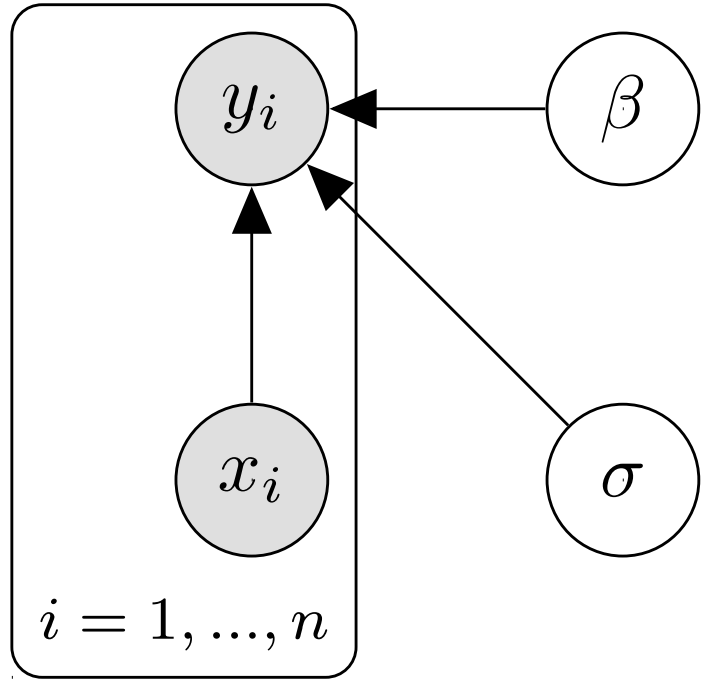
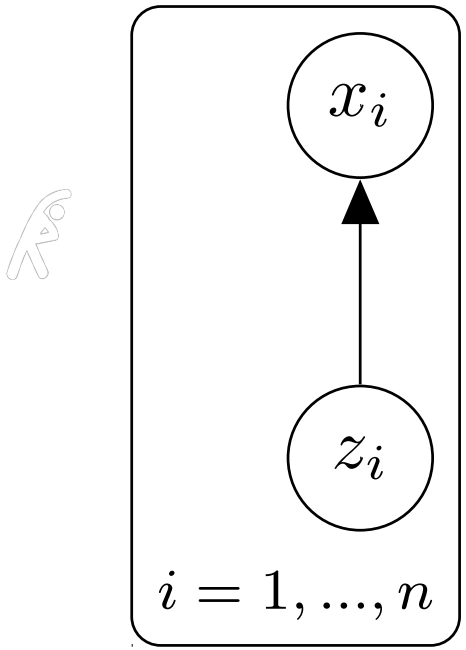
Some concrete examples with 2 nodes: linear regression



$$p(y_i, \beta, \sigma | x_i) = p(\beta, \sigma)\mathcal{N}(y_i | x_i^T \beta, \sigma^2)$$

Sometimes, we make observed nodes in the dataset in grey to discriminate them from unobserved ones

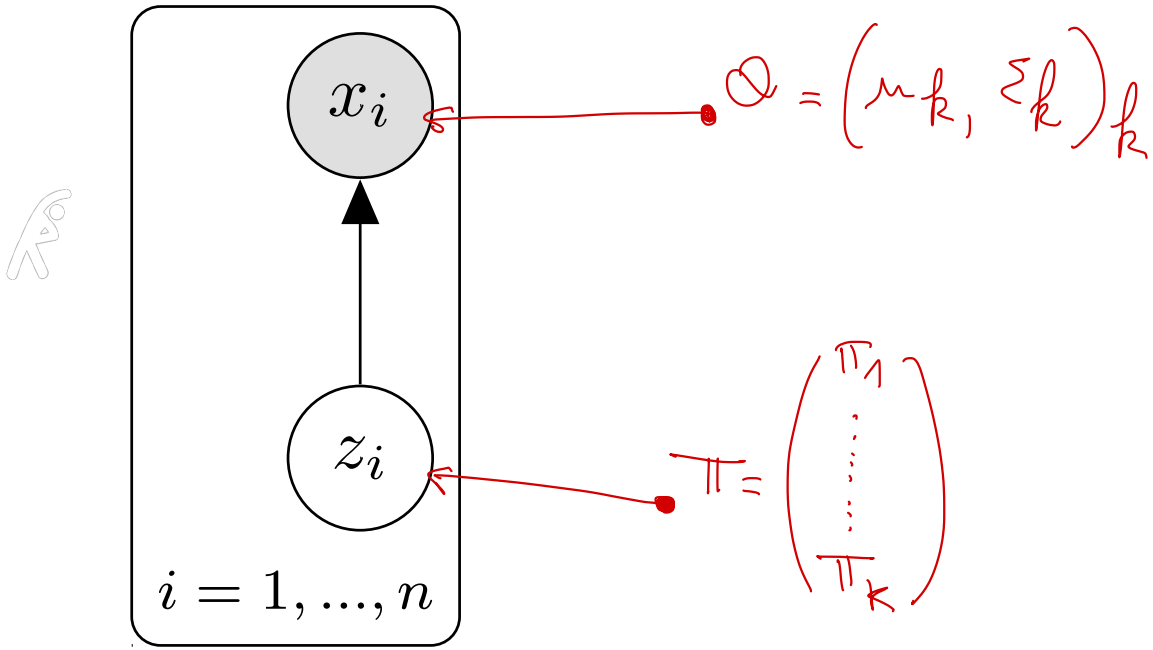
Some concrete examples with 2 nodes: GMM



$$p_{\boldsymbol{\mu},\boldsymbol{\sigma}}(x_i|z_i) = \mathcal{N}(x_i|\mu_{z_i}, \sigma^2_{z_i})$$

Which node should we grey?

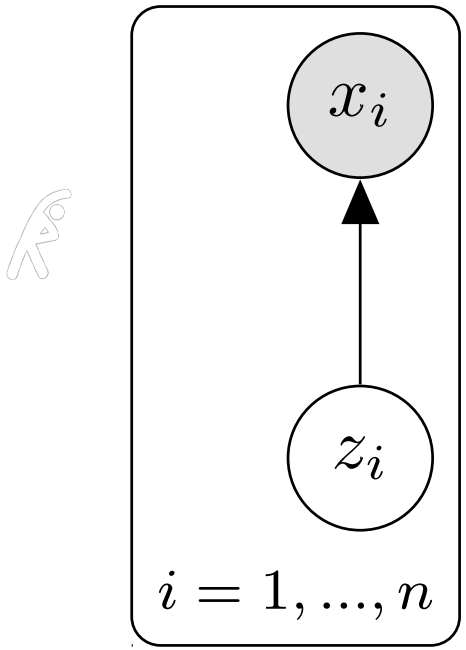
Some concrete examples with 2 nodes: GMM



$$p_{\boldsymbol{\mu},\boldsymbol{\sigma}}(x_i|z_i) = \mathcal{N}(x_i|\mu_{z_i}, \sigma^2_{z_i})$$

Which node should we grey?

Some concrete examples with 2 nodes: GMM



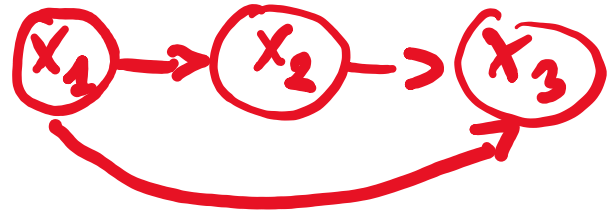
$$p_{\boldsymbol{\mu},\boldsymbol{\sigma}}(x_i|z_i) = \mathcal{N}(x_i|\mu_{z_i}, \sigma^2_{z_i})$$

Which node should we grey?

Let's look at all the possible graphs with 3 nodes

- It gets a bit trickier! Again, we have the trivial graph that corresponds to independence…

- …and the complete graph in which any distribution factorises:



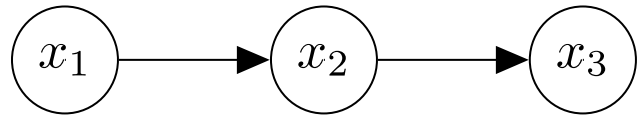
- **But now there are also other graphs!**

Let's look at all the possible graphs with 3 nodes

- The **Markov chain** corresponds to models with limited memory: if you remember x_2, there is no use in remembering x_1 to predict x_3. This corresponds to the factorisation

$$p(x) = p(x_1)p(x_2|x_1)p(x_3|x_2)$$

and therefore to the graph



- A consequence is that $x_3 \perp\!\!\!\perp x_1|x_2$.

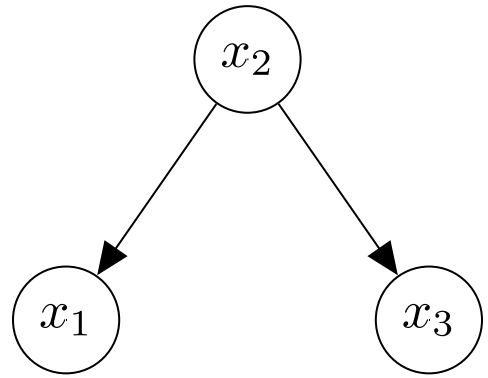
$$p(x_3, x_1 | x_2) = p(x_3 | x_2) \times p(x_1 | x_2)$$
$$= p(x_3 | x_2)\, p(x_1 | x_2)$$

Let's look at all the possible graphs with 3 nodes

- The **latent cause** corresponds to the factorisation $p(x) = p(x_2)p(x_1|x_2)p(x_3|x_2)$

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

and therefore to the graph



$\mathbb{P}(\theta|X)$

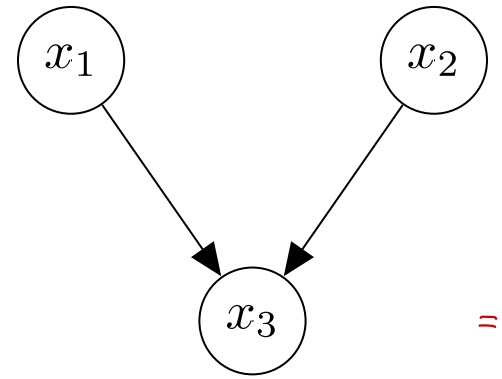
- A consequence is that $x_3 \perp\!\!\!\perp x_1 | x_2$. That's the same than for the Markov chain!

Let's look at all the possible graphs with 3 nodes

- The **v-structure (aka explaining away)** corresponds to the factorisation

$$p(x) = p(x_1)p(x_2)p(x_3|x_1, x_2)$$

and therefore to the graph



$p(x)$

$p(x_1, x_2) = \int p(x_1, x_2, x_3)\, dx_3$

$= \int p(x_3 | x_1, x_2) p(x_1) p(x_2)\, dx_3$

$= p(x_1) p(x_2) \underbrace{\int p(x_3 | x_1, x_2)\, dx_3}_{=1}$

$= p(x_1)\, p(x_2)$

- A consequence is that $x_1 \perp\!\!\!\perp x_2$, like for the trivial graph.

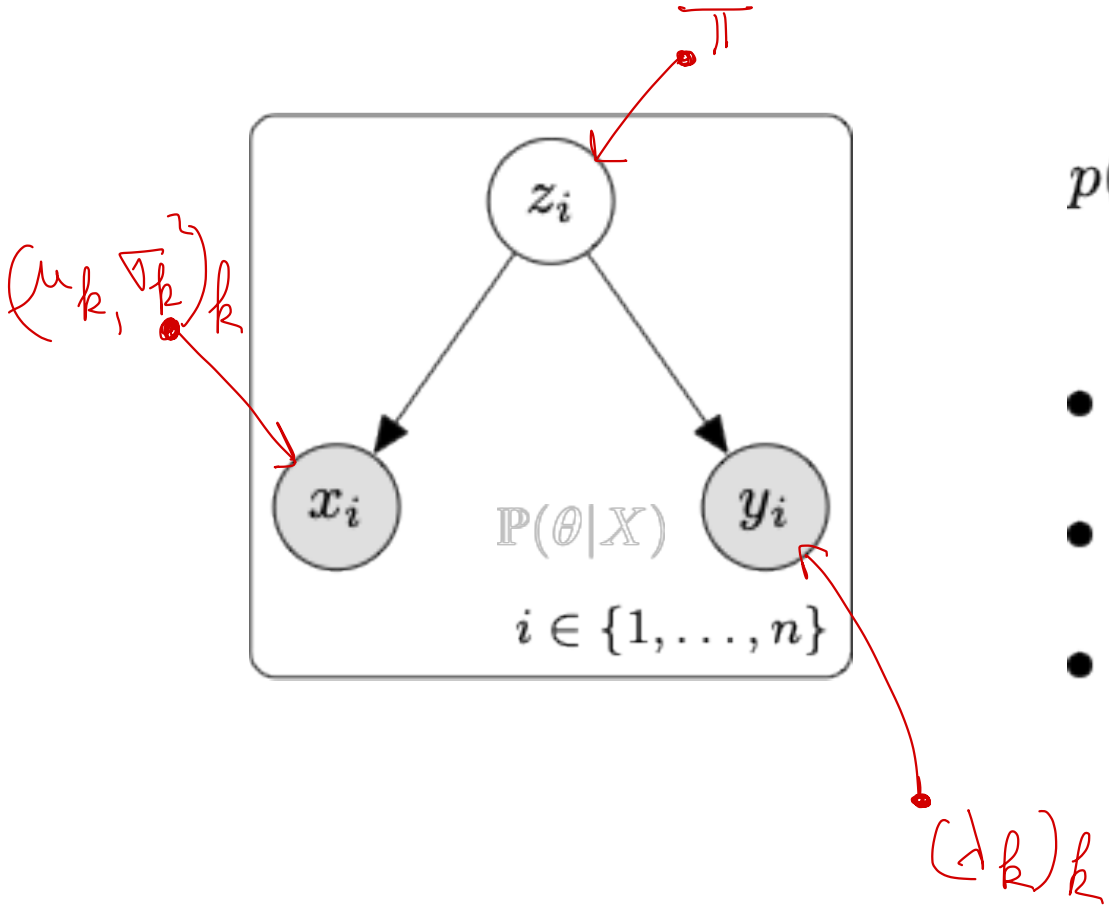
Examples of graphs with 3 nodes

- **Latent cause: mixed-data clustering:** our data are $(x_i, y_i)_{i \leq n}$ with $x_i \in \mathbb{R}$ and $y_i \in \mathbb{N}$

$\mathbb{P}(\theta|X)$

Examples of graphs with 3 nodes

- **Latent cause: mixed-data clustering:** our data are $(x_i, y_i)_{i \leq n}$ with $x_i \in \mathbb{R}$ and $y_i \in \mathbb{N}$



$$p(x, y, z) = \prod_{i=1}^{n} p(z_i) p(x_i|z_i) p(y_i|z_i)$$

- $p(z_i) = \mathcal{M}(z_i|1, \pi) = \text{Categorical}(z_i|\pi)$
- $p(x_i|z_i) = \mathcal{N}(x_i|\mu_{z_i}, \sigma^2_{z_i})$
- $p(y_i|z_i) = \mathcal{P}(y_i|\lambda_{z_i})$

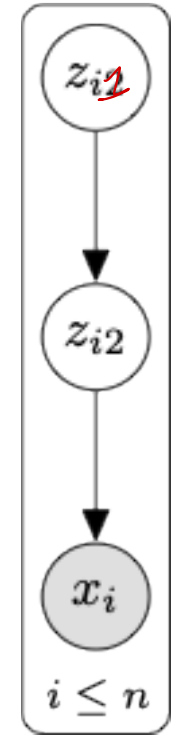
See, e.g. Marbac et al., Variable selection for mixed data clustering: Application in Human Population Genomics, J Classif, 2020

Examples of graphs with 3 nodes

- **Markov chain: Hierarchical Variational Autoencoder (HVAE)**

- We want to **model/explain high-dimensional data using a hierarchy of two latent variables**, tied together by neural nets.

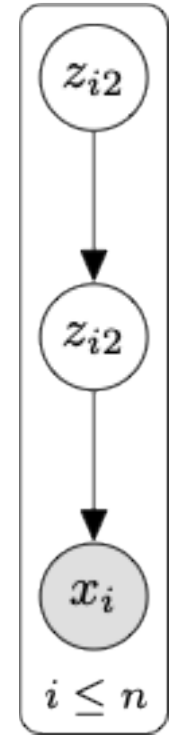
$\mathbb{P}(\theta|X)$



Examples of graphs with 3 nodes

- **Markov chain: Hierarchical Variational Autoencoder (HVAE)**
- We want to **model/explain high-dimensional data using a hierarchy of two latent variables**, tied together by neural nets.
- There are two DAGs that fit different purpose here:

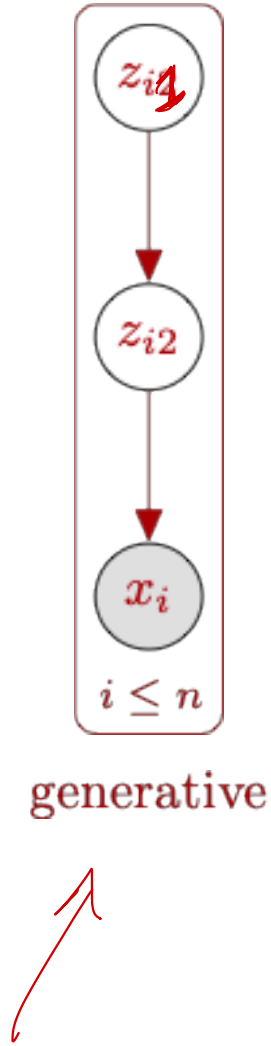
$\mathbb{P}(\theta|X)$



Examples of graphs with 3 nodes

- **Markov chain: Hierarchical Variational Autoencoder (HVAE)**
- We want to **model/explain high-dimensional data using a hierarchy of two latent variables**, tied together by neural nets.
- There are two DAGs that fit different purpose here:
  - one for the **generative model**

$$\mathbb{P}(\theta|X)$$



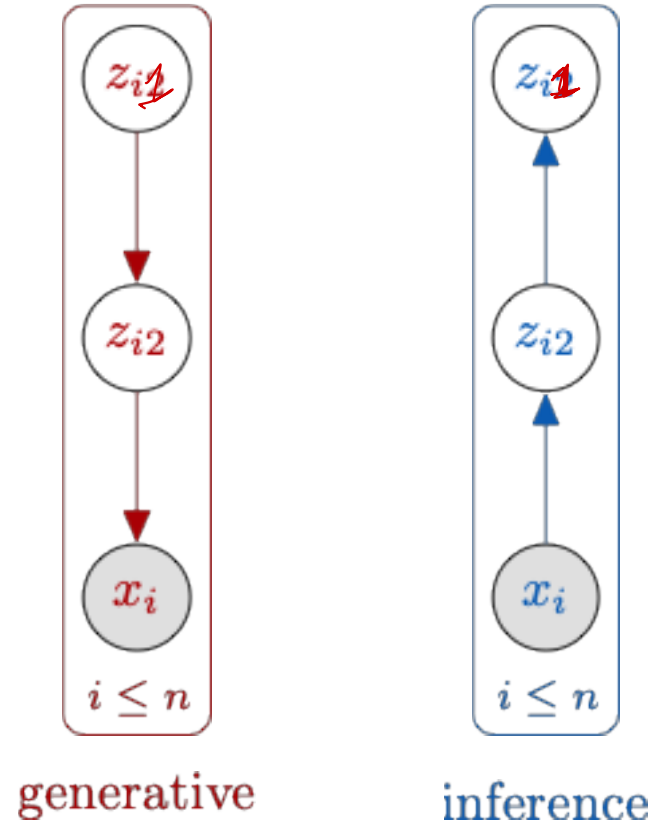
generative

Examples of graphs with 3 nodes

- **Markov chain: Hierarchical Variational Autoencoder (HVAE)**

- We want to **model/explain high-dimensional data using a hierarchy of two latent variables**, tied together by neural nets.

- There are two DAGs that fit different purpose here:
  - one for the **generative model**
  - and one for training the model (aka **inference network**). We'll talk more about this when we discuss VAEs.

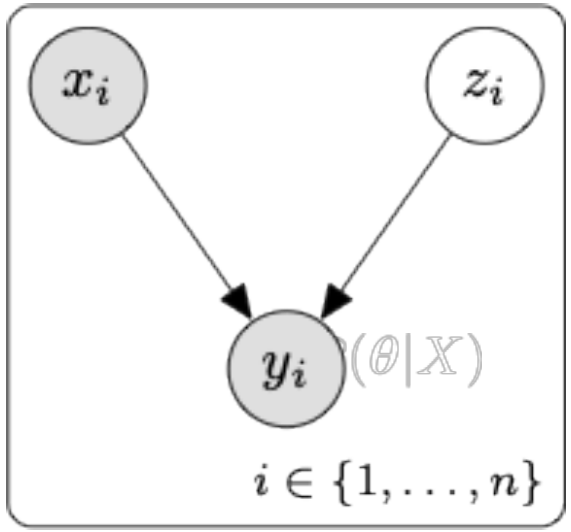


See, e.g. Sønderby et al., Ladder Variational Autoencoders, NeurIPS, 2016

Examples of graphs with 3 nodes

NN : $\mu_{\theta_1}(x_i)$
$\mu_{\theta_2}(x_i)$
$\mu_{\theta_K}(x_i)$

- **Explaining away: mixture of experts with constant gating:**
  - regression task, with data $(x_i, y_i)_{i \leq n}$ and $y_i \in \mathbb{R}$



$$p(x, y, z) = \prod_{i=1}^{n} p(x_i)p(z_i)p(y_i|x_i, z_i)$$

- $p(z_i) = \mathcal{M}(z_i|1, \pi)$
- $p(y_i|x_i, z_i) = \mathcal{N}(y_i|\mu_{\theta_{z_i}}(x_i), \sigma^2_{\theta_z}(x_i))$
- $\mu_{\theta_1}, \ldots \mu_{\theta_K}, \sigma_{\theta_1}, \ldots \sigma_{\theta_K}$ are neural nets called *experts*

The gating is ”constant” because x and z are not connected.

3

Further theory

Unicity of DAG factorisation?

Let's go back to the definition…

- A graphical model $\mathcal{L}(G)$ is a family of probability distributions that can be factorised as

$$p(x) = \prod_{i=1}^{d} p(x_i | x_{\mathrm{pa}_i})$$


- In maths, when we have factorisation results, a natural question is: **is the factorisation unique?**

- For instance, a classical arithmetics theorems asserts that **any natural number can be written as a product of prime numbers in a unique way**.

- Here, do you think that a decomposition like $p(x) = \prod_{i=1}^{d} p(x_i | x_{\mathrm{pa}_i})$, when it exists, is unique in a sense?

DAG factorisations are unique

Theorem (unicity of the DAG factorisation). *Let $p(x)$ be a distribution such that there exists some nonnegative functions $f_1, ..., f_d \geq 0$ such that, for all x,*

$$p(x) = \prod_{i=1}^{d} f_i(x_i, x_{pa_i})$$

and, for all $i \in \{1, ..., d\}$ and for all x_i, x_{pa_i},

$$\int_{x_i} f_i(x_i, x_{pa_i}) dx_i = 1.$$

Then, for all $i \in \{1, ..., d\}$,

$$f_i(x_i, x_{pa_i}) = p(x_i | x_{pa_i}).$$

DAG factorisations are unique

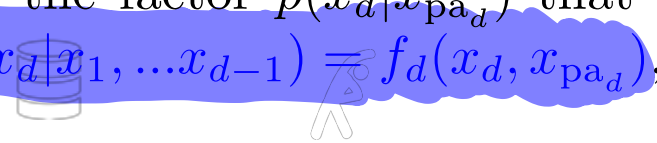
Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

$\mathbb{P}(\theta|X)$

DAG factorisations are unique

Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

We start with the factor $p(x_d|x_{\mathrm{pa}_d})$ that corresponds to this leaf. A first remark is that $p(x_d|x_1, ...x_{d-1}) = f_d(x_d, x_{\mathrm{pa}_d})$, indeed



$$\mathbb{P}(\theta|X)$$

DAG factorisations are unique

Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

We start with the factor $p(x_d|x_{\mathrm{pa}_d})$ that corresponds to this leaf. A first remark is that $p(x_d|x_1, ...x_{d-1}) = f_d(x_d, x_{\mathrm{pa}_d})$, indeed

$$p(x_d|x_1, ...x_{d-1}) = \frac{p(x_1, ...x_{d-1}, x_d)}{p(x_1, ...x_{d-1})} = f_d(x_d, x_{\mathrm{pa}_d}) \frac{\prod_{i=1}^{d-1} f_i(x_i, x_{\mathrm{pa}_i})}{p(x_1, ...x_{d-1})},$$

$\mathbb{P}(\theta|X)$

DAG factorisations are unique

Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

We start with the factor $p(x_d|x_{\mathrm{pa}_d})$ that corresponds to this leaf. A first remark is that $p(x_d|x_1, ...x_{d-1}) = f_d(x_d, x_{\mathrm{pa}_d})$, indeed

$$p(x_d|x_1, ...x_{d-1}) = \frac{p(x_1, ...x_{d-1}, x_d)}{p(x_1, ...x_{d-1})} = f_d(x_d, x_{\mathrm{pa}_d}) \frac{\prod_{i=1}^{d-1} f_i(x_i, x_{\mathrm{pa}_i})}{p(x_1, ...x_{d-1})},$$

does not depend on x_d because x_d is the parent of no-one!

DAG factorisations are unique

Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

We start with the factor $p(x_d|x_{\text{pa}_d})$ that corresponds to this leaf. A first remark is that $p(x_d|x_1, ...x_{d-1}) = f_d(x_d, x_{\text{pa}_d})$, indeed

$$p(x_d|x_1, ...x_{d-1}) = \frac{p(x_1, ...x_{d-1}, x_d)}{p(x_1, ...x_{d-1})} = f_d(x_d, x_{\text{pa}_d}) \frac{\prod_{i=1}^{d-1} f_i(x_i, x_{\text{pa}_i})}{p(x_1, ...x_{d-1})},$$

does not depend on x_d because x_d is the parent of no-one!

which means that $p(x_d|x_1, ...x_{d-1}) \propto f_d(x_d, x_{\text{pa}_d})$, and since both $p(x_d|x_1, ...x_{d-1})$ and $f_d(x_d, x_{\text{pa}_d})$ are density functions, we have $p(x_d|x_1, ...x_{d-1}) = f_d(x_d, x_{\text{pa}_d})$.

DAG factorisations are unique

Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

We start with the factor $p(x_d|x_{\text{pa}_d})$ that corresponds to this leaf. A first remark is that $p(x_d|x_1, ...x_{d-1}) = f_d(x_d, x_{\text{pa}_d})$, indeed

$$p(x_d|x_1, ...x_{d-1}) = \frac{p(x_1, ...x_{d-1}, x_d)}{p(x_1, ...x_{d-1})} = f_d(x_d, x_{\text{pa}_d}) \frac{\prod_{i=1}^{d-1} f_i(x_i, x_{\text{pa}_i})}{p(x_1, ...x_{d-1})},$$

does not depend on x_d because x_d is the parent of no-one!

which means that $p(x_d|x_1, ...x_{d-1}) \propto f_d(x_d, x_{\text{pa}_d})$, and since both $p(x_d|x_1, ...x_{d-1})$ and $f_d(x_d, x_{\text{pa}_d})$ are density functions, we have $p(x_d|x_1, ...x_{d-1}) = f_d(x_d, x_{\text{pa}_d})$.

Now, since $p(x_d|x_1, ...x_{d-1}) = f_d(x_d, x_{\text{pa}_d})$, this means that $p(x_d|x_1, ...x_{d-1})$ is just a function of x_d and x_{pa_d}, therefore $p(x_d|x_1, ...x_{d-1}) = p(x_d|x_{\text{pa}_d})$ and we finally get

$$f_d(x_d, x_{\text{pa}_d}) = p(x_d|x_{\text{pa}_d}).$$

DAG factorisations are unique

Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

We started with the factor $p(x_d|x_{\mathrm{pa}_d})$ and showed that $f_d(x_d, x_{\mathrm{pa}_d}) = p(x_d|x_{\mathrm{pa}_d})$. This initialises our proof by induction!

$\mathbb{P}(\theta|X)$

DAG factorisations are unique

Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

We started with the factor $p(x_d|x_{\mathrm{pa}_d})$ and showed that $f_d(x_d, x_{\mathrm{pa}_d}) = p(x_d|x_{\mathrm{pa}_d})$. This initialises our proof by induction!
Now, we will remove the leaf x_d from the graph, leaving us with the distribution $p(x_1, ..., x_{d-1})$. Looking at a ratio from the previous slide, it is clear that

$$p(x_1, ..., x_{d-1}) = \prod_{i=1}^{d-1} f_i(x_i, x_{\mathrm{pa}_i}),$$

so we can redo what we did in the previous slide to show that

$$p(x_{d-1}|x_{\mathrm{pa}_{d-1}}) = f(x_{d-1}|x_{\mathrm{pa}_{d-1}}).$$

DAG factorisations are unique

Proof. We assume, without loss of generality, that the nodes are sorted in topological order. Therefore, the last node x_d has no descendant (it's what we call a **leaf**). The idea will be, starting from this node, to do a proof by induction.

We started with the factor $p(x_d|x_{\mathrm{pa}_d})$ and showed that $f_d(x_d, x_{\mathrm{pa}_d}) = p(x_d|x_{\mathrm{pa}_d})$. This initialises our proof by induction!
Now, we will remove the leaf x_d from the graph, leaving us with the distribution $p(x_1, ..., x_{d-1})$. Looking at a ratio from the previous slide, it is clear that

$$p(x_1, ..., x_{d-1}) = \prod_{i=1}^{d-1} f_i(x_i, x_{\mathrm{pa}_i}),$$

so we can redo what we did in the previous slide to show that

$$p(x_{d-1}|x_{\mathrm{pa}_{d-1}}) = f(x_{d-1}|x_{\mathrm{pa}_{d-1}}).$$

And we can repeat that inductively until we have seen the whole graph, proving finally that $p(x_i|x_{\mathrm{pa}_i}) = f(x_i|x_{\mathrm{pa}_i})$, for all $i \in \{1, ..., d\}$. □

4

Why are directed models useful?

Conditional independence

A few comments on these graphs with 3 nodes

- We saw that, in general, the structure of the graph could be translated into **conditional independence statements.** Sometimes, quite different graphs let to the same conditional independence statements (like the Markov chain and the latent cause).


- **Are there systematic ways of seeing conditional independence by looking at graphs?**

- Yes, and that's one of the big strenghts of graphical models. We will see two today: **conditional independence with the non-descendants**, and **d-separation.**

Conditional independence with the non-descendants

Definition. *The set of non-descendants of of a node i, denoted by $nd(i)$, is the set of nodes that are not descendants of i.*

Theorem. $p \in \mathcal{L}(G) \iff \forall i \in \{1, ..., d\}, \; X_i \perp\!\!\!\perp X_{nd(i)} | X_{pa_i}$

Proof. See e.g. [MVA17, Sec. 4.2.3]. □

$\mathbb{P}(\theta|X)$

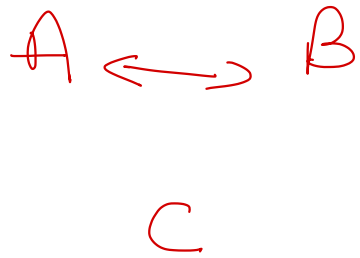
- This generalises the « memoryless » property of Markov chains.

D-separation

- D-separation means **directed separation.**

- It's a general framework for answering questions à la « $X_A \perp\!\!\!\perp X_B | X_C ?$ » where A, B, C are three subsets of the set of nodes.

$\mathbb{P}(\theta|X)$



D-separation

- D-separation means **directed separation.**

- It's a general framework for answering questions à la « $X_A \perp\!\!\!\perp X_B | X_C$? » where A, B, C are three subsets of the set of nodes.

- **D-separation recipe:** we consider all chains between any node in A and any node in B. Any of these chains is said to be **blocked by C** if it includes a node such that either
  - the chain is a v-structure at the node, and neither the node, nor its descendants, are in C
  - the arrows on the chain meet either head-to-tail or tail-to-tail at the node, and the node belongs to C

D-separation

- D-separation means **directed separation.**

- It's a general framework for answering questions à la « $X_A \perp\!\!\!\perp X_B | X_C ?$ » where A, B, C are three subsets of the set of nodes.


- **D-separation recipe:** we consider all chains between any node in A and any node in B. Any of these chains is said to be **blocked by C** if it includes a node such that either
  - the chain is a v-structure at the node, and neither the node, nor its descendants, are in C
  - the arrows on the chain meet either head-to-tail or tail-to-tail at the node, and the node belongs to C

Definition. *We say that A **and** B **are d-separated by** C if all chains between A and B are blocked by C.*

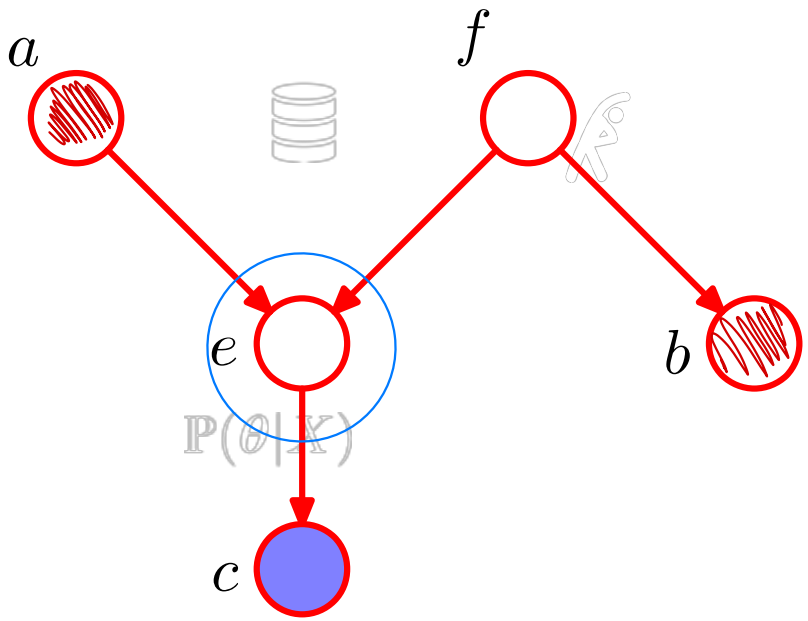
Examples of d-separation

$A = \{a\}$

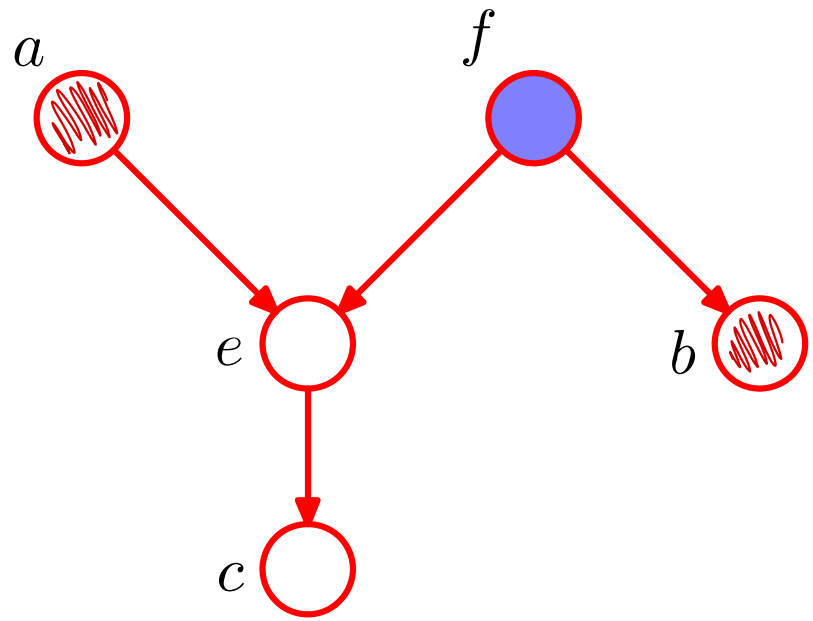
$B = \{b\}$

$C = \{c\}$

- I took this figure from [Bishop, Fig. 8.22]. Does the blue node d-separate a and b?



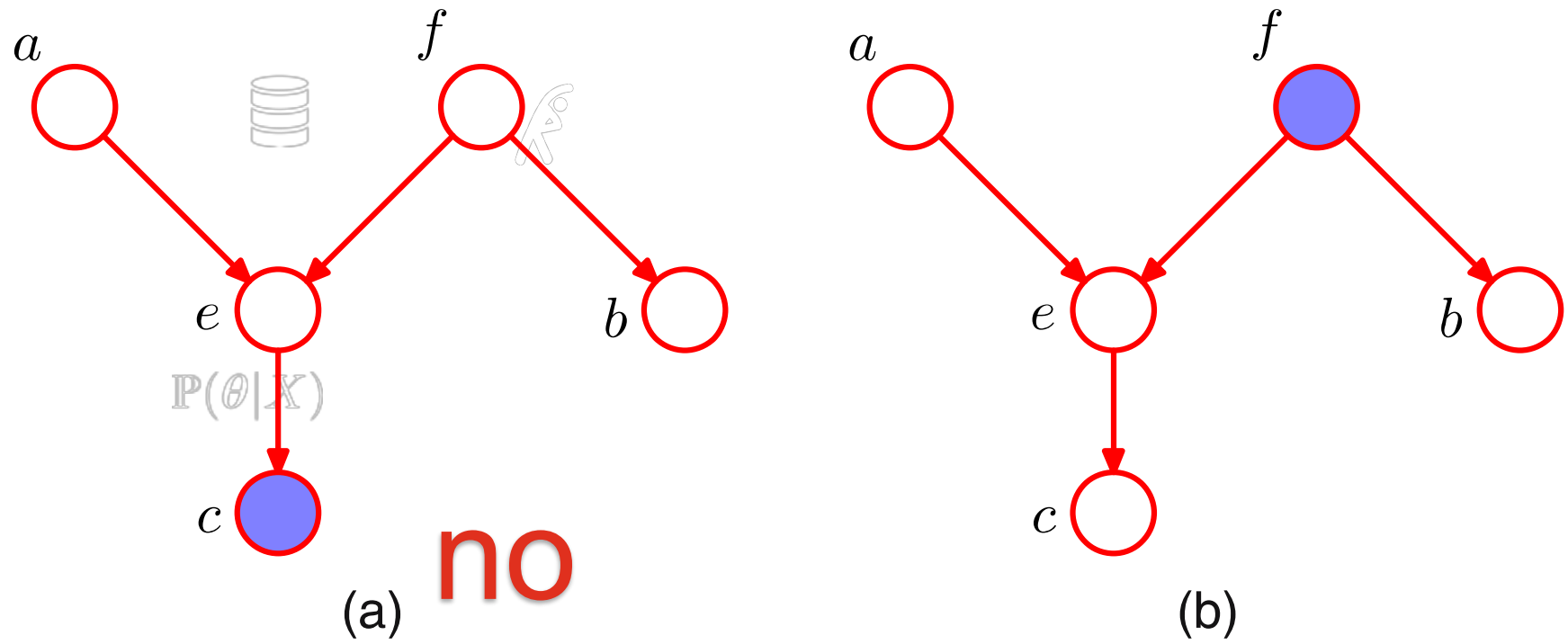
(a)



(b)

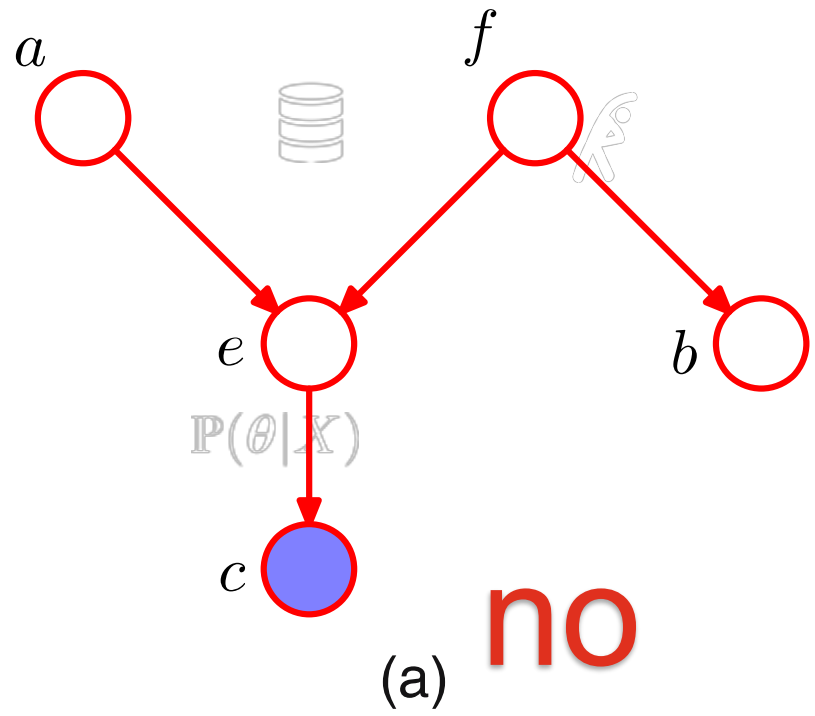
Examples of d-separation

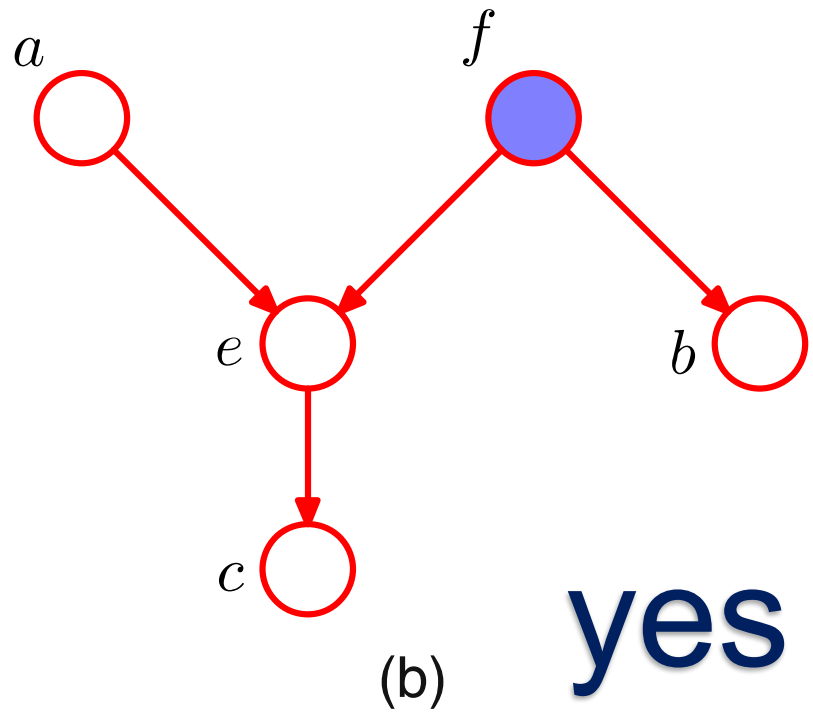
- I took this figure from [Bishop, Fig. 8.22]. Does the blue node d-separate a and b?



Examples of d-separation

- I took this figure from [Bishop, Fig. 8.22]. Does the blue node d-separate a and b?





Properties of d-separation

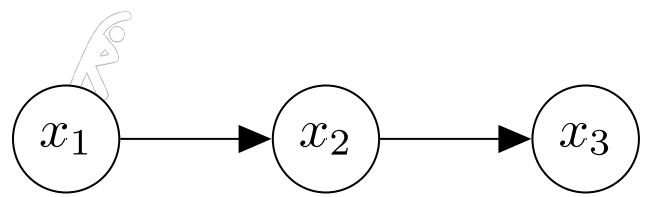
Theorem (soundness of d-separation)**.** *If A and B are d-separated by C and $p \in \mathcal{L}(G)$, then $X_A \perp\!\!\!\perp X_B | X_C$.*

Theorem (completeness of d-separation)**.** *If A and B are not d-separated by C, then there exist $p \in \mathcal{L}(G)$ such that $X_A \not\perp\!\!\!\perp X_B | X_C$.*

For more details, including proofs, see [PGM, Sec. 3.3.2].

Applications of d-separation

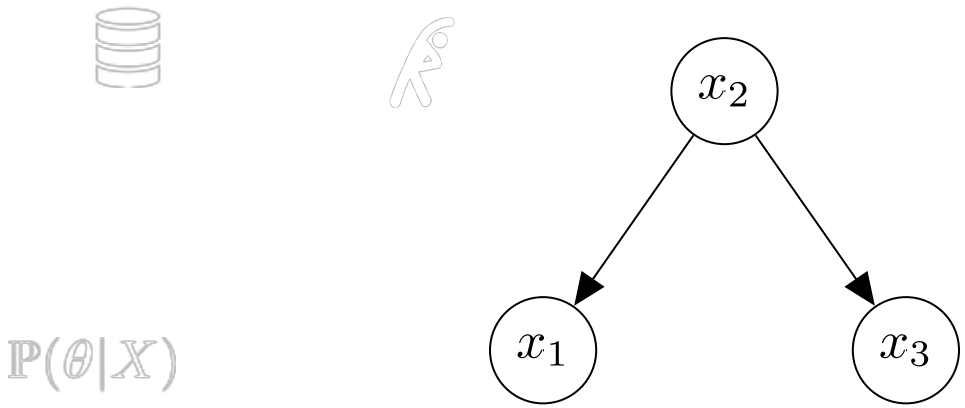
- We previously mentioned conditional independence properties. Try to prove them using d-separation:



$$x_3 \perp\!\!\!\perp x_1 | x_2$$

Applications of d-separation

- We previously mentioned conditional independence properties. Try to prove them using d-separation:



$$x_3 \perp\!\!\!\perp x_1 | x_2$$

5

Why are directed models useful?

Sampling

Why sampling from a graphical model

1. The samples are useful by themselves

$\mathbb{P}(\theta|X)$

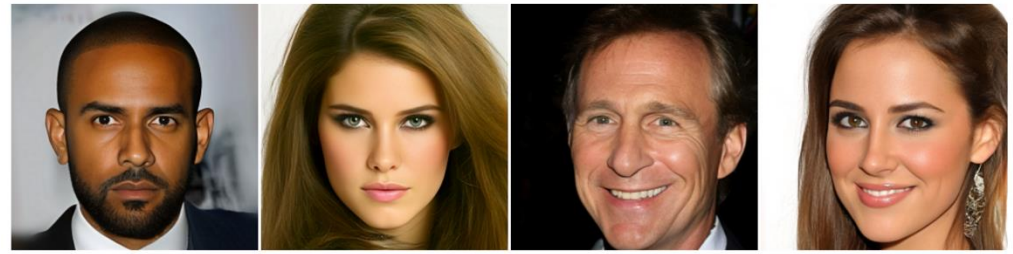
Images generated by a GAN

Brock, Donahue, and Simonyan (ICLR 2019)

2. Samples allow us to appximate integrals using Monte Carlo

Why sampling from a graphical model

1. The samples are useful by themselves



Images generated by a VAE

Vahdat and Kautz (NeurIPS 2020)

$\mathbb{P}(\theta|X)$

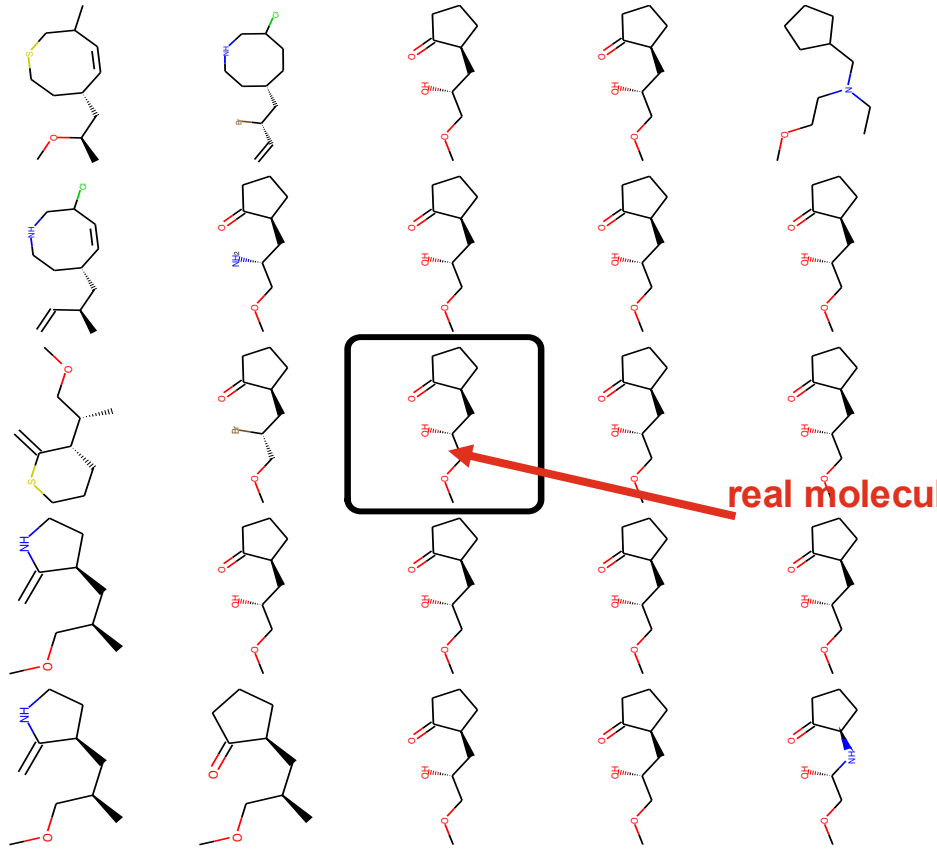
2. Samples allow us to appximate integrals using Monte Carlo

Why sampling from a graphical model

1. The samples are useful by themselves

Kusner, Paige, and Hernández-Lobato (ICML 2017)

$$\mathbb{P}(\theta|X)$$



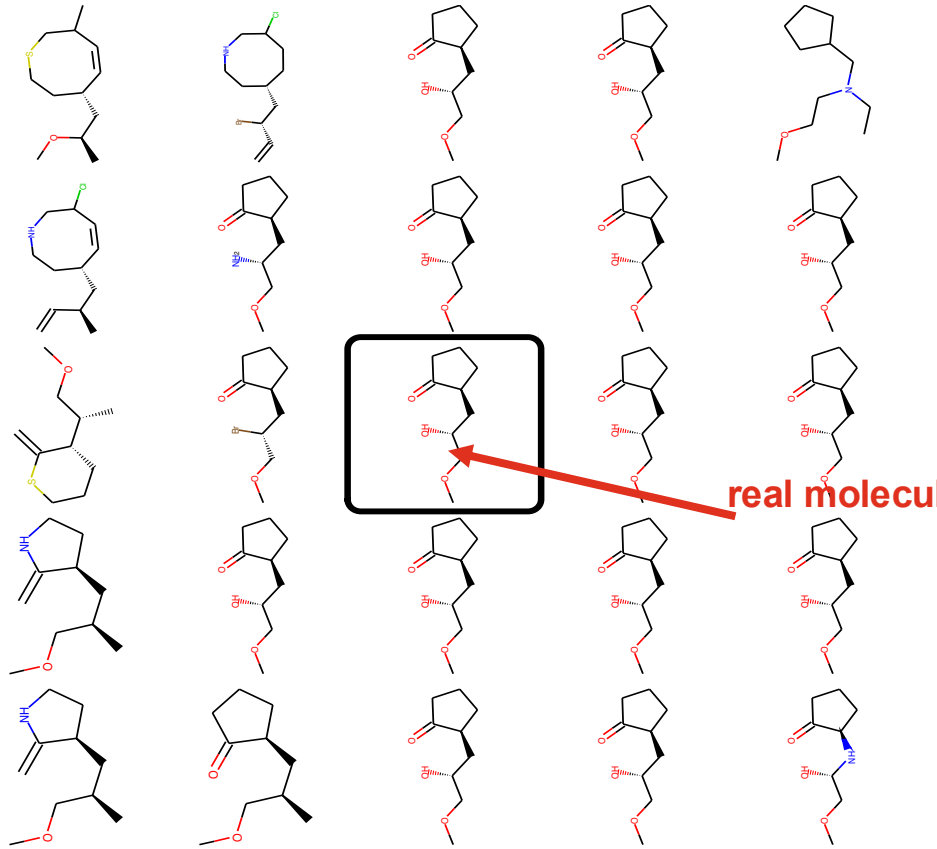
2. Samples allow us to appximate integrals using Monte Carlo

Why sampling from a graphical model

1. The samples are useful by themselves

Kusner, Paige, and Hernández-Lobato (ICML 2017)

$\mathbb{P}(\theta|X)$



2. Samples allow us to appximate integrals using Monte Carlo

$$\int f(x)p(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$

Monte Carlo basics

- Until now, we have written integrals as sums that we can compute exhaustively, but this is quite rarely the case.

- When this integral is an expected value over something we know how to sample from, we can use **simple Monte Carlo**

Algorithm 1 Simple Monte Carlo

1: Draw $X^{(1)}, ..., X^{(n)} \overset{i.i.d.}{\sim} p$
2: $\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} f(X^{(i)})$

- Why are we expecting this to work?

Monte Carlo basics

Algorithm 1 Simple Monte Carlo

1: Draw $X^{(1)}, ..., X^{(n)} \overset{i.i.d.}{\sim} p$
2: $\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} f(X^{(i)})$


Proposition (Law of Large Numbers)**.**

$$\hat{\mu} \xrightarrow{a.s.} \mu \quad iff \quad \mu \ is \ finite$$

Proposition (Central Limit Theorem)**.** *If* $\mathbb{V}ar(f(X)) = \sigma^2 < \infty$, *then*

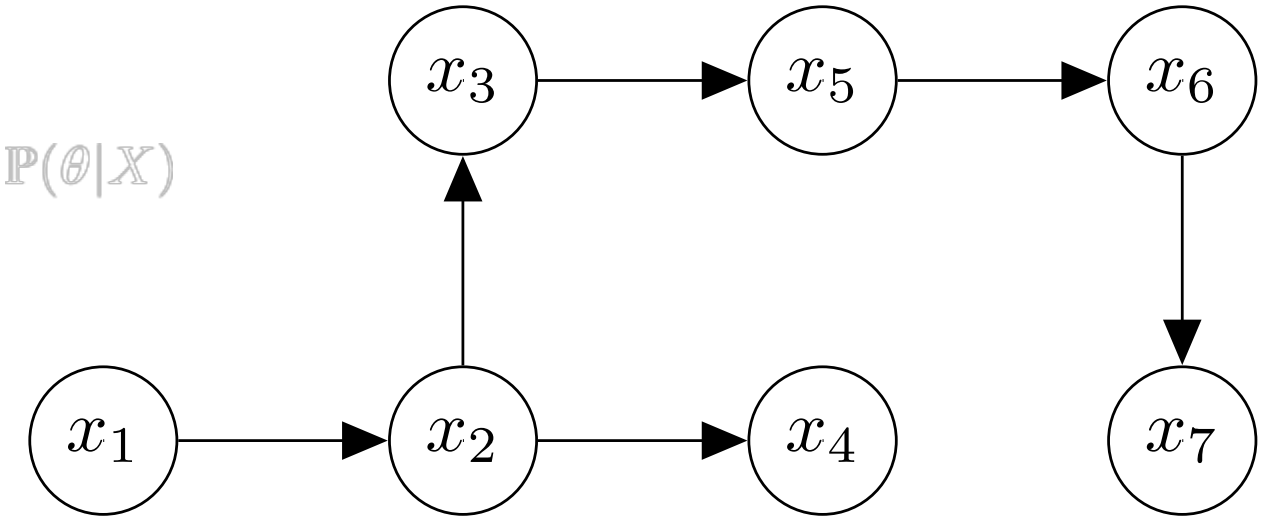
$$\sqrt{n}(\hat{\mu} - \mu) \xrightarrow{\mathcal{D}} \mathcal{N}(0, \sigma^2)$$

How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

We follow the ordering, sampling one descendant after the other…

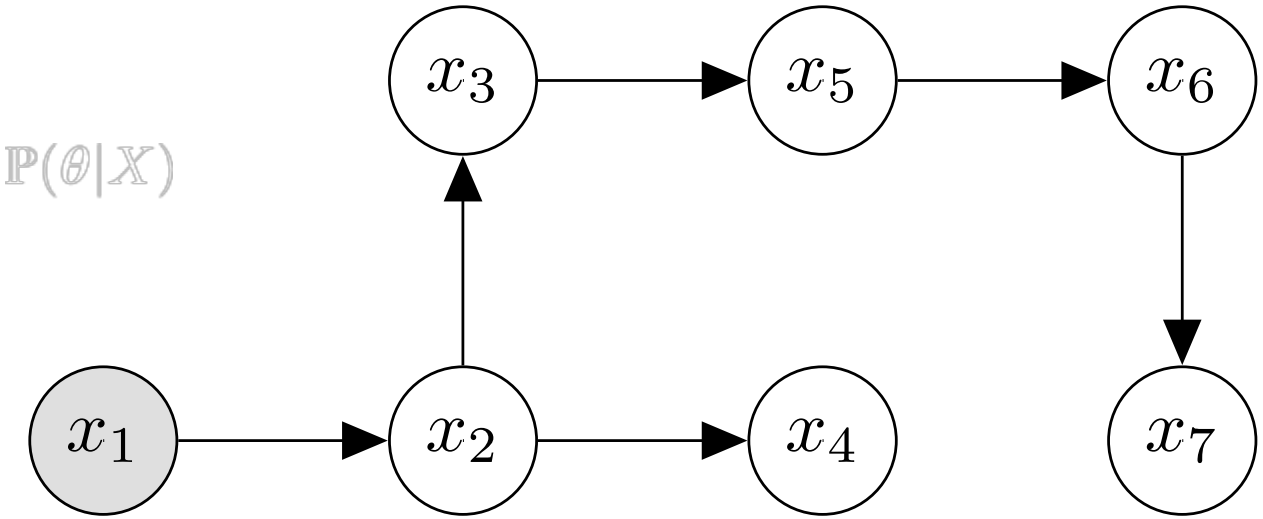


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

We follow the ordering, sampling one descendant after the other…

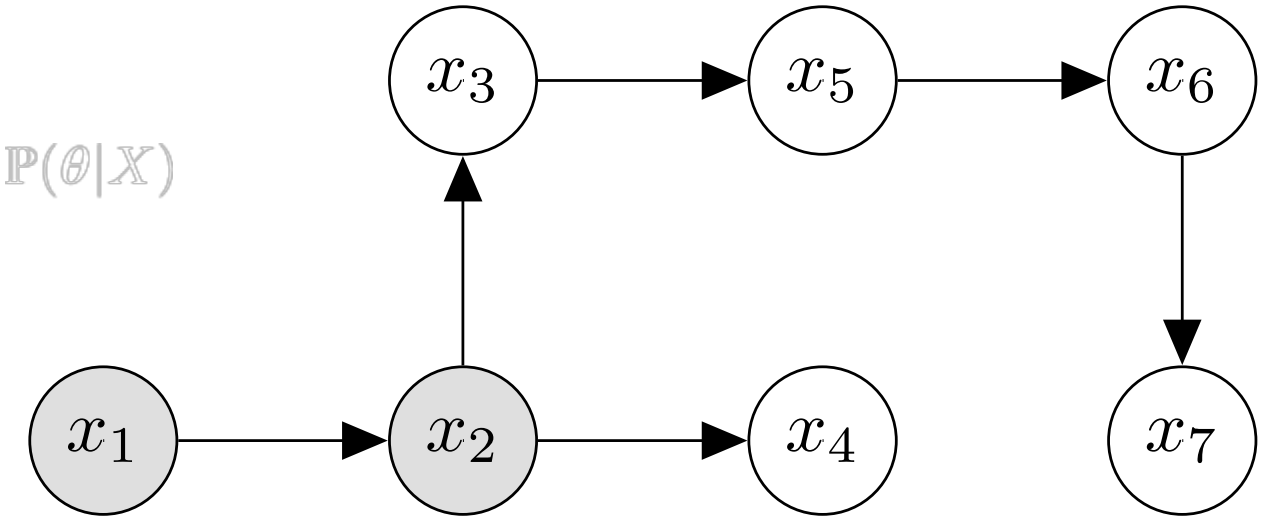


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

We follow the ordering, sampling one descendant after the other…

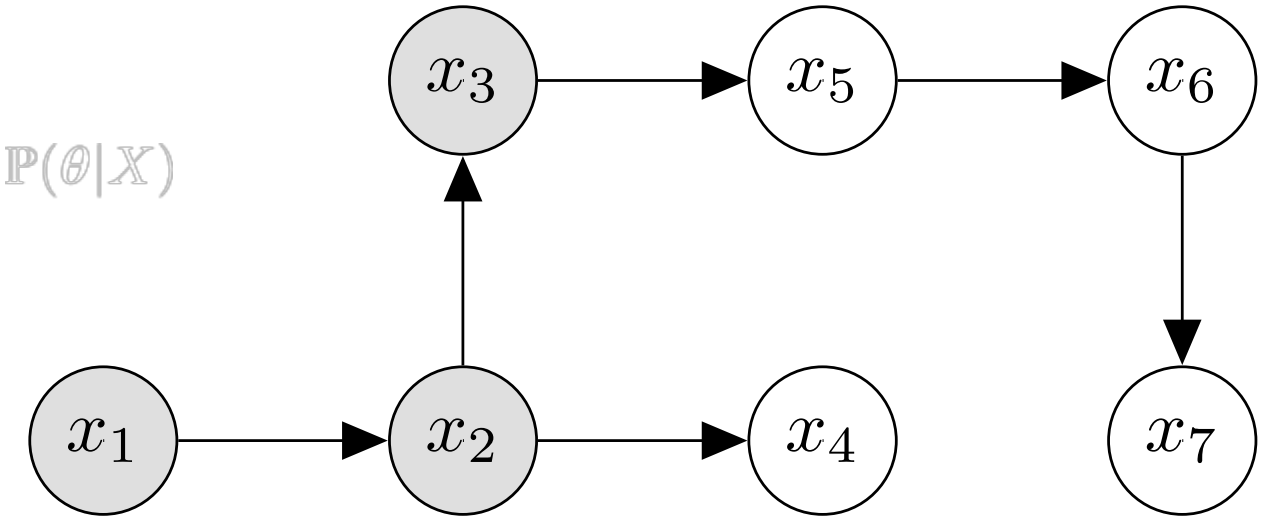


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

We follow the ordering, sampling one descendant after the other…

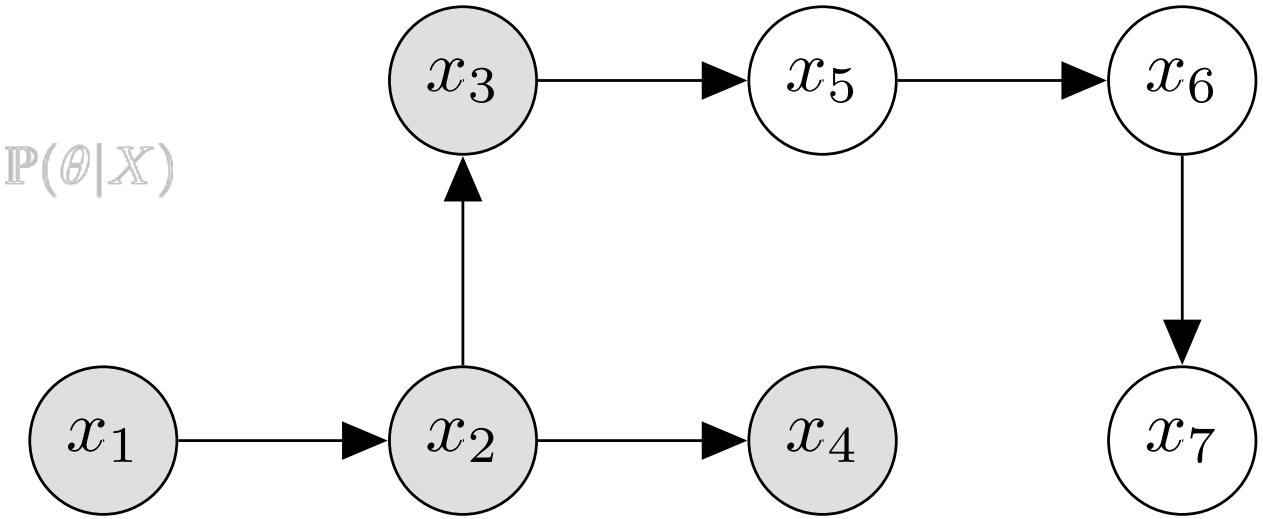


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

We follow the ordering, sampling one descendant after the other…

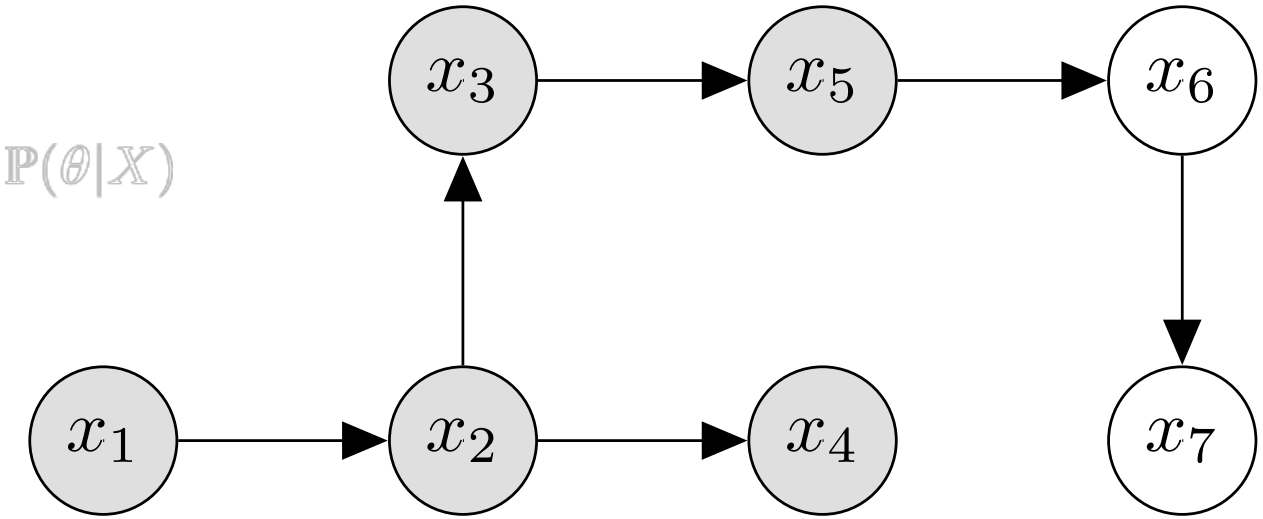


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

We follow the ordering, sampling one descendant after the other…

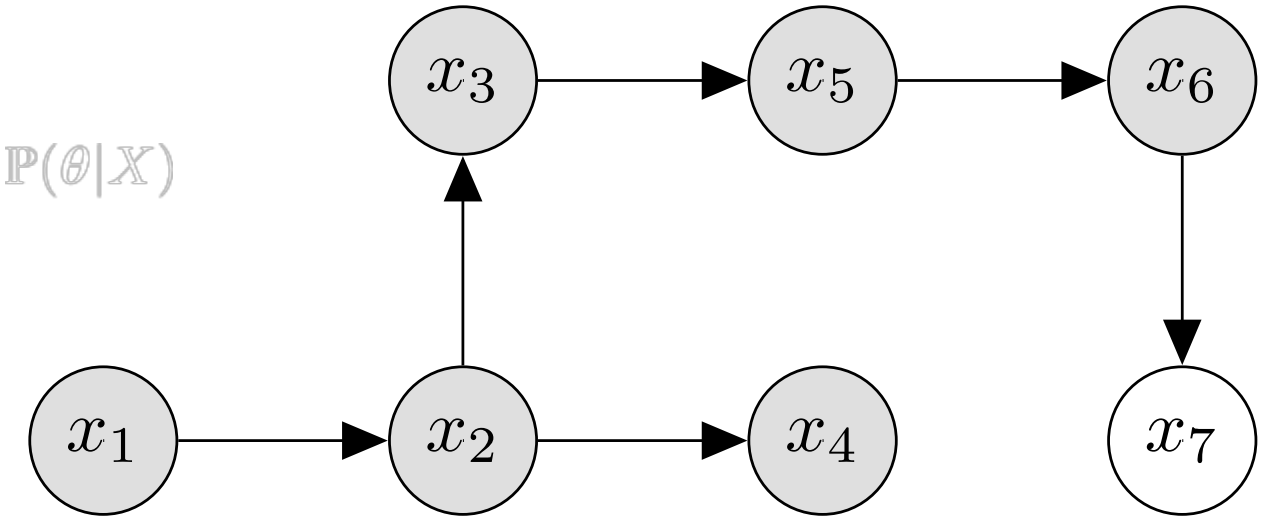


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

We follow the ordering, sampling one descendant after the other…

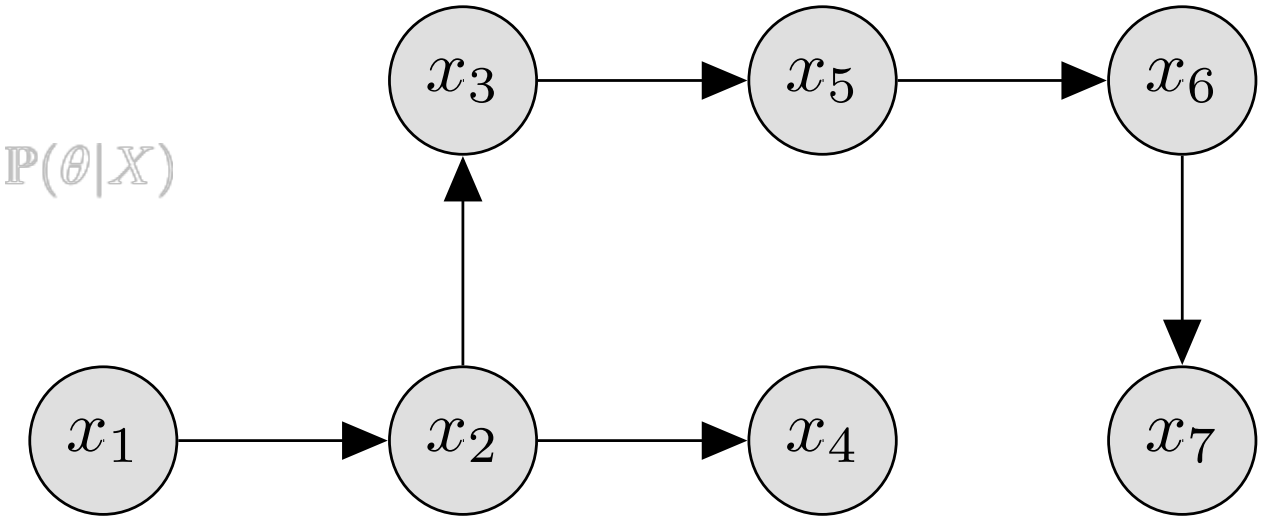


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

We follow the ordering, sampling one descendant after the other…

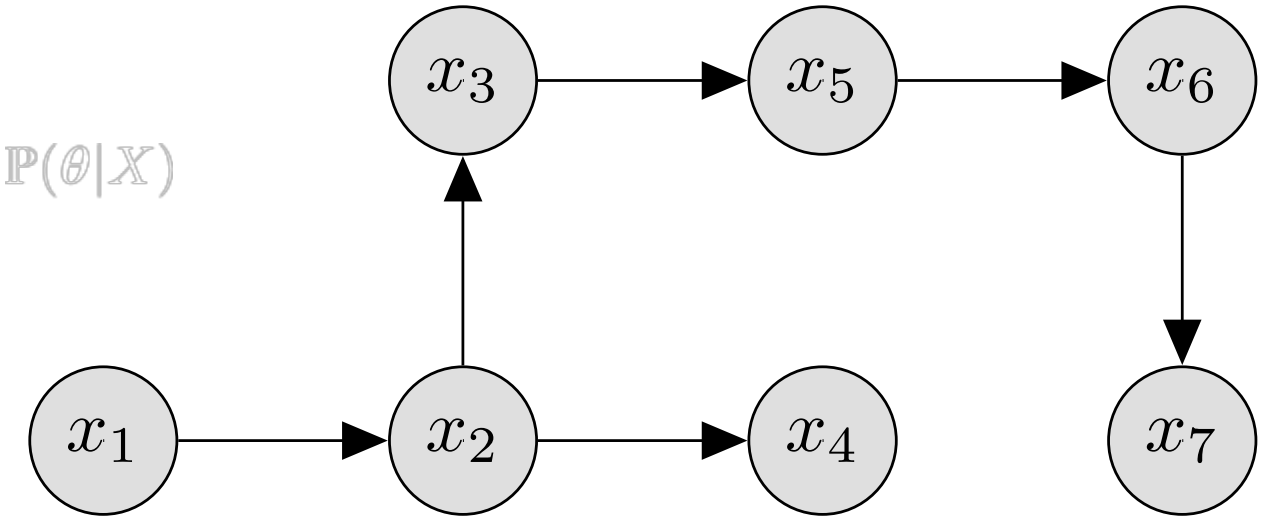


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

Of course, we could have chosen **any topological ordering!**

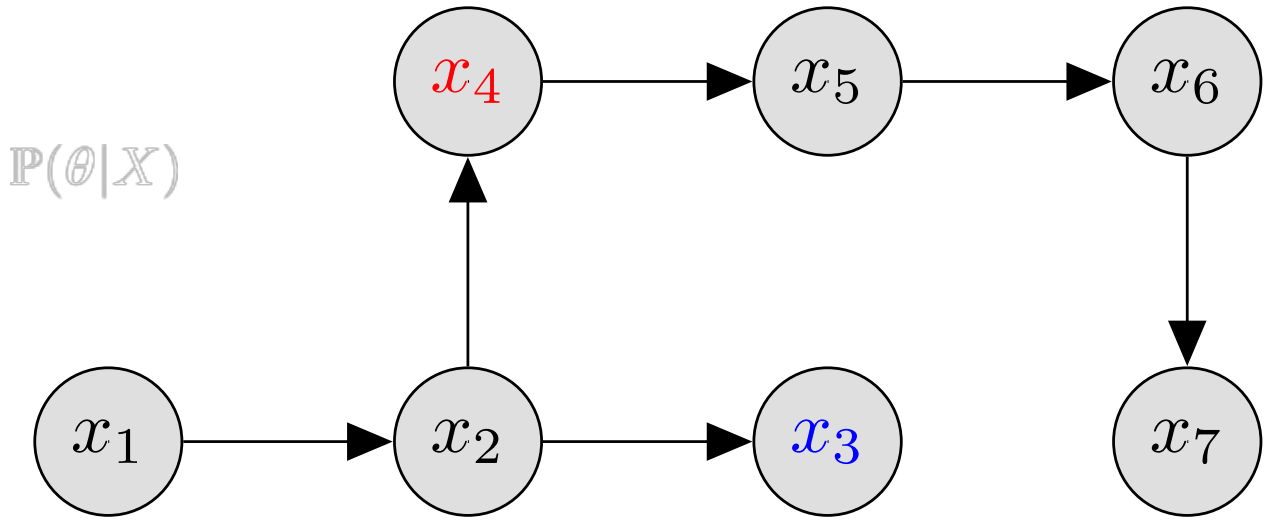


How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

Of course, we could have chosen **any topological ordering!**



How to sample from a directed graph?

Algorithm 1 Ancestral sampling with topological ordering

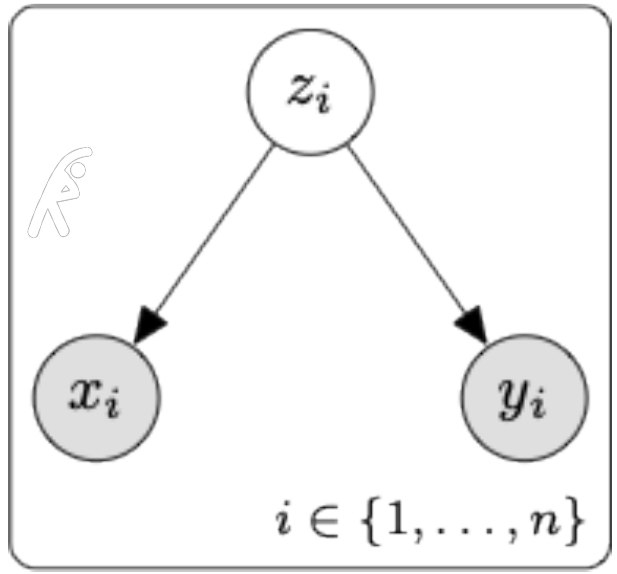
for $i = 1$ to d **do**
  Draw z_i from $p(X_i = .|X_{\mathrm{pa}_i} = z_{\mathrm{pa}_i})$
end for
  return $(z_1, \ldots, z_d)$

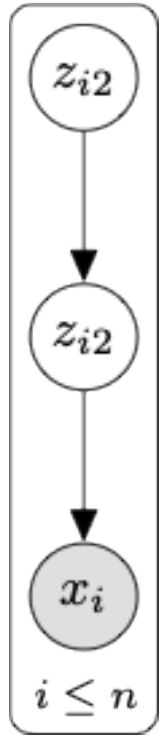

Proposition (ancestral sampling)**.** *Algorithm 1 provides samples from the joint distribution* $p(x_1, ..., x_d)$.

Proof. By induction on the topological ordering. □

Think of ways to implement this for some of the graphs we saw

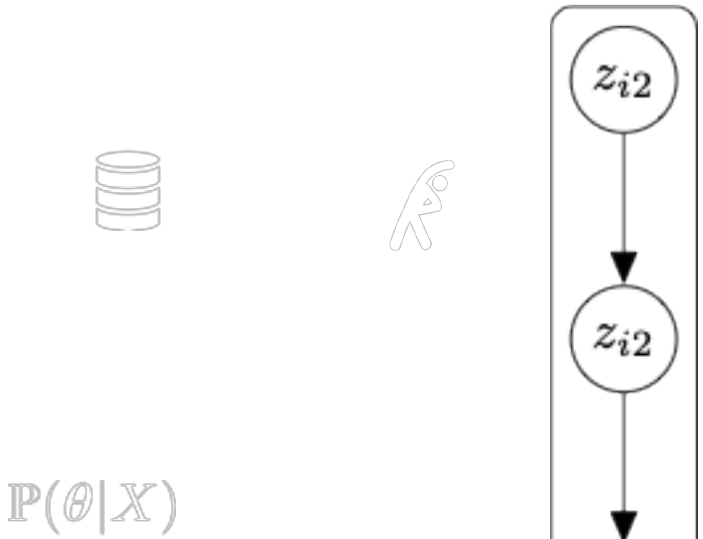
- Mixed-data clustering





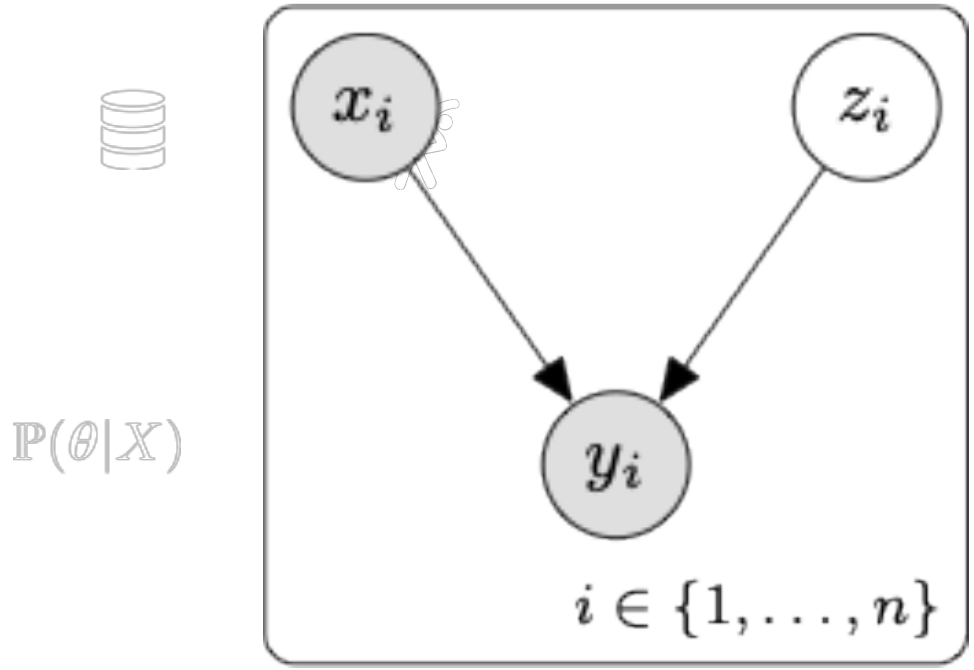
Think of ways to implement this for some of the graphs we saw

- Hierarchical VAE



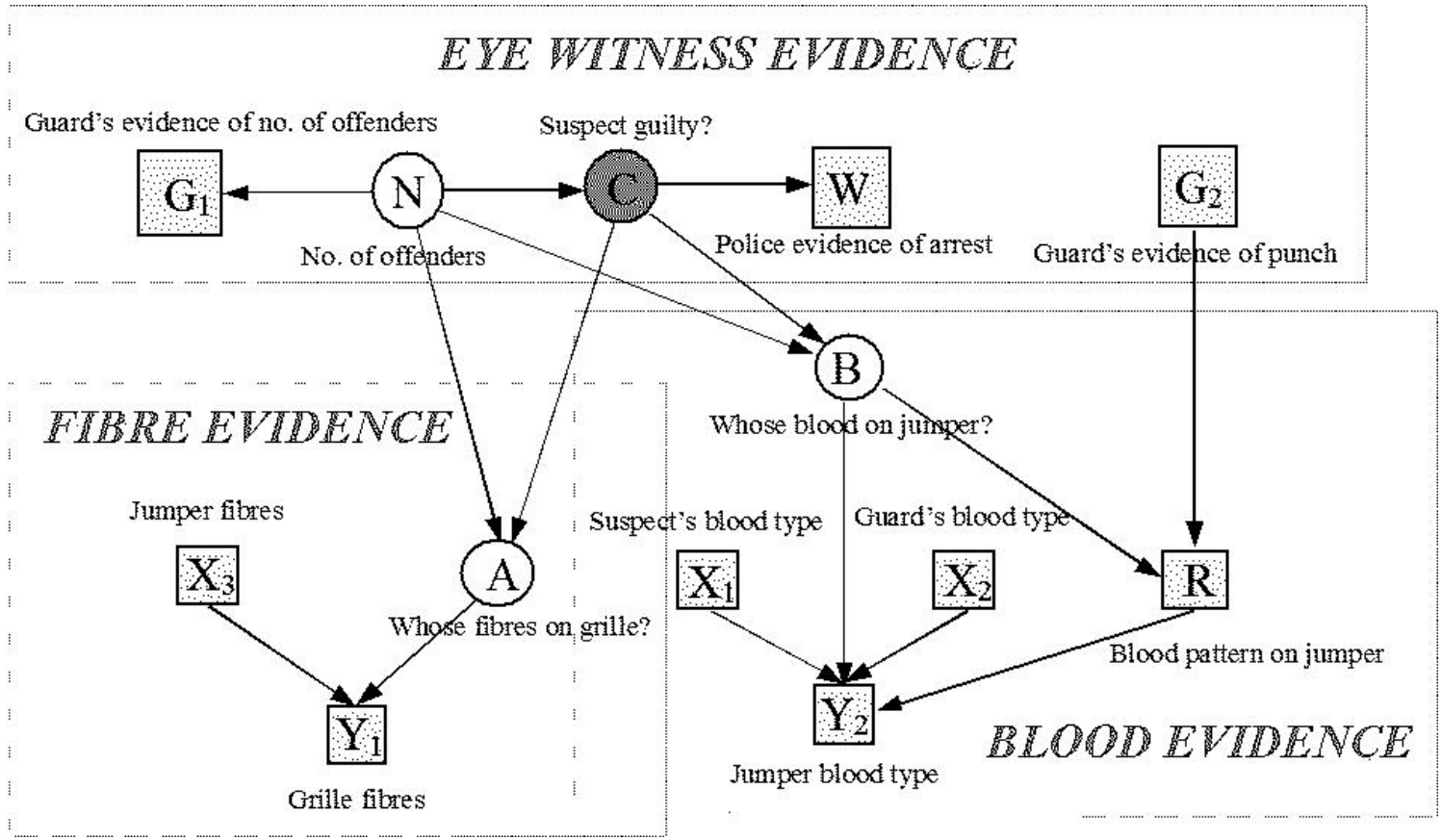
Think of ways to implement this for some of the graphs we saw

- Mixture of experts with constant gating



Think of ways to implement this for some of the graphs we saw

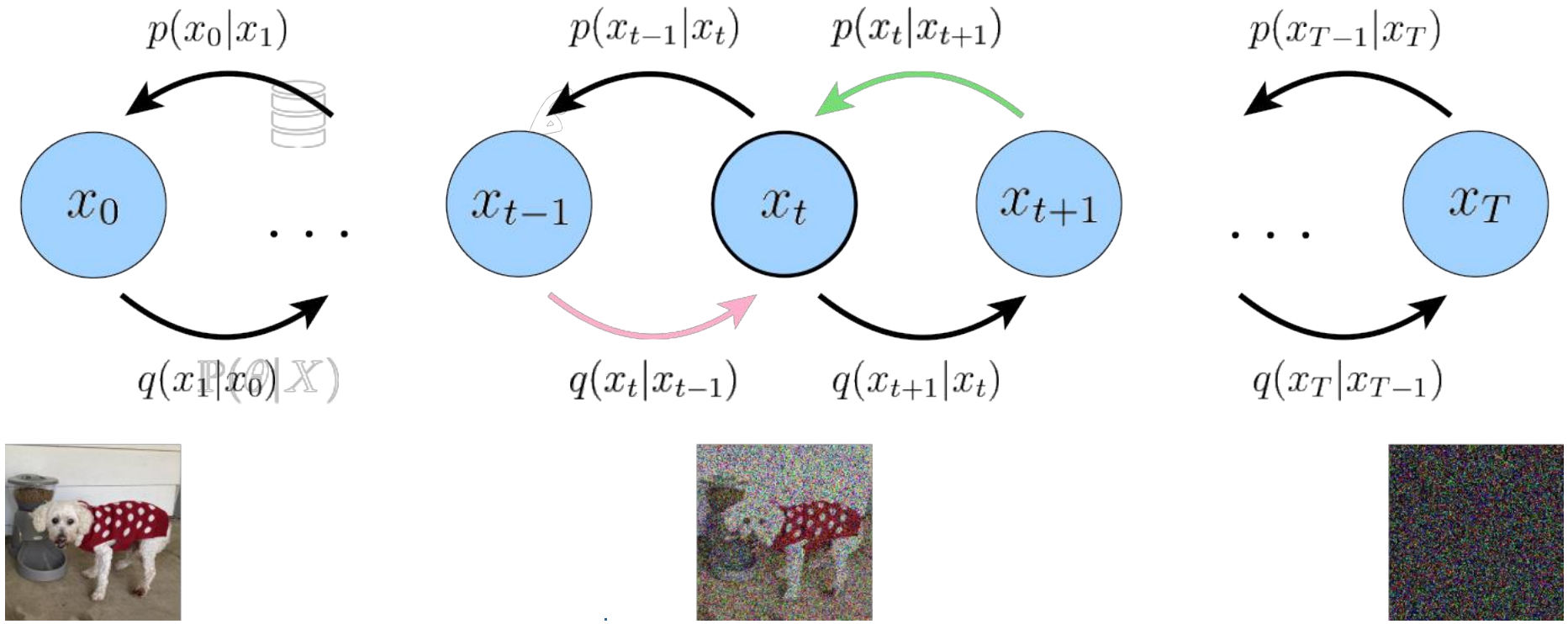
- The big graph from the forensic science example… Here the topological ordering is related to the time of the events…



Dawid and Evett, Using a graphical method to assist the evaluation of complicated patterns of evidence. Journal of Forensic Sciences, 1997

Think of ways to implement this for some of the graphs we saw

- Diffusion model (again, two DAGs!)



Luo, Understanding Diffusion Models: A Unified Perspective, arXiv preprint, 2022