrapolate}\\ \text{missile from}\\ \text{time } t}} + \mathbf{K}_{t+1}\underbrace{(\mathbf{o_{t+1}} - \mathbf{BA}\boldsymbol{\mu}_t)}_{\text{error signal}}$$

The matrix $\mathbf{K}_{t+1} \in \Re^{n \times m}$ multiplies the error signal.
It corrects the update for unexpected observations.

Outline

1. **Review**

2. **Linear dynamical systems**

3. **Reinforcement learning**

Reinforcement learning (RL)

How can autonomous decision-making agents learn from experience in the world?

Many applications:

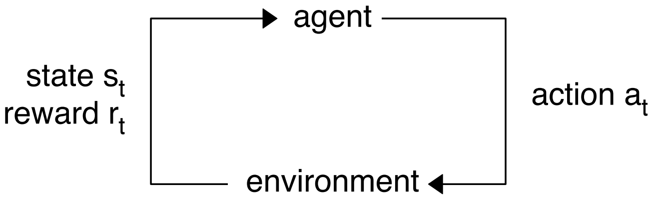
- robot navigation
- game-playing AIs
- operations research

Agents are actors of any kind: they can be *embodied* in the physical world or *embedded* in a virtual environment.

Challenges of RL

1. How to learn in noisy, uncertain environments?
2. When to explore, versus when to exploit?
3. How to learn from delayed (versus immediate) rewards?
4. How to learn from evaluative (versus instructive) feedback?
5. How to navigate complex worlds with tractable models?
6. How to prove computational guarantees
   (e.g., convergence, optimality, efficiency)?

A probabilistic framework for RL



How do we formalize this process?
How do we handle uncertainty?

We define a **Markov decision process**.

Definition

A Markov decision process (MDP) is defined by the following:

- A **state space** $\mathcal{S}$ with states $s \in \mathcal{S}$
- An **action space** $\mathcal{A}$ with actions $a \in \mathcal{A}$
- **Transition probabilities**

$$P(s'|s,a) \; = \; P(S_{t+1}=s'|S_t=s, A_t=a)$$

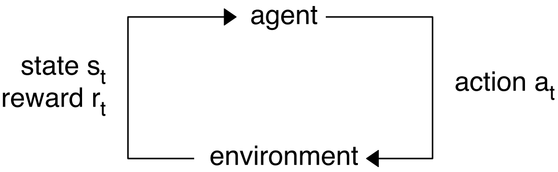
  that indicate, at any time t, how frequently an agent moves from state s to state s' after taking action a

- A **reward function** $R(s, s', a)$, providing immediate feedback when the agent takes action a in state s and moves to state s'.

  Rewards are **scalar**: the higher, the better.

$$\boxed{\text{MDP} \; = \; \{\mathcal{S}, \mathcal{A}, P(s'|s,a), R(s,s',a)\}}$$

Markov assumptions



1. **Conditional independence**

$$
\begin{aligned}
& P(S_{t+1}=s' | S_t=s, A_t=a) \\
& \quad = \quad P(S_{t+1}=s' | S_t=s, A_t=a, S_{t-1}, A_{t-1}, S_{t-2}, A_{t-2}, \ldots)
\end{aligned}
$$

2. **Transition probabilities are constant over time:**

$$
P(S_{t+1}=s' | S_t=s, A_t=a) \; = \; P(\underbrace{S_{t+1+\tau}=s' | S_{t+\tau}=s, A_{t+\tau}=a}_{\text{shifted by } \tau})
$$

Simplifications for CSE 250a

1. State space is discrete and finite: $\mathcal{S} = \{1, 2, \ldots, |\mathcal{S}|\}$.
2. Action space is discrete and finite: $\mathcal{A} = \{1, 2, \ldots, |\mathcal{A}|\}$.
3. Rewards depend only on the state: $R(s, s', a) = R(s)$.
4. Rewards are bounded: $\max_s |R(s)| < \infty$.
5. Rewards are deterministic.

$$\boxed{\text{MDP} \;=\; \{\mathcal{S}, \mathcal{A}, P(s'|s,a), R(s)\}}$$

Example: board games (with dice)

agent

state s_t
reward r_t

action a_t

environment

$s \in \mathcal{S}$ board position and results of roll of dice

$a \in \mathcal{A}$ one of any allowed moves

$$R(s) = \begin{cases} +1 & \text{if agent wins the game} \\ -1 & \text{if agent loses the game} \\ 0 & \text{for all preceding board positions} \end{cases}$$

$$P(s'|s,a) \sim \begin{cases} \text{agent moves} \\ \text{opponent rolls dice} \\ \text{opponent moves} \\ \text{agent rolls dice} \end{cases}$$

Decision-making in MDPs

- **Definition**

  A **policy** $\pi : \mathcal{S} \rightarrow \mathcal{A}$ is a mapping of states to actions.
  In this class we will only consider deterministic policies.

- **Number of policies**

  If there are $|\mathcal{A}|$ possible actions in each of $|\mathcal{S}|$ states, then there are *combinatorially* many policies:

$$\#\ \text{policies} \;=\; |\mathcal{A}|^{|\mathcal{S}|}$$

- **Experience under policy π**

$$\begin{array}{rcccccc} \text{state} & s_0 & \xrightarrow{\pi(s_0)} & s_1 & \xrightarrow{\pi(s_1)} & s_2 & \cdots \\ \text{reward} & r_0 & & r_1 & & r_2 & \cdots \end{array}$$

Transitions occur with probabilities $P(s'|s, \pi(s))$.

How to measure long-term return?

1. **Finite-horizon return**

$$\text{return} \;=\; \frac{1}{T}(r_0 + r_1 + \cdots + r_{T-1}) \quad \text{for a } T\text{-step horizon}$$

2. **Undiscounted return with infinite horizon**

$$\text{return} \;=\; \lim_{T\to\infty}\left[\frac{1}{T}\sum_{t=0}^{T-1} r_t\right]$$

These are the most obvious ways to accumulate rewards.
But they are **not** the most commonly used in practice ...

How to measure long-term return? (con't)

3. **Discounted return with infinite horizon**

   Let $\gamma \in [0, 1)$ denote the so-called **discount factor**.
   Then define

$$\text{return} \;=\; r_0 + \gamma r_1 + \gamma^2 r_2 + \gamma^3 r_3 + \cdots \;=\; \sum_{t=0}^{\infty} \gamma^t r_t$$

   When $\gamma \ll 1$, future rewards are heavily discounted.
   These returns can be optimized by short-sighted agents.

   When γ is close to 1, future rewards are lightly discounted.
   These returns can only be optimized by far-sighted agents.

Motivation for $\gamma \in [0, 1)$

Psychologist: *Why discount rewards from the distant future?*
Economist: *Why favor investments with short-term payoffs?*

1. **Intuition**

   Many models are only approximations to the real world; we should not attempt to extrapolate them indefinitely.

2. **Mathematical convenience**

   Discounted returns lead to simple iterative algorithms with strong guarantees of convergence.

What to optimize?

The discounted return $\sum_{t=0}^{\infty} \gamma^t r_t$ is a random variable.
But we can try to optimize its expected value:

$$\mathrm{E}^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \,\middle|\, s_0 = s\right] =$$

the expected value of the discounted infinite-horizon return, starting in state s at time $t=0$, and following policy π.

Maximizing the expected return is:

– generally wiser than maximizing the best-case return,
– but not as robust as minimizing the worst-case return.

Next lecture: how to compute this expected return,
and how to find the policy that maximizes it ...