

MONTE CARLO, BAYESIAN NETWORKS

PROBABILISTIC DECISION MAKING VU – PRACTICALS
(REINFORCEMENT LEARNING KU)

Thomas Wedenig

Nov 07, 2025

Institute of Machine Learning and Neural Computation
Graz University of Technology, Austria

MONTE CARLO

Monte Carlo Estimation

Let X be an RV on state space \mathcal{X} with density p_X . With $f : \mathcal{X} \rightarrow \mathbb{R}^d$, we wish to compute

$$\mathbb{E}[f(X)] = \int_{\mathcal{X}} p_X(x) f(x) dx$$

With i.i.d. samples X_1, \dots, X_N from p_X , we can **approximate** this using Monte Carlo:

$$\hat{\mathbb{E}}_N := \frac{1}{N} \sum_{n=1}^N f(X_n)$$

- For any N , $\hat{\mathbb{E}}_N$ is unbiased 😊:

$$\mathbb{E} \left[\hat{\mathbb{E}}_N \right] = \mathbb{E} [f(X)]$$

- Variance decreases linearly with N :

$$\text{var} \left[\hat{\mathbb{E}}_N \right] = \frac{\text{var} [f(X)]}{N}$$

MONTE CARLO DEMO

$$\text{var} \left[\hat{\mathbb{E}}_N \right] = \frac{\text{var} [f(X)]}{N}$$

- By definition

$$\begin{aligned}\text{var}[Z] &:= \mathbb{E}[(\mathbb{E}[Z] - Z)^2] \\ &= \mathbb{E}[\mathbb{E}[Z]^2 - 2\mathbb{E}[Z]Z + Z^2] \\ &= \mathbb{E}[Z]^2 - 2\mathbb{E}[Z]\mathbb{E}[Z] + \mathbb{E}[Z^2] \\ &= \mathbb{E}[Z^2] - \mathbb{E}[Z]^2\end{aligned}$$

$$f(x) = \sqrt{1 - x^2}, \quad p(x) = \begin{cases} 1 & x \in [0, 1] \\ 0 & \text{otherwise} \end{cases}$$

- Recall:

$$\text{var}[f(X)] = \mathbb{E}[f(X)^2] - \mathbb{E}[f(X)]^2$$

- First term:

$$\mathbb{E}[f(X)^2] = \int_{-\infty}^{\infty} p(x)f(x)^2 dx = \int_0^1 1 - x^2 dx = \left[x - \frac{x^3}{3} \right]_0^1 = 1 - \frac{1}{3} = \frac{2}{3}$$

- By design, $\mathbb{E}[f(X)] = A/4$ and hence,

$$\text{var}[f(X)] = \frac{2}{3} - \left(\frac{A}{4} \right)^2$$

$$g(x, y) = \mathbf{1}[x^2 + y^2 \leq 1], \quad p(x, y) = \begin{cases} 1 & \text{if } x \in [0, 1] \text{ and } y \in [0, 1] \\ 0 & \text{otherwise} \end{cases}$$

- Recall:

$$\text{var}[g(X, Y)] = \mathbb{E}[g(X, Y)^2] - \mathbb{E}[g(X, Y)]^2$$

- Note that since $g(x, y) = g(x, y)^2$ for any x, y , we have

$$\mathbb{E}[g(X, Y)^2] = \mathbb{E}[g(X, Y)]$$

- By design, $\mathbb{E}[g(X, Y)] = A/4$ and hence,

$$\text{var}[g(X, Y)] = \frac{A}{4} - \left(\frac{A}{4}\right)^2$$

$$\text{var}[f(X)] = \frac{2}{3} - \left(\frac{A}{4}\right)^2$$

$$\text{var}[g(X, Y)] = \frac{A}{4} - \left(\frac{A}{4}\right)^2$$

- Since $A = \pi$:

$$\frac{A}{4} \approx 0.785 > \frac{2}{3} \approx 0.667$$

- Method 2 needs ≈ 3.4 times as many samples for the same variance, since

$$\frac{\text{var}[g(X, Y)]}{\text{var}[f(X)]} \approx 3.4$$

$$\text{var} \left[\hat{\mathbb{E}}_N \right] = \frac{\text{var} [f(X)]}{N}$$

- Expected error (standard deviation) drops with $\mathcal{O}(1/\sqrt{N})$

Is this a “good” estimator ?

$$\text{var} \left[\hat{\mathbb{E}}_N \right] = \frac{\text{var} [f(X)]}{N}$$

- Depends on how we look at it ... 🤔
- Asymptotically, MC converges at least as fast as any other unbiased estimator 😍
 - Follows from the Cramér-Rao bound
- Assume $\mathcal{X} = \mathbb{R}^D$. D does not “directly” enter in $\text{var} \left[\hat{\mathbb{E}}_N \right]$ 😮
 - But $\text{var} [f(X)]$ may depend on D !
 - e.g., $p_X(x) = \mathcal{N}(x; 0, \sigma^2 I)$, $f(x) = \prod_{i=1}^D x_i \Rightarrow \text{var} [f(X)] = (\sigma^2)^D \in \mathcal{O}(\exp(D))$
 - But it's often independent: e.g., $p_X(x) = \mathcal{N}(x; \mu, \Sigma)$, $f(x) = x_i \Rightarrow \text{var} [f(X)] = \Sigma_{ii} \in \mathcal{O}(1)$
 - Check out [Nicola Branchini's blog](#) for details

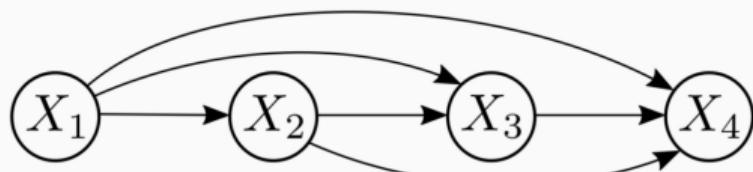
- Bottom line: MC often scales favorably to **high dimensions** ❤️

But was estimating π with Monte Carlo a good idea ?

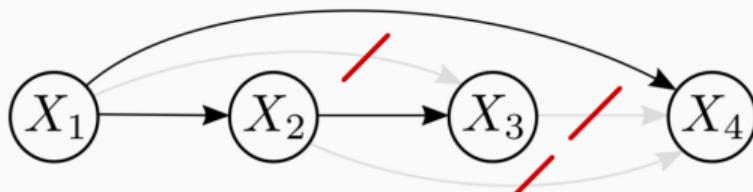
- No! In **low dimensions** ($D < 10$), much more efficient **deterministic** algorithms for **numerical integration** exist!
 - Convergence often exponentially fast in N for “nice” functions f !

BAYESIAN NETWORKS

- $p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2 | x_1)p(x_3 | x_2, x_1)p(x_4 | x_3, x_2, x_1)$



- Assume $X_3 \perp\!\!\!\perp X_1 | X_2$, and $X_4 \perp\!\!\!\perp X_2 | X_1$, and $X_4 \perp\!\!\!\perp X_3 | X_1$
 - Then, $p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2 | x_1)p(x_3 | x_2)p(x_4 | x_1)$



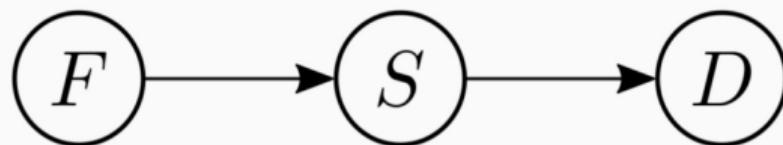
- \mathcal{G} is a **directed acyclic graph** (DAG) over random variables X_1, \dots, X_N
- The Bayesian Network **defines a joint distribution**:

$$p(x_1, \dots, x_N) = \prod_{i=1}^N p(x_i | \text{pa}_i)$$

where pa_i are the **parents** of node x_i .

Assume you have a smoke detector in your apartment

- Each day, there is a 0.01 probability of fire (F) 🔥
- If there is fire, there is smoke (S) with probability 0.995 ☁
 - Even if there is no fire, you might have smoke in your apartment with probability 0.02
- The smoke detector detects smoke with probability 0.98 (D) 🔴
 - It has a false alarm probability of 0.015





f	$p(f)$
0	0.99
1	0.01

s	$p(s f)$	f
0	0.98	0
1	0.02	0
0	0.005	1
1	0.995	1

d	$p(d s)$	s
0	0.985	0
1	0.015	0
0	0.02	1
1	0.98	1

f	s	d	$p(f, s, d)$
0	0	0	$0.99 \times 0.98 \times 0.985$
0	0	1	$0.99 \times 0.98 \times 0.015$
0	1	0	$0.99 \times 0.02 \times 0.02$
0	1	1	$0.99 \times 0.02 \times 0.98$
1	0	0	$0.01 \times 0.005 \times 0.985$
1	0	1	$0.01 \times 0.005 \times 0.015$
1	1	0	$0.01 \times 0.995 \times 0.02$
1	1	1	$0.01 \times 0.995 \times 0.98$



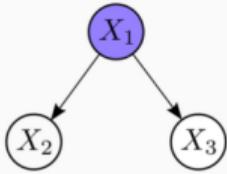
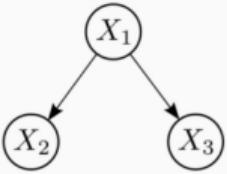
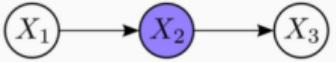
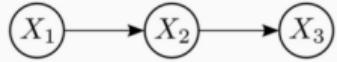
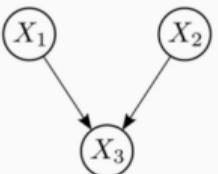
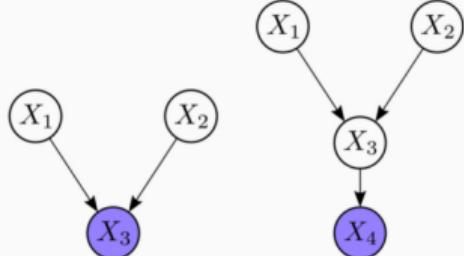
f	s	d	$p(f, s, d)$
0	0	0	0.955647
0	0	1	0.0145530
0	1	0	0.000396
0	1	1	0.019404
1	0	0	0.00004925
1	0	1	0.00000075
1	1	0	0.000199
1	1	1	0.009751

- Assume we are given a Bayesian Network and ask:

$$X \perp\!\!\!\perp Y | Z$$

for RVs X, Y and set of RVs Z .

- If X, Y are directly connected \implies they are dependent
- If they are not directly connected, things are more tricky ... 🤔
- If X and Y are **d-separated** by Z , then $X \perp\!\!\!\perp Y | Z$ in the Bayesian Network distribution
 - No matter how the parameters of the conditional distributions look like !

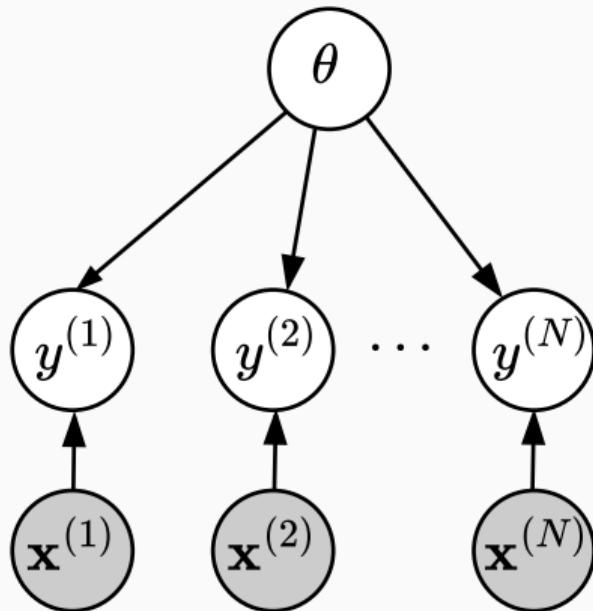
blocked	unblocked
fork	
	
chain	
	
collider	
	

- Given a Bayesian Network over RVs $X = \{X_1, \dots, X_N\}$
- Let $X, Y \in X$ and $Z \subseteq X$
- A **path** (with arcs of either direction) between X and Y is **blocked** if
 - there is a $Z \in Z$ on the path that is a **fork** or **chain** (w.r.t. the path)
 - there is a Z on the path that is a **collider** (w.r.t. the path) and neither Z is in Z , nor any of its **descendants** is in Z
- X and Y are **d-separated** by Z if all paths between X and Y are **blocked** by Z

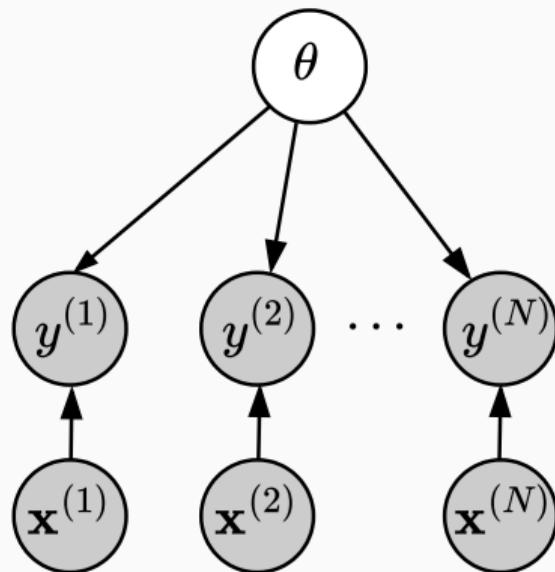
- Recall last week's setup:
- Given $\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})\}$, with $\mathbf{x}^{(i)} \in \mathbb{R}^d, y \in \mathbb{R}$
- Let $\mathbf{X} := \begin{bmatrix} \phi(\mathbf{x}^{(1)}) \\ \vdots \\ \phi(\mathbf{x}^{(N)}) \end{bmatrix}$, and $\mathbf{y} := (y^{(1)}, \dots, y^{(N)})^\top \in \mathbb{R}^N$.
- Model consists of **prior** $p(\theta) = \mathcal{N}(\theta; \mu, \Sigma)$ and **likelihood** $p(\mathbf{y} | \theta, \mathbf{X}) = \mathcal{N}(\mathbf{y}; \mathbf{X}\theta, \sigma^2 I)$

Let's draw this as a Bayesian Network 

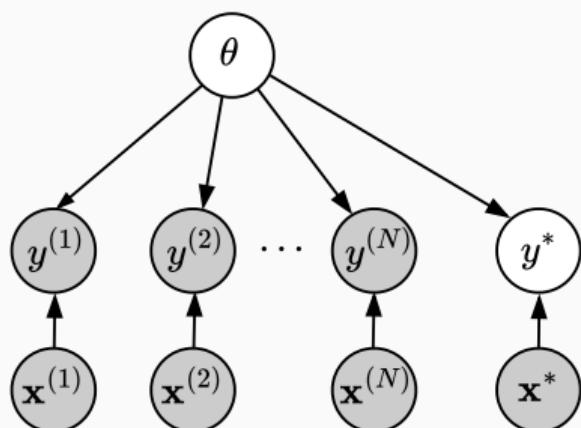
- Model consists of **prior** $p(\theta) = \mathcal{N}(\theta; \mu, \Sigma)$ and **likelihood** $p(y | \theta, X) = \mathcal{N}(y; X\theta, \sigma^2 I)$



- We then also **conditioned** on y , i.e., computed the **posterior** $p(\theta | X, y)$



- Given a test point \mathbf{x}^* , we then wanted to predict y^* , i.e., $p(y^* | \mathbf{X}, \mathbf{y}, \mathbf{x}^*)$



$$\begin{aligned} p(y^* | \mathbf{X}, \mathbf{y}, \mathbf{x}^*) &= \int p(y^* | \mathbf{X}, \mathbf{y}, \mathbf{x}^*, \theta) p(\theta | \mathbf{X}, \mathbf{y}, \mathbf{x}^*) d\theta \\ &= \underbrace{\int p(y^* | \mathbf{x}^*, \theta)}_{\text{Likelihood}} \underbrace{p(\theta | \mathbf{X}, \mathbf{y})}_{\text{Posterior}} d\theta \end{aligned}$$

QUIZ TIME!
fbr.io/pdmp5

