Advanced Inference & Search Algorithms

Recitation: Kalman Filters, Gibbs, A*, and MCTS

Current Week

Overview

Kalman Filters: Overview

What is a Kalman Filter?

- It optimally estimates the hidden state of a linear dynamical system disturbed by Gaussian noise.
- This is exact inference for a **Linear-Gaussian HMM**:

$$x_t = Ax_{t-1} + w_t$$
$$y_t = Hx_t + v_t$$

- We recursively compute $P(x_t \mid y_{0:t}) = \mathcal{N}(\hat{x}_{t|t}, Q_{t|t})$.

Notation

- x_t: State vector.
- y_t: Observation vector.
- A: Process model.
- H: Measurement model.
- $w_t \sim \mathcal{N}(0, W)$: process noise.
- $v_t \sim \mathcal{N}(0, R)$: measurement noise.

Kalman Filters: Equations

1. Predict (Transition Update)

Push the belief through the dynamics. Transition adds uncertainty:

$$\hat{x}_{t|t-1} = A\hat{x}_{t-1|t-1}$$
$$Q_{t|t-1} = AQ_{t-1|t-1}A^T + W$$

2. Measurement Update

Fuse prediction with the new observation y_t. Observation reduces uncertainty:

$$K_t = Q_{t|t-1}H^T(HQ_{t|t-1}H^T + R)^{-1}$$
$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t(y_t - H\hat{x}_{t|t-1})$$
$$Q_{t|t} = (I - K_tH)Q_{t|t-1}$$

Practice: Kalman Filters

Problem 1: 1D Predict & Update

Assume a 1D system where dynamics $A = 1$ and emissions $H = 1$. The process noise variance is $W = 0.5$ and measurement noise is $R = 2.0$.

Your previous belief is $P(x_{t-1} \mid y_{0:t-1}) = \mathcal{N}(10, 1.0)$.

1. Calculate the prior prediction: mean $\hat{x}_{t|t-1}$ and variance $Q_{t|t-1}$.
2. A new measurement arrives: $y_t = 13$. Calculate the Kalman Gain K_t.
3. What is your new updated state belief $x_{t|t} \sim \mathcal{N}(\hat{x}_{t|t}, Q_{t|t})$?

Practice: Kalman Filters (Concepts)

Problem 2: Max Likelihood Trajectory

When using a Kalman filter, if we want to recover the maximum likelihood trajectory, can we discard the history of observations and just record the most likely states while filtering? Explain.

Practice: Kalman Filters (Modeling)

Problem 3: Population Tracking

Consider three species U, V, and W that grow independently, exponentially with growth rates: U grows 2%/hr, V grows 6%/hr, and W grows 11%/hr. The goal is to estimate the initial size of each based on measurements of the total population.
Let $x_U(t)$ denote population of U after t hours, so $x_U(t+1) = 1.02x_U(t)$, similarly for $x_V(t)$ and $x_W(t)$. The total measurements are $y(t) = x_U(t) + x_V(t) + x_W(t) + v(t)$, where $v(t) \sim \mathcal{N}(0, 0.36)$.
Prior is that initial populations are IID $\mathcal{N}(\text{mean} = 6, \text{variance} = 2)$.

How do you formulate this as a Kalman filtering problem by providing A, H, W, R?

Gibbs Sampling: Overview

The Gibbs Solution (MCMC)

- Gibbs sampling is a simple type of Markov chain Monte Carlo (MCMC).
- The stationary distribution of the chain is the desired joint distribution $P(V_1, ..., V_n)$.
- We sample one variable at a time from its conditional distribution given its Markov blanket.

Execution details

Algorithm:

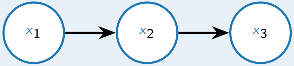
1. Initialize values $\overline{v} = (v_1, \ldots, v_N)$ at random.
2. Loop: Choose i from $1, \ldots, N$ (e.g., via systematic scan or random scan).
3. Set v_i to a sample from $P(V_i \mid v_{mb(V_i)})$.

Burn-in phase: Run the chain for a while and throw those samples away so we are in the stationary distribution.

Ergodicity: If there are no 0 entries in the factors, the chain is ergodic.

Gibbs Sampling: Markov Blanket & Process

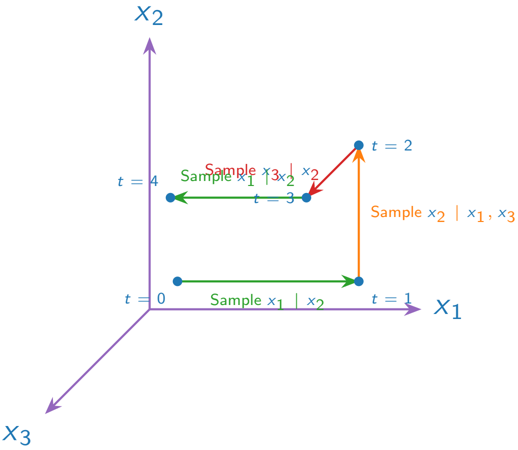
1. Bayesian Network



2. Markov Blankets

Gibbs relies on sampling from a variable's conditional distribution given its **Markov blanket** (parents, children, and children's parents).

- $MB(x_1) = \{x_2\}$
- $MB(x_2) = \{x_1, x_3\}$
- $MB(x_3) = \{x_2\}$



We transition through the state space by resampling one variable at a time given its Markov blanket, moving along one axis per step.

Practice: Gibbs Sampling

Problem 4: Gaussian Bayesian Network

Consider the earlier Bayesian Network: $x_1 \to x_2 \to x_3$. Assume the conditional distributions are:

- $x_1 \sim \mathcal{N}(0, 1)$
- $x_2 \mid x_1 \sim \mathcal{N}(x_1, 1)$
- $x_3 \mid x_2 \sim \mathcal{N}(x_2, 1)$

Suppose our current state is $(x_1 = 2, x_2 = 0, x_3 = 6)$. We select x_2 to be resampled using Gibbs sampling.

1. What is the distribution $P(x_2 \mid x_1 = 2, x_3 = 6)$ we must sample from?
2. Draw a sample from this distribution (assume a random standard normal draw yields $z = 0.5$). What is the new state?

Importance & Particle Filtering: Overview

Importance Sampling

- Used when we can't sample from $P(x)$ directly.
- Sample x from a proposal distribution $Q(x)$.
- Weight the sample by $w = \frac{P(x)}{Q(x)}$.
- Statistics using weighted samples approximately match the statistics of P.

Particle Filtering

- Used for approximate inference in HMMs when dynamics aren't linear-Gaussian.
- **Initialize** N particles from the prior.
- **Update:** Sample $x_{t+1}^{[i]}$ from the transition model given $x_t^{[i]}$.
- **Weight:** Calculate $w^{[i]} = P(y_{t+1} \mid x_{t+1}^{[i]})$.
- **Resample:** Generate new particles by sampling N times with replacement, proportional to weights.

Practice: Importance Sampling

Problem 5: Proposal Distributions

Consider the probability distribution with density

$$p(x, y) \propto f(x, y) = \exp\left(-\frac{1}{2}(x^2 + y^2 + x^2y^2 + \cos(x + 0.1y) + 1)\right)$$

Describe an algorithm using importance sampling to obtain samples from this distribution. What is a good proposal distribution $q(x, y)$?

Kullback-Leibler Divergence: Overview

What is KL Divergence?

Kullback-Leibler Divergence is a measure of how far apart two distributions p and q are:

$$D_{\mathrm{KL}}(p \parallel q) = \sum_{x \in X} p(x) \log\left(\frac{p(x)}{q(x)}\right)$$

(where p and q are distributions defined on the same domain).

Problem 6: Divergence Extrema

In general, what are the minimum and maximum of $D_{\mathrm{KL}}(p \parallel q)$? In what situations do they occur?

Practice: Divergence Graphs

Problem 7: Calculating Divergence

Compute $D_{\mathrm{KL}}(p \parallel q)$ in the situations below, using $\log_2$.

a) $p(x)$ has peaks at 1 and 3 (mass 1/2 each), while $q(x)$ has misaligned peaks.

b) $p(x)$ has one peak at $x = 3$ (mass 1), while $q(x)$ has peaks at $x = 1$ and $x = 3$ (mass 1/2 each).

c) $p(x)$ has two peaks with mass 1/2. $q(x)$ has three peaks with mass 1/3.

d) $p(x)$ and $q(x)$ have identical overlapping peaks with mass 1/2.

Uniform-Cost Search (UCS)

Overview

- Best-first search with $f(n) = n.\text{path\_cost}$.
- Assumes all step costs are positive.
- Expands nodes in contours of equal path cost.
- The first path to a state that is **expanded** has the least cost.
- The first path to a state merely **generated** (or visited) need not have the least cost.

Properties

- Let C^* be the optimal solution cost, and $\epsilon > 0$ the minimum edge cost.
- Complexity depends on C^* and ϵ ($O(b^{\lfloor C^*/\epsilon \rfloor + 1})$).
- Closely related to Dijkstra's algorithm.

Informed State-Space Search Methods

Transition to Informed Search

- Without hints, we cannot do better than Uniform-Cost Search (exploring equally in all directions).
- We can use a **heuristic function** $h : \mathcal{S} \to \mathbb{R}$.
- $h(s)$ estimates the least-cost path from state s to a goal.
- Standard example: Euclidean distance in route finding.

Greedy Best-First Search (GBFS)

Overview

- Best-first search using only the heuristic: $f(n) = h(n.s)$ (or equivalently $h(s)$).
- Expands the node that seems closest to the goal.
- **Not guaranteed to be optimal** (can get stuck traversing a suboptimal but seemingly good path).
- Overcomes the expanding contours of UCS—often reaches a goal much faster.

A* Search: Overview

The Evaluation Function

- **A* is a best-first search** where $f(n) = n.\text{path\_cost} + h(n.s)$.
- n: Current node in the search tree (state in node is $n.s$).
- $n.\text{path\_cost}$: Exact cost incurred so far from start to n.
- $h(s)$ estimates the least remaining cost from state s to a goal.
- At extremes: if $h(s) = 0$, A* reduces to **Uniform-Cost Search**.

Guarantees and Optimizations

- **Admissible** iff $h(s) \leq h^*(s)$ for all $s \in \mathcal{S}$, where $h^*(s)$ is the true least path cost.
- Admissibility guarantees A* will find an optimal path in **tree search**.
- **Consistent** if $h(s) \leq c(s, a, s') + h(s')$ for all transitions (usually $h(\text{goal}) = 0$).
- Consistency implies f is nondecreasing along any path. It ensures optimality in **graph search**.
- In graph search, keep only the best path found so far to each state; with consistency, the first expanded path to a state is optimal.

More about A*

Search Contours and Efficiency

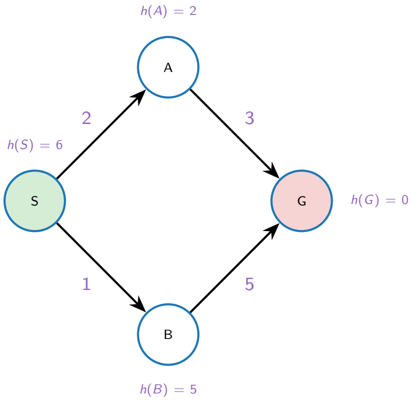
- Better heuristics stretch the search contours toward the goal.
- Let C^* be the optimal solution cost.
- A* expands all nodes on paths with $f(n) < C^*$.
- A* expands no nodes with $f(n) > C^*$.
- If $h = h^*$, A* expands only nodes on an optimal path.
- (Recall: if $h = 0$, A* reduces to UCS and expands equally in all directions).

Heuristic Functions

Sources of Heuristics

- The ideal heuristic is admissible, consistent, close to h^*, and efficient to compute.
- **Problem Relaxation:** We can derive heuristics by relaxing the constraints of the original problem (e.g., ignoring obstacles to get Euclidean distance). The exact cost of a relaxed problem is an admissible heuristic for the original.

Practice: A* Search (Execution)

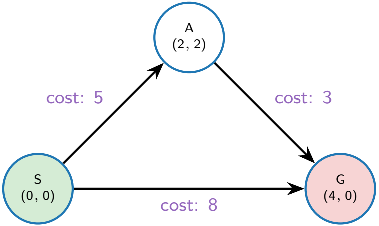


Problem 8: Running A*

You are at the start node S.

1. Expand S. Calculate $f(A)$ and $f(B)$. Which node is popped from the frontier next?
2. Is the given heuristic h **admissible**? Justify your answer.

Practice: A* Search (Heuristics)



True least path cost to $G(4,0)$:
$h^*(S) = 8, \quad h^*(A) = 3$

Problem 9: 2D Heuristics

For a 2D position (x, y), the goal is $G = (4, 0)$. We define 5 heuristics. Determine if each is **admissible** and/or **consistent** for the explicit 3-node graph shown. Let $S = (0, 0)$ and $A = (2, 2)$.

- **h_1:** $h(x,y) = \frac{1}{2}|x-4| + |y|$
- **h_2:** $h(x,y) = \frac{1}{2}(x-4)^2$
- **h_3:** Can we find any heuristic here that is **Consistent but NOT Admissible**?
- **h_4:** $h(x,y) = |y|$
- **h_5:** $h(x,y) = (x-4)^2$

Recall: Admissible iff $h(s) \leq h^*(s)$. Consistent iff $h(s) \leq c(s,a,s') + h(s')$.

Why Admissible and Consistent?

Why Admissible?

- Admissibility guarantees that A* **Tree Search** returns an optimal (least-cost) path.
- If an inadmissible heuristic overestimates cost, A* might terminate early by expanding a suboptimal goal node on the frontier.

Why Consistent?

- Consistency ensures optimality in A* **Graph Search** (which prunes redundant paths by maintaining a closed set of visited states).
- It guarantees that the first time A* expands a state, it has found the optimal path to it.
- This lets graph search safely avoid redundant paths; without consistency, to remain optimal we might be forced to re-expand closed nodes.

MCTS: Overview

Why MCTS?

- MCTS is very general and efficient: can choose, horizon, and how to allocate compute.
- Can think of MCTS as using random action samples/rollouts to estimate a heuristic for each action path and allocating compute in a way that balances exploration and exploitation.
- Also see examples: davidkoplow.github.io/searching_demos

Upper Confidence Bound (UCB)

Balances **exploitation** (good past results) with **exploration** (visiting the unknown):

$$\text{UCT}_i = \frac{w_i}{n_i} + c\sqrt{\frac{\ln N}{n_i}}$$

- w_i, n_i: Wins / Visits for child node i.
- N: Total visits to the parent node.
- c: Exploration constant (often $\sqrt{2} \approx 1.414$).

Practice: MCTS

Problem 10: UCT (UCB for MCTS) Calculation

You are at the root node of an MCTS tree, which has been visited $N = 20$ times. It has two explored children:

- **Action A:** $w = 10$, $n = 12$
- **Action B:** $w = 3$, $n = 8$

Using the exploration parameter $c = 1.414$, calculate the UCT score for both Action A and Action B. Which action will the Selection phase choose to explore next?

(Hint: $\ln(20) \approx 2.996$*)*

Practice: MCTS

1 MCTS

1.1 MCTS Review

Upper-confidence bound The UCB formula is:

$$\text{UCB}(N, N_k, U_k) = \begin{cases} \frac{U_k}{N_k} + C\sqrt{\frac{\log N}{N_k}} & \text{if } N_k > 0 \\ \infty & \text{otherwise} \end{cases}$$

Here, N is the visitation count to the parent node; N_k is the visitation count to a child node; U_k is the total utility accumulated at the child.

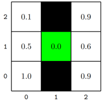
MCTS with UCB

1. **Selection** Starting at the root of the search tree, choose moves down to a leaf node according to UCB. Return the leaf node.
2. **Expansion** Grow the search tree from the leaf node by generating all its children. If the leaf node is already at the end of the horizon, skip this step.
3. **Simulation** Perform Monte-Carlo playout from the leaf node. Note that no new node is added to the search tree.
4. **Back-propagation** Use simulation's result to update all visitation counts and cumulative rewards, going up to the root.

1.2 MCTS Practice

Consider the reward maximization problem in the grid on the right.

- The agent starts at s_0 at the center
- The agent can move left, right, up, or down, except that it cannot move onto black grid cells
- Entering each grid cell yields a reward (specified within the box)
- Assume a horizon of 2
- Assume that $C = 1.0$



Simulate running MCTS and fill in the search tree below until you run out of cells (note: some table cells may remain empty). Assume that the random choices are $[0, 2, 1, 2]$ (select the first child $\rightarrow$ the third child $\rightarrow$ the second child $\rightarrow$ the third child). Break ties during the selection phase by selecting the leftmost child in the tree.

The following computations might be helpful:

- $\sqrt{\frac{\log(n)}{1}} - \sqrt{\frac{\log(n)}{n-1}} \leq 0.5$ for $2 \leq n \leq 4$
- $\sqrt{\frac{\log(n)}{2}} - \sqrt{\frac{\log(n)}{n-2}} \leq 0.5$ for $3 \leq n \leq 8$

6.4110/16.420 Problem Solving Session One Fall 2023