Graphical Models and causality

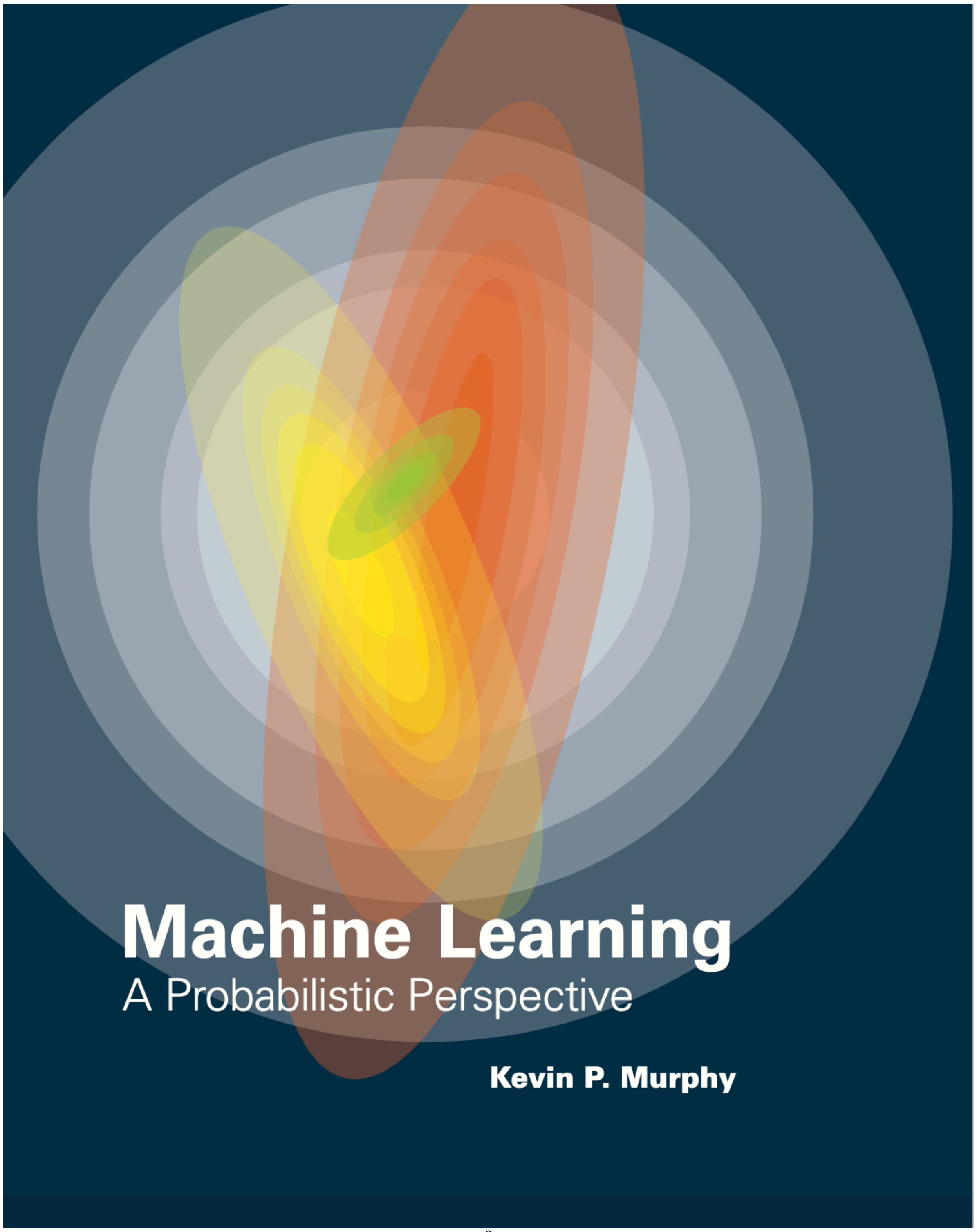
Christophe Ambroise

2025-09-03

Table of contents

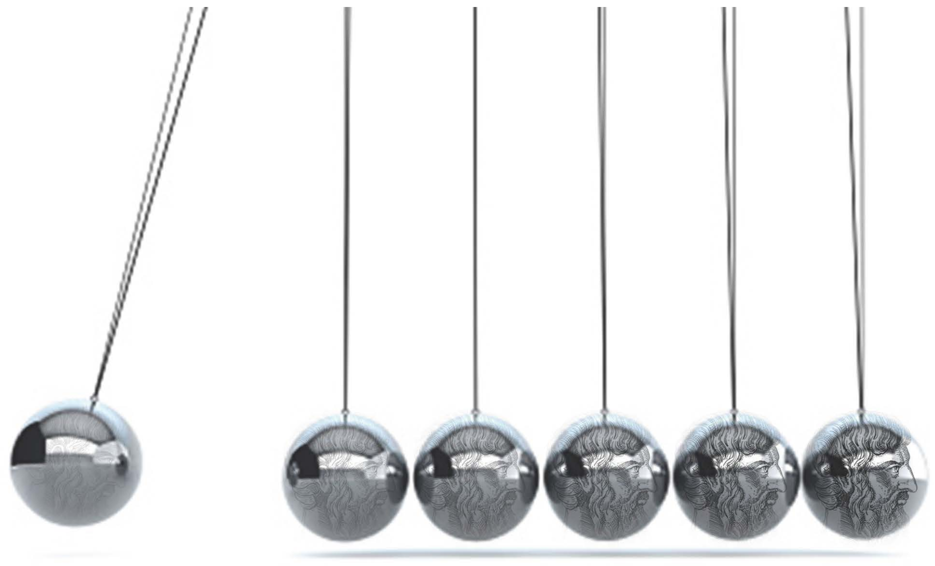
1 Introduction

1.1 Reference documents



Machine Learning: A Probabilistic Perspective

by Kevin P. Murphy
A comprehensive introduction to graphical models and probabilistic modeling across applied domains.



CAUSAL INFERENCE IN STATISTICS

A Primer

Judea Pearl
Madelyn Glymour
Nicholas P. Jewell

WILEY

Causal Inference in Statistics: A Primer
by Judea Pearl, Madelyn Glymour & Nicholas P. Jewell
Accessible approach to causal reasoning, interventions, and graphical causal models.

1.2 Reference documents

The lecture closely follows and largely borrows material from "Machine Learning: A Probabilistic Perspective" (MLAPP) from Kevin P. Murphy, chapters:

- Chapter 10: Directed graphical models (Bayes nets)
- Chapter 19: Undirected graphical models (Markov random fields)
- Chapter 20: Exact inference for graphical models
- Chapter 26: Graphical model structure learning

and from "Causal inference in statistics" from Judea Pearl et. al

1.3 Practical matters

1.3.1 Evaluation

- a written exam

1.4 What is a graphical model ?

A graphical model is a probability distribution in a factorized form

There a two main type of representation of the factorization:

- directed graphical model
- undirected graphical model

1.4.1 Why the term graph ?

Conditionnal independences between variables are well modeled via Graphs

1.5 What is it usefull for ?

- reduce the number of parameters
  - may be used for supervised or unsupervised approaches
- allow exploratory data analysis by providing a simple graphical representation
  - "approach causality"

1.6 What problems does it raise ?

- learning the parameters of a given factorized form
- learning the structure of the graphical model (factorized form)

1.7 What is Causality?

> Cause = That which **produces** or **modifies** an effect.

Causality refers to a relationship in which:

- One phenomenon (the **cause**)
- Brings about or influences another (the **effect**)

1.8 Historically:

1.8.1 Aristotle (384-322 BC): cause explains *why* things happen

"We do not have knowledge of a thing until we have grasped its why—that is to say, its cause." Wikipedia

1.8.2 Hume (1711-1776): we only observe sequences, not true causation

> When you drop a vase and it breaks you don't observe the 'causation.' You observe a vase dropping and you observe it shattering.

Causality stems from **habit or mental expectation**, built over repeated experience.

1.9 Causality in Science

- Causality relies on **temporal order** and **natural laws**
- Often formalized in equations:

$$\frac{dY}{dt} = f(X, Y, t)$$

This shows that variable X drives changes in Y over time.

1.10 Causality in Statistics

Three main views:

- **Dependence**: $P(Y \mid X = x) \neq P(Y)$
- **Prediction**: Granger causality — past X predicts future Y
- **Intervention** (Pearl):

$$X \text{ causes } Y \Leftrightarrow P(Y \mid do(X = x)) \neq P(Y)$$

$\rightarrow$ **Causal effect statistical association**

1.11 Simpson's Paradox: When Numbers Lie

Simpson's paradox occurs when a statistical association observed in the **aggregate population** reverses when the data are split into **subgroups**.

$\rightarrow$ It shows why **causality statistics**.

1.12 Kidney Stone Treatment Example

Suppose a new treatment for kidney stones seems worse overall:

- **New drug**: 78% recovery
- **Standard**: 83% recovery

But when we stratify by **stone size**, we find the paradox…

1.13 Stratified Recovery Rates

Stone Size	Treatment	Recovered	Total	Recovery Rate
Small	New	81	87	93%
Small	Standard	234	270	87%
Large	New	192	263	73%
Large	Standard	55	80	69%
Total	New	273	350	**78%**
Total	Standard	289	350	**83%**

Contingency Table (Kidney Stone Example)

1.14 What's Going On?

- **New drug** is better for both small and large stones
- But the drug was more often given to patients with **large stones**
  - These are harder to treat
- Aggregated data hides this → **confounding bias**

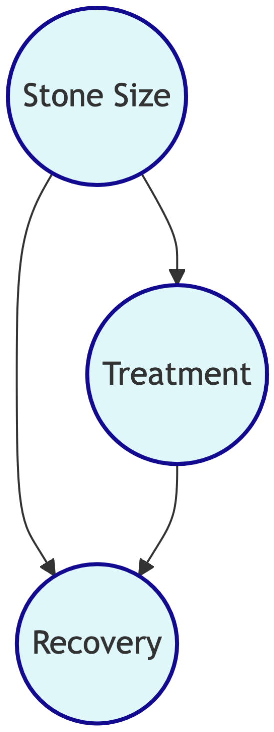


Figure 1: Causal DAG for Simpson's paradox: treatment (T) influences recovery (R), but stone size (S) is a confounder.

1.15 Causal Graph Explains It

2 Directed Graphical Models (Chapter 10 MLAPP)

2.1 Joint distribution

2.1.1 Observation

Suppose we observe multiple correlated variables, such as words in a document, pixels in an image, or genes in a microarray.

2.1.2 Joint distribution

How can we compactly represent the joint distribution $p(x|\theta)$?

2.2 Chain Rule

By the chain rule of probability, we can always represent a joint distribution as follows, using any ordering of the variables:

$$p(x_{1:V}) = p(x_1)p(x_2|x_1)p(x_3|x_2, x_1)p(x_4|x_1, x_2, x_3)...p(x_V|x_{1:V-1})$$

2.2.1 The problem of the number of parameters

$O(K) + O(K^2) + O(K^3) + ...$ There are $O(K^V)$ parameters in the system

2.3 Conditional independence

The key to efficiently representing large joint distributions is to make some assumptions about conditional independence (CI).

$$X \perp Y|Z \Leftrightarrow p(X, Y|Z) = p(X|Z)p(Y|Z)$$

X is conditionaly independent of Y knowing Z if once you know Z knowing Y does not help you to guess X

2.4 Conditional independence: an example

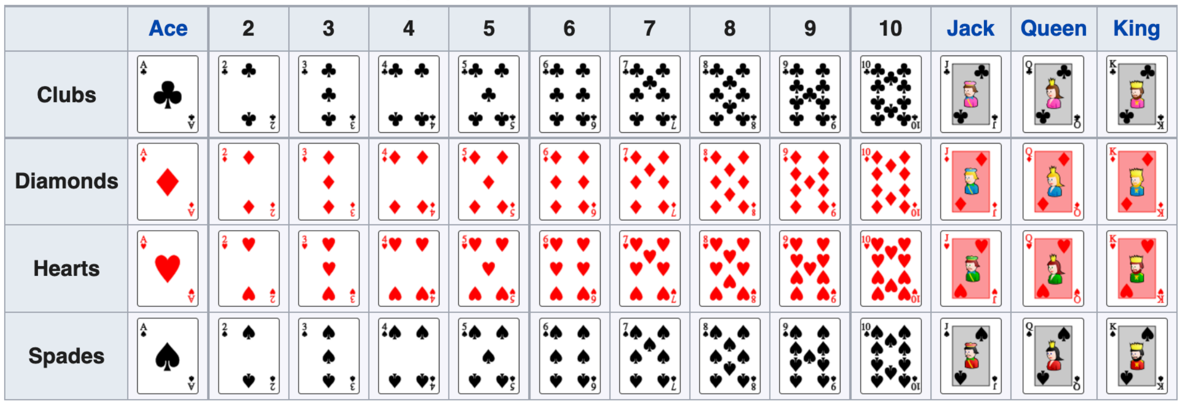
Setting: picking a card at random in a traditional set of cards

1. if full set of color and values then $color \perp value$
2. if all diamond faces ($\blacklozenge$) are discarded from the set then $color \not\perp value$ but still $color \perp value | Facecard$

$$P(King|Facecard) = 1/3 = P(\clubsuit|Facecard)$$

$$P(King\clubsuit|Facecard) = 1/9 = P(King|Facecard)P(\clubsuit|Facecard)$$

Example set of 52 playing cards; 13 of each suit: clubs, diamonds, hearts, and spades


Ace
2
3
4
5
6
7
8
9
10
Jack
Queen
King
Clubs
Diamonds
Hearts
Spades

2.5 Simplification of chain rule

2.5.1 Simplficiation of chain rule factorization

Let assume that $x_{t+1} \perp x_{1:t-1} | x_t$, first order Markov assumption.

$$p(x_{1:V}) = p(x_1) \prod_{t=2}^{V} p(x_t|x_{t-1})$$

$K - 1 + K^2$ parameters

2.6 Graphical models

A graphical model (GM) is a way to represent a joint distribution by making Conditional Independence (CI) assumptions.

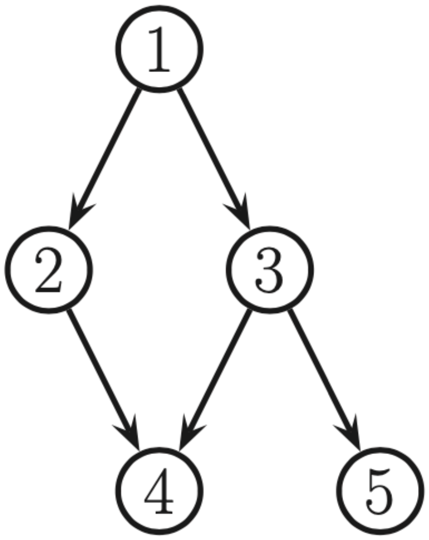
- the nodes in the graph represent random variables,
- and the (lack of) edges represent CI assumptions.

A better name for these models would in fact be ‘’independence diagrams’ ’

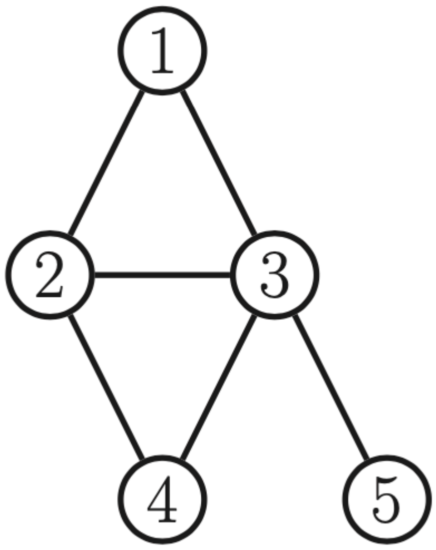
There are several kinds of graphical model, depending on whether

- the graph is directed,
- undirected,
- or some combination of directed and undirected.

2.7 Example of directed and undirected graphical model



(a)



(b)

2.8 Graph terminology

A graph $G = (V, E)$ consists of

- a set of nodes or vertices, $V = \{1, ..., V\}$, and

- a set of edges, $E = \{(s,t) : s, t \in V\}$.

2.8.1 Adjacency matrix

We can represent the graph by its adjacency matrix, in which we write $G(s,t) = 1$ to denote $(s,t) \in E$, that is, if $s \to t$ is an edge in the graph. If $G(s,t) = 1$ iff $G(t,s) = 1$, we say the graph is undirected, otherwise it is directed.

We usually assume $G(s,s) = 0$, which means there are no self loops.

2.9 Graph terminology

- **Parent**: For a directed graph, the parents of a node is the set of all nodes that feed into it: $pa(s) \triangleq \{t : G(t,s) = 1\}$.
- **Child**: For a directed graph, the children of a node is the set of all nodes that feed out of it: $ch(s) \triangleq \{t : G(s,t) = 1\}$.
- **Family**: For a directed graph, the family of a node is the node and its parents, $fam(s) = s \cup pa(s)$.
- **Root**: For a directed graph, a root is a node with no parents.
- **Leaf**: For a directed graph, a leaf is a node with no children.
- **Ancestors**: For a directed graph, the ancestors are the parents, grand-parents, etc of a node. That is, the ancestors of t is the set of nodes that connect to t via a trail: $anc(t) \triangleq \{s : s \rightsquigarrow t\}$.
- **Descendants**: For a directed graph, the descendants are the children, grand-children, etc of a node. That is, the descendants of s is the set of nodes that can be reached via trails from s: $desc(s) \triangleq \{t : s \rightsquigarrow t\}$.

2.10 Graph terminology

- **Clique**: For an undirected graph, a clique is a set of nodes that are all neighbors of each other.
- **A maximal clique** is a clique which cannot be made any larger without losing the clique property.
- **Neighbors** For any graph, we define the neighbors of a node as the set of all immediately connected nodes, $nbr(s) \triangleq \{t : G(s,t) = 1vG(t,s) = 1\}$. For an undirected graph, we write $s \sim t$ to indicate that s and t are neighbors.
- **Degree**: The degree of a node is the number of neighbors. For directed graphs, we speak of the in-degree and out-degree, which count the number of parents and children.
- **Cycle or loop**: For any graph, we define a cycle or loop to be a series of nodes such that we can get back to where we started by following edges
- **DAG** A directed acyclic graph or DAG is a directed graph with no directed cycles.

2.11 Directed graphical models

- A directed graphical model or DGM is a GM whose graph is a DAG.
- These are more commonly known as **Bayesian networks**
- These models are also called **belief networks**
- Finally, these models are sometimes called **causal networks**, because the directed arrows are sometimes interpreted as representing causal relations.

2.12 Topological ordering of DAGs

- nodes can be ordered such that parents come before children
- it can be constructed from any DAG

2.12.1 The ordered Markov property

a node only depends on its immediate parents

$$x_s \perp x_{pred(s) \setminus pa(s)} | x_{pa(s)}$$

where pa(s) are the parents of node s, and pred(s) are the predecessors of node s in the ordering.

2.13 General form of factorization

$$p(x_{1:V}) = \prod_{t=1}^{V} p(x_t | x_{pa(t)})$$

if the Conditional Independence assumptions encoded in DAG G are correct

3 Examples

3.1 Naive Bayes classifiers

$$p(y, x) = p(y) \prod_j p(x_j | y)$$

The naive Bayes assumption is rather naive, since it assumes the features are conditionally independent.

3.2 Markov and hidden Markov models

3.2.1 Markov chain

$$p(x_{1:T}) = p(x_1)p(x_2|x_1)p(x_3|x_2)... = p(x_1)\prod_{t=2}^{T} p(x_t|x_{t-1})$$

3.2.2 Hidden Markov Model

The hidden variables often represent quantities of interest, such as the identity of the word that someone is currently speaking. The observed variables are what we measure, such as the acoustic waveform.

3.3 Directed Gaussian graphical models

Consider a DGM where all the variables are real-valued, and all the Conditional Proba. Distributions have the following form:

$$p(x_t|x_{pa(t)}) = \mathcal{N}(x_t|\mu_t + w_t^T x_{pa(t)}, \sigma_t^2)$$

3.3.1 Directed GGM (Gaussian Bayes net)

$$p(x) = \mathcal{N}(x|\mu, \Sigma)$$

3.4 Directed GGM (Gaussian Bayes net)

For convenience let rewrite the CPDs

$$x_t = \mu_t + \sum_{s \in pa(t)} w_{ts}(x_s - \mu_s) + \sigma_t z_t$$

where $z_t \sim \mathcal{N}(0, 1)$, σ_t is the conditional standard deviation of x_t given its parents, wts is the strength of the $s \to t$ edge, and μ_t is the local mean.

3.4.1 Mean

The global mean is just the concatenation of the local means

$$\mu = (\mu_1, ..., \mu_D)^t.$$

3.5 Directed GGM (Gaussian Bayes net)

3.5.1 Covariance matrix

$$(x-\mu) = W(x-\mu) + Sz$$

where $S \triangleq diag(S)$ Let consider $e \triangleq Sz = (I-W)(x-\mu)$

We have

$$\Sigma = cov(x-\mu) = cov((I-W)^{-1}e) = cov(USz) = US^2U^t$$

where $U = (I-W)^{-1}$

3.6 Examples

Two extreme cases

- Isolated vertices : Naive Bayes where $\Sigma = S$, p vertices, no edges
- Fully connected Graph: p vertices, $p(p-1)/2$ directed edges

Click to goto exercice on Directed GGM

4 Learning

4.1 Learning from complete data (with known graph structure)

If all the variables are fully observed in each case, so there is no missing data and there are no hidden variables, we say the data is complete.

$$p(\mathcal{D}|\theta) = \prod_{i=1}^{N} p(x_i|\theta) = \prod_{i=1}^{N}\prod_{t\in V} p(x_{it}|x_{i,pa(t)},\theta_t)$$

The likelihhod decomposes according the **graph structure**

Click to goto exercice on Sprinkler

4.1.1 Discrete distribution

$$N_{tck} \triangleq \sum_{i=1}^{N} \mathbb{I}(x_{i,t}=k, x_{i,pa(t)}=c)$$

and thus $\hat{p}(x_t=k, x_{pa(t)}=c) = \frac{N_{tck}}{\sum_{k'} N_{tck'}}$ Of course, the MLE suffers from the zero-count

5 Conditional independence properties of DGMs

5.1 Diverging edges (fork)

With the DAG

$$A \leftarrow C \rightarrow B$$

with have

$$A \not\perp B$$

but

$$A \perp B | C$$

5.1.1 Exercice

Show it

5.2 Chain (Head - tail)

With the DAG

$$A \rightarrow C \rightarrow B$$

with have

$$A \not\perp B$$

but

$$A \perp B | C$$

5.2.1 Exercice

Show it

5.3 Converging edges (V) and collider

With the DAG

$$A \to C \leftarrow B$$

with have

$$A \perp B$$

but

$$A \not\perp B | C$$

5.3.1 Exercice

Show it

5.4 Independence map

a directed graph G is an I-map (independence map) for p, or that p is Markov wrt G,

- iff $I(G) \subseteq I(p)$, where $I(p)$ is the set of all CI statements that hold for distribution p.

This allows us to use the graph as a safe proxy for p

5.4.1 Minimal I-map

- The fully connected graph is an I-map of all distributions,
- G is a of Minimal I-map p
    1. if G is an I-map of p,
    2. if there isno $G' \subseteq G$ which is an I-map of p.

5.5 d-separation

- The “d” in d-separation and d-connection stands for dependence.
- d-separation is related the ideas of active path and active vertex on a path
- **a path is active if it carries information, or dependence.**
- Thus, when the conditioning set is empty, only paths that correspond to “causal connection” are active (creating dependance).

5.6 d-separation: example of Pearl (1988)

two independent causes of your car refusing to start: having no gas and having a dead battery.

dead battery –> car won’t start <– no gas

- Telling you that the battery is charged tells you nothing about whether there is gas,
- Telling you that the battery is charged after I have told you that the car won’t start tells me that the gas tank must be empty.

So independent causes are made dependent by conditioning on a common effect, which in the directed graph representing the causal structure is the same as conditioning on a collider.

5.7 d-separation

When a vertex is in the conditioning set, its status with respect to being active or inactive **flip-flops**. **If we condition by C**

Are variables A and B are d-separated by C (in boldface).

1. A –> **C** –> B **Inactive**
2. A <– **C** <– B **Inactive**
3. A <– **C** –> B **Iactive**
4. A –> **C** <– B, C is a collider and thus inactive when the conditioning set is empty, so condiitionning by C it becomes **Active** (produce dependence)

5.8 Formulation d-separation definition

an undirected path P is d-separated by a set of nodes E iff at least one of the following conditions hold:

- P contains a chain, $s \rightarrow m \rightarrow t$ or $s \leftarrow m \leftarrow t$ where $m \in E$
- P contains a fork, $s \leftarrow m \rightarrow t$ where $m \in E$
- P contains a collider, $s \rightarrow m \leftarrow t$ where $m \notin E$ and nor is any descendant of m.

5.9 Alternative formulation of d-connection:

If G is a directed graph in which X, Y and E are disjoint sets of vertices, then X and Y are d-connected by E in G if and only if there exists an undirected path P between some vertex in X and some vertex in Y such that

- for every collider C on P, either C or a descendent of C is in E (active path),
- and no non-collider on P is in E (no inactive path).

X and Y are d-separated by E in G if and only if they are not d-connected by E in G (all path are inactives…).

Independance requires **all possible paths** to be inactive whereas dependence requires only on leak (one active path)

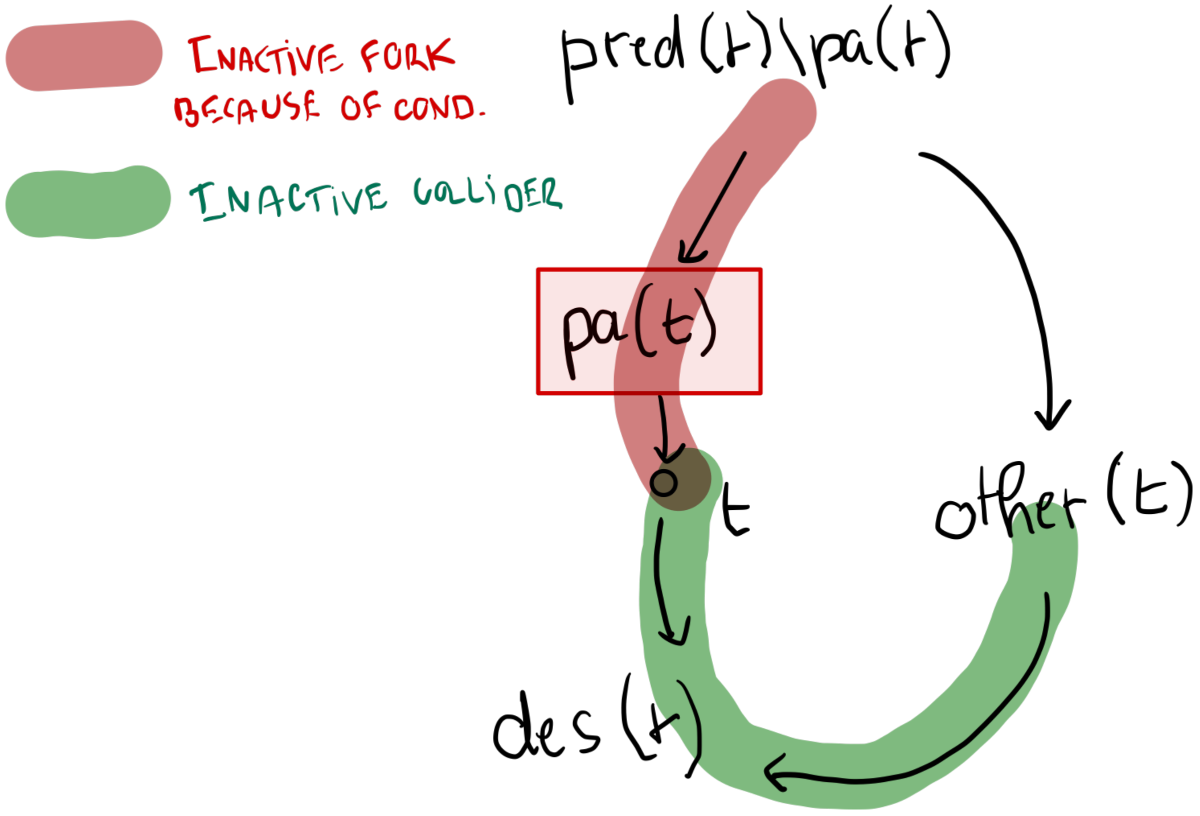
see https://www.youtube.com/watch?v=yDs_q6jKHb0 for examples

5.10 d-separation versus conditional independence

a set of nodes A is d-separated from a different set of nodes B given a third observed set E iff each undirected path from every node $a \in A$ to every node $b \in B$ is d-separated by E:

$x_A \perp_G x_B | x_E \Leftrightarrow$ A is d-separated from B given E

5.11 Consequences of d-separation



5.11.1 Directed local Markov property

From the d-separation criterion, one can conclude that $t \perp nd(t) \backslash pa(t) | pa(t)$ where the non-descendants of a node $nd(t)$ are all the nodes except for its descendants

5.12 Consequences of d-separation

5.12.1 Ordered Markov property

A special case of directed local Markov property is when we only look at predecessors of a node according to some topological ordering. We have $t \perp pred(t) \backslash pa(t) | pa(t)$

5.13 Markov blanket

The set of nodes that renders a node t conditionally independent of all the other nodes in the graph is called t's Markov blanket

$$mb(t) \triangleq pa(t) \cup ch(t) \cup copa(t)$$

The Markov blanket of a node in a DGM is equal to the parents, the children, and the co-parents.

5.14 Markov blanket

To understand the Markov blanket, one could start with the local Markov property which block the dependence to non-descendant by conditioning on the parents.

To further block the path the descendants of t one has to

- Condition on the children of t.
- But **conditioning on the children** open the path to the coparents.
- Thus one needs conditioning **on the coparents** to block all paths.

6 Graphical Model Learning Structure (chapter 26 MLAPP)

6.1 Introduction

Two main applications of structure learning:

1. knowledge discovery (requires a graph topology)
2. density estimation (requires a fully specified model).

6.1.1 main obstacle

the number of possible graphs is exponential in the number of nodes: a simple upper bound is $O(2^{V(V-1)/2})$.

6.2 Relevance network

A relevance network is a way of visualizing the pairwise mutual information between multiple random variables:

- we simply choose a threshold α
- draw an edge from node i to node j if $\mathbb{I}(X_i; X_j) > \alpha$

6.2.1 Major problem

- the graphs are usually very dense,
- most variables are dependent on most other variables, even after thresholding the MIs.

6.3 Gaussian case

In the Gaussian case, $\mathbb{I}(X_i; X_j) = -1/2 \log{(1 - \rho_{ij}^2)}$ where ρ_{ij} is the correlation coefficient so we are essentially visualizing Σ;

this is known as the covariance graph.

6.3.1 Exercice : Gaussian mutual information

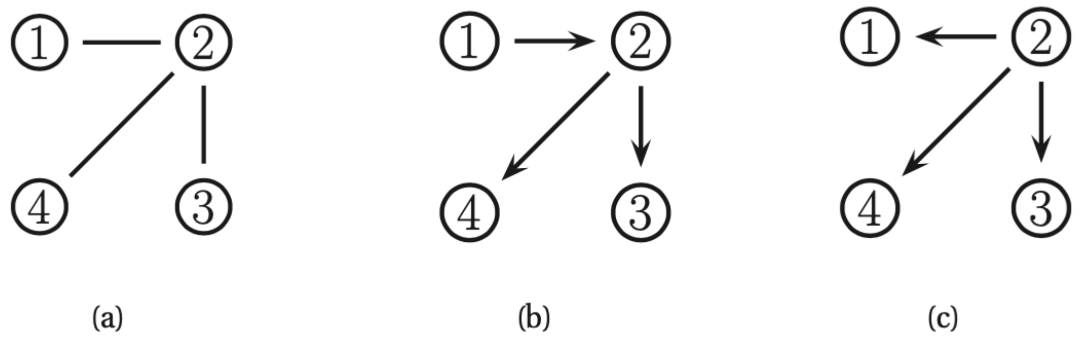
Show the previous statement

6.4 Dependency networks

- A simple and efficient way to learn a graphical model structure is to independently fit D sparse full-conditional distributions $p(x_t|x_{-t})$
- any kind of sparse regression or classification method to fit each CPD
  - (Heckerman et al. 2000) uses classification/regression trees,
  - (Meinshausen and Buhlmann 2006) use ℓ_1-regularized linear regression,
  - (Wainwright et al. 2006) use ℓ_1-regularized logistic regression (see depnetFit for some code),
  - (Dobra 2009) uses Bayesian variable selection

6.5 Learning tree structures

Since the problem of structure learning for general graphs is NP-hard (Chickering 1996), we start by considering the special case of trees. Trees are special because we can learn their structure efficiently



An undirected tree and two equivalent directed trees.

6.6 Joint Distribution associated to a directed tree

A directed tree, with a single root node r, defines a joint distribution as follows

$$p(x|T) = \prod_{t \in V} p(x_t | x_{pa(t)})$$

The distribution is a product over the edges and the choice of root does not matter

6.6.1 Symetrization

To make the model more symmetric, it is preferable to use an undirected tree:

$$p(x|T) = \prod_{t \in V} p(x_t) \prod_{(s,t) \in E} \frac{p(x_s, x_t)}{p(x_s) p(x_t)}$$

6.7 Chow-Liu algorithm for finding the ML tree structure (1968)

Goal: Chow Liu algorithm constructs tree distribution approximation that has the minimum Kullback–Leibler divergence to the actual distribution (that maximizes the data likelihood)

6.7.1 Principle

1. Compute weight $I(s,t)$ of each (possible) edge (s,t)
2. Find a maximum weight spanning tree (MST)
3. Give directions to edges in MST by chosing a root node

6.8 Chow-Liu algorithm for finding the ML tree structure (1968)

6.8.1 log-likelihood

$$\log P(\theta|\mathcal{D},T) = \sum_{tk} N_{tk} \log p(x_t = k) + \sum_{st}\sum_{jk} N_{stjk} \log \frac{p(x_s = j, x_t = k)}{p(x_s = j)p(x_t = k)}$$

thus $\hat{p}(x_t = k) = \frac{N_{tk}}{N}$ and $\hat{p}(x_s = j, x_t = k) = \frac{N_{stjk}}{N}$.

6.8.2 Mutual information of a pair of variables

$$I(s,t) = \sum_{jk} \hat{p}(x_s = j, x_t = k) \log \frac{\hat{p}(x_s = j, x_t = k)}{\hat{p}(x_s = j)\hat{p}(x_t = k)}$$

6.8.3 The Kullback–Leibler divergence

$$\frac{\log P(\hat{\theta}_{ML}|\mathcal{D},T)}{N} = \sum_{tk} \hat{p}(x_t = k) \log \hat{p}(x_t = k) + \sum_{st} I(s,t)$$

6.9 Chow-Liu algorithm

There are several algorithms for finding a max spanning tree (MST). The two best known are - Prim's algorithm and - Kruskal's algorithm.

Both can be implemented to run in $O(ElogV)$ time, where $E = V^2$ is the number of edges and V is the number of nodes.

6.10 Kruskal Algorithm for Maximum Spanning Tree

1. Sort the edges of G into decreasing order by weight. Let T be the set of edges comprising the maximum weight spanning tree. Set $T = \emptyset$.
2. Add the first edge to T.
3. Add the next edge to T if and only if it does not form a cycle in T. If there are no remaining edges exit and report G to be disconnected.
4. If T has $n-1$ edges (where n is the number of vertices in G) stop and output T. Otherwise go to step 3.

6.11 Exercice Gaussian Chow-Liu

1. Show that in the Gaussian case, $I(s,t) = -\frac{1}{2}\log(1-\rho_{st}^2)$,where ρ_{st} is the correlation coefficient (see Exercise 2.13, Murphy)
2. Given a realisation of n gaussian vector of size p find the ML tree structured covariance matrix using Chow-Liu algorithm.

6.12 TAN: Tree-Augmented Naive Bayes

- Chow-Liu for each class to get a Tree
- Directing each tree
- Estimating the covariance matrices from the tree

6.13 Mixtures of trees

- A single tree is rather limited in its expressive power.
- learning a mixture of trees (Meila and Jordan 2000), where each mixture component may have a different tree topology is an alternative

6.13.1 Integrating out over all possible trees.

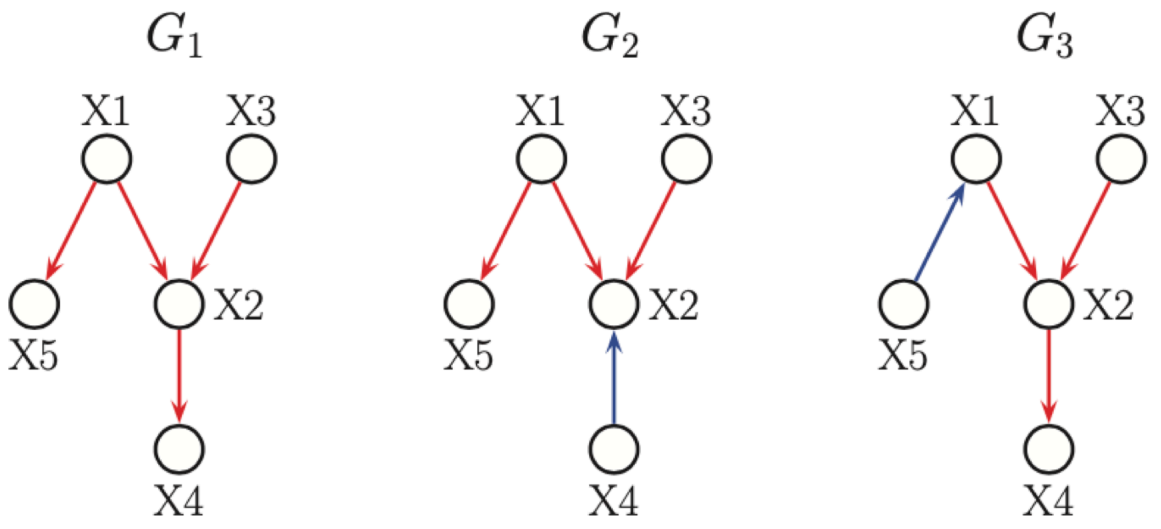
This can be done in V^3 time using the matrix tree theorem.

Matrix Tree Theorem: Chaiken and Kleitman (1978); Meila and Jaakkola (2006). For any symmetric weight matrix W with all entries in $\mathbb{R}^+$, the sum over all spanning trees of the product of the weights of their edges is equal to any minor of its Laplacian Q. That is, for any $1 \leq u, v \leq p$,

$W \triangleq \sum_{T\in\mathcal{T}} \prod_{jk\in T} w_{jk} = |Q_{uv}|$.

Consequently, the operation of summing over all spanning trees can be carried out in a computationally efficient way. Meila and Jaakkola (2006) built on this result to provide a close form expression for the derivative of the sum-product W with respect to each entry of the input weight matrix W. Without loss of generality, we choose Q_{11}.

6.14 Learning DAG structures



Three DAGs. G1 and G3 are Markov equivalent,G2 is not.

6.14.1 Graphs are Markov equivalent

if they encode the same set of CI assumptions

6.15 Learning DAG structures

6.15.1 An ill posed problem

when we learn the DAG structure from data, we will not be able to uniquely identify all of the edge directions

we can learn DAG structure "up to Markov equivalence".

Do not read too much into the meaning of particular edge orientations, since we can often change them without changing the model in any observable way.

6.16 Exact structural inference

Exact structural inference is based on the computation of exact posterior over graphs, $p(G|D)$.

It requires:

- the computation of the likelihood $p(D|G)$
- the computation of the prior $p(G)$

This solution allows to compared different graph in terms of posterior and eventually find the MAP if the search space is small

6.17 Exact structural inference (categorical case)

Consider $x_{it} \in \{1, \cdots, K_t\}$ be the value of node t in case i, where

- K_t is the number of states for node t.
- $\theta_{tck} \triangleq p(x_t = k|x_{pa(t)} = c)$, for $k = 1 : K_t$, and $c = 1 : C_t$, where C_t is the number of parent combinations (possible conditioning cases).

Let $d_t = dim(pa(t))$ be the degree or fan-in of node t, so that $C_t = K^{d_t}$.

6.18 Exact structural inference (categorical case)

6.18.1 Prior

$$p(\theta) = \prod_{t=1}^{V} p(\theta_t) = \prod_{t=1}^{V} \prod_{c=1}^{C_t} p(\theta_{tc})$$

where C_t is the number of parent combinations (possible conditioning cases)

6.18.2 Likelihood

$$p(D|G, \theta) = \prod_{t=1}^{V} \prod_{c=1}^{C_t} \prod_{k=1}^{K_t} \theta_{tck}^{N_{tck}}$$

where N_{tck} is the number of time node t is in state k and its parent in state c.

6.19 Exact structural inference (categorical case)

Chosing a Dirichlet prior $p(\theta_{tc}) = Dir(\theta_{tc}|\alpha_{tc})$ allows to compute the posterior

$$p(D|G) = \prod_{t=1}^{V} \prod_{c=1}^{C_t} \frac{B(N_{tc} + \alpha_{tc})}{B(\alpha_{tc})}$$

where $N_{tc} = \sum_k N_{tck}$, and $\alpha_{tc} = \sum_k \alpha_{tck}$.

6.19.1 Local scoring

For node t and its parents

$$score(N_{t,pa(t)}) \triangleq \prod_{c=1}^{C_t} \frac{B(N_{tc} + \alpha_{tc})}{B(\alpha_{tc})}$$

Marginal likelihood factorizes according to the graph structure.

6.20 Setting the prior

How should we set the hyper-parameters α_{tck} ?

- Jeffreys prior of the form $\alpha_{tck} = 1/2$ violates a property called likelihood equivalence
- This property says that if G1 and G2 are Markov equivalent , they should have the same marginal likelihood

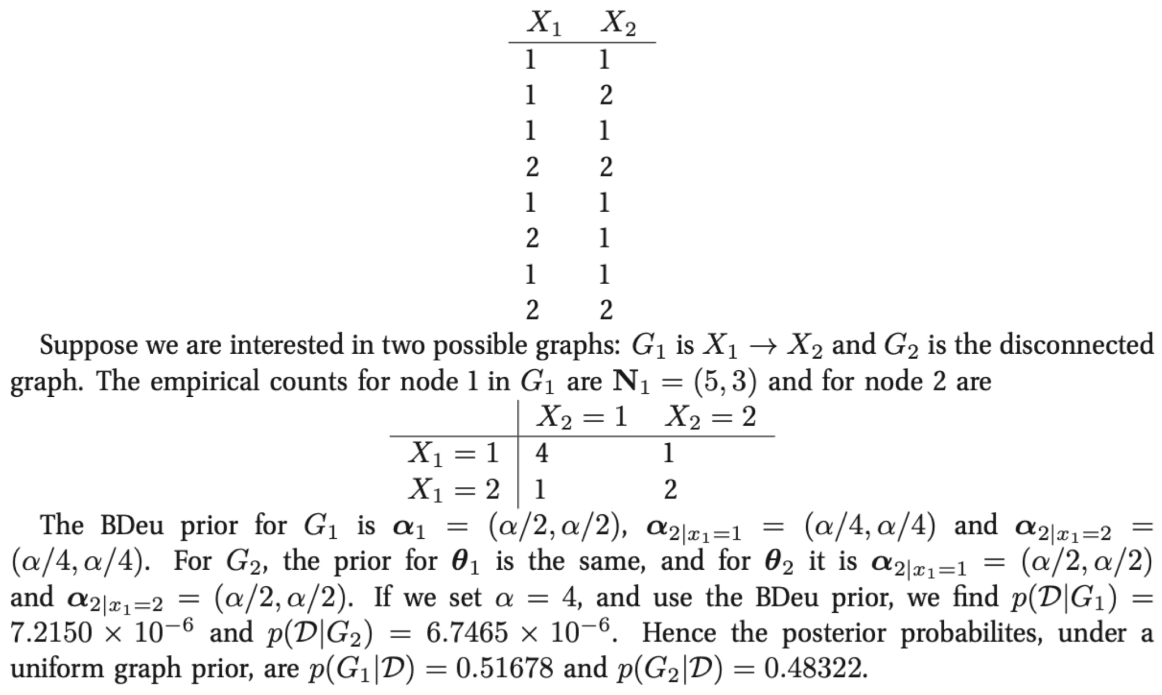
6.20.1 BDe prior

- Geiger and Heckerman (1997) proved that, for complete graphs, the only prior that satisfies likelihood equivalence and parameter independence is the Dirichlet prior, where the pseudo counts have the form

$$\alpha_{tck} = \alpha p_0(x_t = k, x_{pa(t)} = c)$$

where $\alpha > 0$ is called the equivalent sample size, and p_0 is some prior joint probability distribution. This is called the BDe prior (Bayesian Dirichlet likelihood equivalent).

6.21 Example of Exact structural inference (Neapolitan 2003, p.438)



X_1	X_2
1	1
1	2
1	1
2	2
1	1
2	1
1	1
2	2

Suppose we are interested in two possible graphs: G_1 is $X_1 \to X_2$ and G_2 is the disconnected graph. The empirical counts for node 1 in G_1 are $\mathbf{N}_1 = (5, 3)$ and for node 2 are

	$X_2 = 1$	$X_2 = 2$
$X_1 = 1$	4	1
$X_1 = 2$	1	2

The BDeu prior for G_1 is $\boldsymbol{\alpha}_1 = (\alpha/2, \alpha/2)$, $\boldsymbol{\alpha}_{2|x_1=1} = (\alpha/4, \alpha/4)$ and $\boldsymbol{\alpha}_{2|x_1=2} = (\alpha/4, \alpha/4)$. For G_2, the prior for $\boldsymbol{\theta}_1$ is the same, and for $\boldsymbol{\theta}_2$ it is $\boldsymbol{\alpha}_{2|x_1=1} = (\alpha/2, \alpha/2)$ and $\boldsymbol{\alpha}_{2|x_1=2} = (\alpha/2, \alpha/2)$. If we set $\alpha = 4$, and use the BDeu prior, we find $p(\mathcal{D}|G_1) = 7.2150 \times 10^{-6}$ and $p(\mathcal{D}|G_2) = 6.7465 \times 10^{-6}$. Hence the posterior probabilites, under a uniform graph prior, are $p(G_1|\mathcal{D}) = 0.51678$ and $p(G_2|\mathcal{D}) = 0.48322$.

6.22 Scaling up to larger graphs

The main challenge in computing the posterior over DAGs is that there are so many possible graphs.

Consequently, we must settle for finding a locally optimal MAP DAG.

6.22.1 Popular solution: Greedy hill climbing

- at each step, the algorithm proposes small changes to the current graph, such as adding, deleting or reversing a single edge;
- it then moves to the neighboring graph which most increases the posterior.
- The method stops when it reaches a local maximum.

6.23 Learning causal DAGs

6.23.1 Causal models

- predict the effects of interventions to, or manipulations of, a system.

- Causal claims are inherently stronger, yet more useful, than purely associative claims

6.23.2 Causal interpretation of DAGs

- $A \to B$ in a DAG to mean that "A directly causes B" so if we manipulate A, then B will change.
- Known as the causal Markov assumption.

6.24 Intervention

6.24.1 Perfect intervention

- represents the act of setting a variable to some known value
- A real world example of such a perfect intervention is a gene knockout experiment

6.24.2 do calculus notation

$do(X_i = x_i)$ to denote the event that we set X_i to x_i

- A causal model makes inferences of the form $p(x|do(X_i = x_i))$,
- Different from making inferences of the form $p(x|X_i = x_i)$.

6.25 Observing versus doing

Consider a 2 node DGM $S \to Y$

- $S = 1$ if you smoke
- $S = 0$ otherwise,
- $Y = 1$ if you have yellow-stained fingers
- $Y = 0$ otherwise.

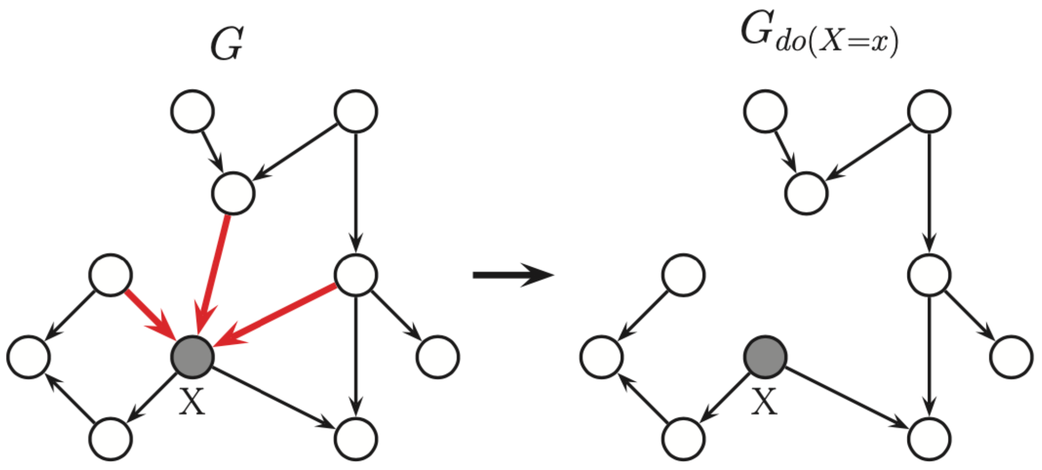
If I observe you have yellow fingers, I am licensed to infer that you are probably a smoker (since nicotine causes yellow stains):

$$p(S = 1|Y = 1) > p(S = 1)$$

If I intervene and paint your fingers yellow, I am no longer licensed to infer this, since I have disrupted the normal causal mechanism. Thus

$$p(S = 1|do(Y = 1)) = p(S = 1)$$

6.26 Graph surgery



One way to model perfect interventions is to use graph surgery: - represent the joint distribution by a DGM, - cut the arcs coming into any nodes that were set by intervention.

7 Markov Random Field or Undirected Graphical Models

8 Markov Random Fields

8.1 Problem of directed graph

- Several graphs can induce the same set of conditional independences .
- Is it possible to associate to each graph a family of distribution so that conditional independence coincides exactly with the notion of separation in the graph?

8.2 Conditional independence properties of UGMs

UGMs define CI relationships via simple graph separation:

8.2.1 Global Markov property for UGMs

for sets of nodes A, B, and C, we say $x_A \perp_G x_B | x_C$ iff C separates A from B in the graph G.

When we remove all the nodes in C, if there are no paths connecting any node in A to any node in B, then the CI property holds.

8.3 Other Markov properties

8.3.1 Local Markov property

A variable is conditionally independent of all other variables given its neighbors:

$$X_v \perp X_{V \setminus \mathrm{N}[v]} \mid X_{\mathrm{N}(v)}$$

8.3.2 Pairwise Markov property

Any two non-adjacent variables are conditionally independent given all other variables:

$$X_u \perp X_v \mid X_{V \setminus \{u,v\}}$$

8.3.3 Links

It is obvious that global Markov implies local Markov which implies pairwise Markov.

8.4 Undirected Graph

8.4.1 A Markov Random Field

Given an undirected graph a Markov Random Field is associated with probability distributions obeying the global Markov property:

$$X_A \perp\!\!\!\perp X_B | X_C$$

8.4.2 The distribution $p(x)$ of a Markov Random Field

is given by the Hammersley-Clifford Theorem (1971)

$$p(x) = \frac{1}{Z} \prod_{c \in \mathrm{cl}(G)} \Psi_c(x_c|\theta_c)$$

where $\mathrm{cl}(G)$ is the set of all cliques of G, and

$$Z \triangleq \sum_x \prod_{c \in \mathrm{cl}(G)} \Psi_c(x_c|\theta_c)$$

is the "partition function".

8.5 Hammersley-Clifford Theorem

A distribution p (with $p(x) > 0$) for all x satisfies the Global Markov property for graph G iff it is a Gibbs distribution associated with G

$$p(x) = \frac{1}{Z} \prod_{c \in \mathrm{cl}(G)} \Psi_c(x_c|\theta_c)$$

It is easy to check the global Markov property if the distribution is Gibbs but more difficult to do the reverse.

8.6 Hammersley-Clifford Theorem

Goal: show that if we have the Global Markov Property then we have a Gibbs Distribution

The idea of the demonstration consists in considering a generic form subject to the global global Markov property:

Consider $Q(x) = \ln \frac{p(x)}{p(0)}$ and its unique decomposition on the interaction space (of the n variables)

$$Q(x) = \sum_i x_i G_i(x_i) + \sum_{i<j} x_i x_j G_{ij}(x_i, x_j) + \cdots + x_1 x_2 ... x_n G_{12...n}(x_1, x_2, ..., x_n)$$

To show that the expression $p(x) \propto \exp(Q(x))$ is a Gibbs distribution, we have only to prove that $G_A(x_A) = 0$ when A is not a complete subset of the graph.

8.7 Hammersley-Clifford Theorem

For example $x_i G_i(x_i) = Q(0, ..., 0, x_i, 0, ..., 0) - Q(0)$

Consider for any vectors x and $x^i = (x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n)$

$$\exp\left(Q(x) - Q(x^i)\right) = \frac{p(x)}{p(x^i)} = \frac{p(x_i, x_{V-i})}{p(0, x_{V-i})} = \frac{p(x_i \mid x_{\mathrm{N}[i]})}{p(0 \mid x_{\mathrm{N}[i]})}$$

Notice that

$$Q(x) - Q(x^1) = x_1(G_1(x_1) + \sum_{j \neq 1} x_j G_{1j}(x_1, x_j)) + \sum_{j \neq 1, j < k} G_{1jk}(x_1, x_j, x_k) + ... + x_2 ... x_n G_{12...n}(x_1, x_2, ..., x_n))$$

Suppose l is not a neighbor of 1. All terms (and thus G functions) involving l must be null. The G functions are thus not null only if the variables form a clique on the graph.

8.8 Markov Blanket in an undirected graph

8.8.1 Definition

The Markov Blanket $MB(i)$ of a node i is the smallest set of nodes $MB(i)$ such that $X_i \perp\!\!\!\perp X_R | X_{MB(i)}, with R = V \setminus (MB(i) \cup i)$ or equivalently such that $p(X_i | X_{\setminus i}) = p(X_i | X_{MB(i)})$.

For a Markov Random field the Markov blanket of X_i are its neighbors on G:

$$X_{MB(i)} = X_{N[i]}$$

8.9 Moralization

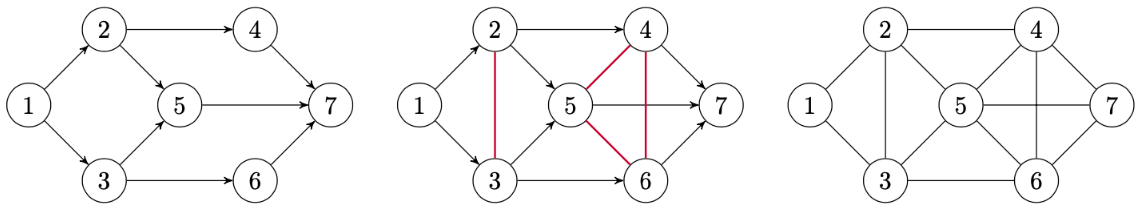
For a given oriented graphical model

- is there an unoriented graphical model which is equivalent?
- is there a smallest unoriented graphical which contains the oriented graphical model?

$p(x) = \frac{1}{Z} \prod_c \psi(x_c)$ vs $p(x) = \prod_i p(x_i \mid x_{\pi(i)})$

8.10 Moralization

Given a directed graph G, its moralized graph G_M is obtained by 1. For any node i, add undirected edges between all its parents 2. Remove the orientation of all the oriented edges



8.11 Moralization

8.11.1 Proposition

If a probability distribution factorizes according to a directed graph G then it factorizes according to the undirected graph G_M.

A distribution that factorizes according to a directed model is a Gibbs distribution for the cliques $C_i = \{i\} \cup \pi(i)$. As a consequence, it factorizes according to an undirected graph in which C_i are cliques.

8.12 Ising model

- The Ising model is an example of an MRF that arose from statistical physics.
- It was originally used for modeling the behavior of magnets.

Let $x_s \in \{-1, +1\}$ represents the spin of an atom, which can either be spin down or up.

In some magnets, called ferro-magnets, neighboring spins tend to line up in the same direction, whereas in other kinds of magnets, called anti-ferromagnets, the spins "want" to be different from their neighbors.

8.13 Ising model Gibbs distribution

Consider a graph with pairwise clique potential:

$$\psi_{st}(x_s, x_t) = e^{w_{st} x_s x_t}$$

where w_{st} is the coupling strength between neighboring nodes s and t.

The log probability is then

$$p(x) = \frac{1}{Z} e^{\sum_{s \sim t} w_{st} x_s x_t} = \frac{1}{Z} e^{\frac{1}{2} x^T W x}$$

If $w_{st} = \beta > 0$, we get high probability if neighboring states agree.

8.14 Ising model Gibbs distribution with external field

Sometimes there is an external field, which is an energy term which is added to each spin.

This can be modelled using a local energy term of the form $b^T x$, where $b = (b_s)_s$ is sometimes called a bias term.

The modified distribution is given by

$$p(x) = \frac{1}{Z} e^{\frac{1}{2} x^T W x + b^T x}$$

8.15 Links with Gaussian

Distribution looks similar to a Gaussian but

8.15.1 Beware normalization constant

- in the case of Gaussians, the normalization constant, $Z = |2\pi\Sigma|$, requires the computation of a matrix determinant, which can be computed in O(D3) time,
- whereas in the case of the Ising model, the normalization constant requires summing over all 2^D bit vectors;

8.16 Exercice: Ising Model with $w_{st} = \beta$ and $b_s = \alpha$

Compute the conditional distribution

$$p(x_i = 1|x_{\setminus i})$$

and use it to design a Gibbs sampler.

8.17 Gibbs sampler for ising model

8.17.1 The conditional site probability

can be used to build a Gibbs sampler:

$$p(x_i = 1|x_{\mathrm{N}[i]}) = \frac{exp(\alpha + \beta \sum_{j \sim i} x_j)}{exp(\alpha + \beta \sum_{j \sim i} x_j) + exp(-\alpha - \beta \sum_{j \sim i} x_j)}$$

8.17.2 Gibbs sampler

1. Init the random field $x = \{x_i\}$
2. loop through sites

a. Pick a site i at random
b. Simulate from $p(x_i = 1|x_{\mathrm{N}[i]})$

8.18 R code for fetching neighbors of site x_{ij}

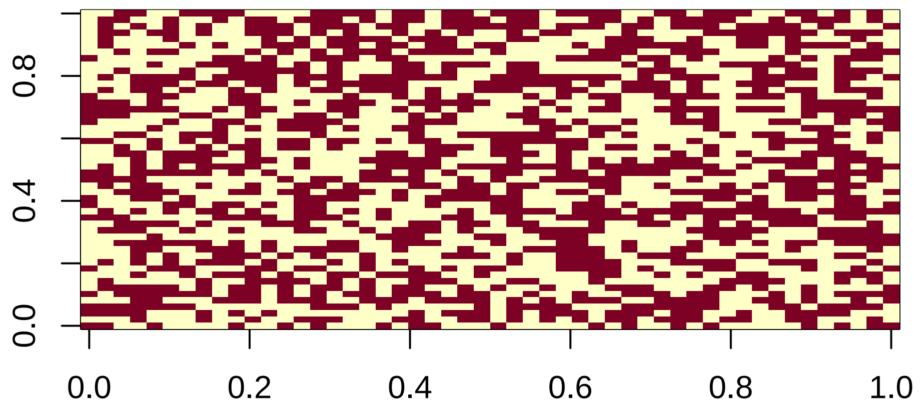
```
get_neighbours<-function(ij,n,p){
  # Get the 4 neighbours of a pixel i,j in a field of size nxp
  #             aj
  #             |
  #       il - ij - ir
  #             |
  #             uj
  j<-(ij-1) %/% n+1 ; i<-ij-(j-1)*n
  u<-ifelse(i+1>n,1,i+1)# V3: u j
  a<-ifelse(i-1<1,n,i-1)# V1: a j
  l<-ifelse(j-1<1,p,j-1)# V3: i l
  r<-ifelse(j+1>p,1,j+1)# V4: i r
  neighbours.coord<-matrix(c(u,a,i,i,j,j,l,r),4,2)
  neighbours.index<-neighbours.coord[,1]+
                  (neighbours.coord[,2]-1)*n
  return(neighbours.index)
}
```

8.19 R Gibbs sample for Ising model

```
gibbs.ising<-function(n=50,p=50,prob=0.5,alpha=0,
                      beta=1/2,nb.cycle=20){
  # Inititialisation
  MRF<-2*matrix(rbinom(n*p,size=1,prob=prob),n,p)-1
  np<-n*p
  cycle<-1
  while(cycle<=nb.cycle){
  cycle<-cycle+1
  walk.order<-sample(1:np,np,replace=FALSE)
  sapply(walk.order,function(ij){
     sum.Nij<-sum(MRF[get_neighbours(ij,n,p)])
     pXij.cond.Nij<- exp(alpha+beta*sum.Nij) / (exp(alpha+beta*sum.Nij)+exp(-alpha-beta*sum.N
     MRF[ij]<<- 2*rbinom(1,1,prob=pXij.cond.Nij)-1}
     )
  }
  return(MRF)
}
```

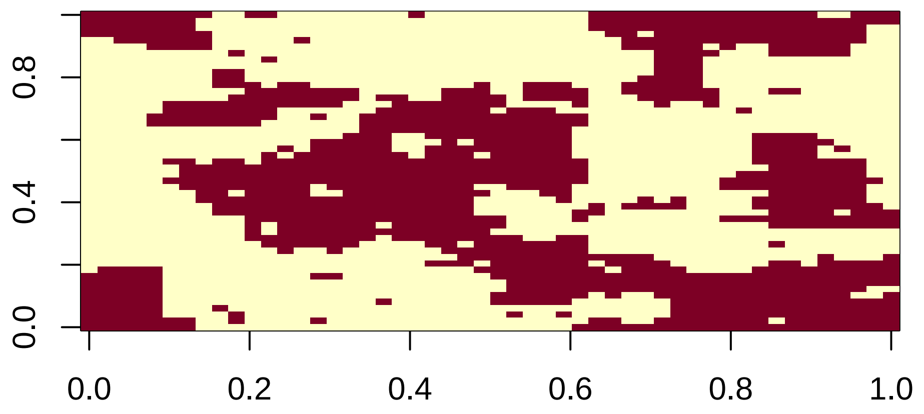
8.20 Ising illustration $\beta = 0$

```
image(gibbs.ising(beta=0))
```



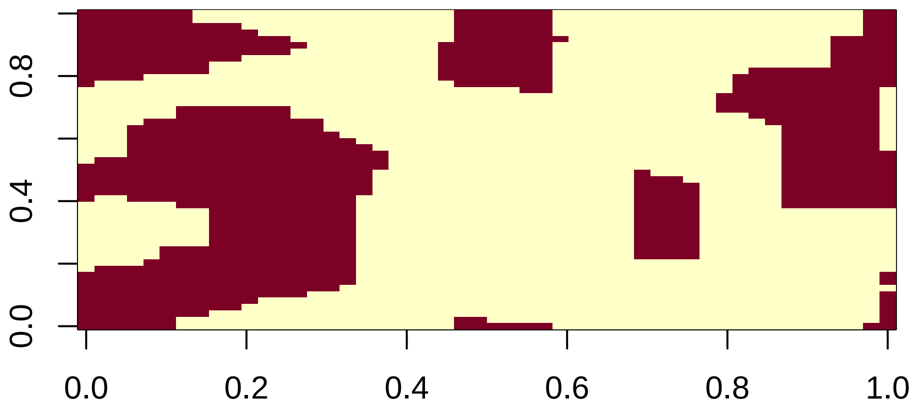
8.21 Ising illustration $\beta = 0.5$

```
image(gibbs.ising(beta=0.5))
```



8.22 Ising illustration $\beta = 3$

```
image(gibbs.ising(beta=3))
```



8.23 Hopfield networks

A Hopfield network (Hopfield 1982) is a fully connected Ising model with a symmetric weight matrix, $W = W^T$

8.24 Boltzmann machine

A fully connected graph with Bernoulli random variable X_i at each node i whith parameter θ_i given by a logistic regression on the other variables:

$$\text{logit}\,(\theta_i) = \alpha_i + \sum_{i \neq j} w_{ij} X_j$$

The weight are symetric and $w_{ii} = 0$.

The joint distribution is

$$p(x) = \frac{1}{Z} \exp \sum_i \alpha_i x_i + \sum_{i<j} w_{ij} x_i x_j$$

- Boltzmann machine are used to "learn" distributions for prediction or summary.
- Estimation of the parameters is not trivial because of the partition function.

8.25 Boltzmann machine estimation of the parameters when the graph is known

- Assuming observation $x_i = (x_{i1}, x_{i2}, ..., x_{ip}) \in \{0, 1\}^p$, with $i = 1, ..., N$. The log-likelihood is
- to avoid handling the bias terms α_i we assume a vetex 0 and condition w.r.t. $x_0 = 1$

$$
\begin{aligned}
L(W) &= \sum_i \log P_W(X_i = x_i), \\
&= \sum_i \sum_{(j,k)\in E} w_{jk} x_{ij} x_{ik} - \log Z(W)
\end{aligned}
$$

8.25.1 The gradient of the log-likelihood

$$\frac{\partial L(W)}{\partial w_{jk}} = \sum_i x_{ij} x_{ik} - N \frac{\partial \log Z(W)}{\partial w_{jk}}$$

where $\frac{\partial \log Z(W)}{\partial w_{jk}} = \frac{1}{Z(W)} \frac{\partial Z(W)}{\partial w_{jk}} = \sum_{x\in\mathcal{X}} x_j x_k P_W(x) = E_W[X_J X_K]$

8.26 Boltzmann machine estimation of the parameters when the graph is known

Setting the gradient to zero gives

$$\hat{E}(X_j X_k) - E_W(X_j X_k) = 0$$

where $\hat{E}(X_j X_k) = \frac{1}{N} \sum_i x_{ij} x_{ik}$

To find the maximum likelihood estimates,

- we can use gradient search or Newton methods.
- However the computation of the expectation is usually not possible

8.27 Boltzmann machine estimation of the parameters when the graph is known

- The mean field approximation estimates $E_W(X_j X_k)$ by $E_W(X_j) E_W(X_k)$, and replaces the input variables by their means, leading to a set of nonlinear equations for the parameters w_{jk}.
- To obtain near-exact solutions, Gibbs sampling (Section is used to approximate $E_W(X_j X_k)$ by successively sampling from the estimated model probabilities $P_W(X_j | X_{-j})$.
- ...

8.28 Boltzmann machine with Hidden Nodes

- We assume 2 types of variables ($\mathcal{V}$ visible and $\mathcal{H}$ hidden)

The Log-likelihood for K samples is

$$\begin{aligned} L &= \sum_i \log P(X_\mathcal{V} = x_{\mathcal{V}_i}) \\ &= \sum_i \log \sum_{x_\mathcal{H}} P(X_\mathcal{V} = x_{\mathcal{V}_i}, X_\mathcal{H} = x_\mathcal{H}) \end{aligned}$$

$$L = \sum_i \left(\log \sum_{x_\mathcal{H}} \exp \sum_{(j,k)\in E} w_{jk} x_{ij} x_{ik} - \log Z \right)$$

where the sum over $x_\mathcal{H}$ means that we are summing over all possible $\{0,1\}$ values for the hidden units.

8.29 Boltzmann machine with Hidden Nodes

The gradient of the log-likelihood with respect to w_{jk} is:

$$\frac{\partial L(W)}{\partial w_{jk}} = \sum_{i=1}^{N} \underbrace{\mathbb{E}_{P_W(X_\mathcal{H}|X_\mathcal{V}=x_\mathcal{V}^{(i)})}[X_j X_k]}_{\text{Expected sufficient statistic under posterior}} \quad - \quad \underbrace{\mathbb{E}_{P_W(X)}[X_j X_k]}_{\text{Model expectation (from}} \quad \log Z(W))$$

Noticing that the numerator is:

$$\sum_{x_\mathcal{H}} x_j x_k \cdot \frac{\exp\left(\sum_{(a,b)\in E} w_{ab} x_a x_b\right)}{Z(W)} = \mathbb{E}_W \left[X_j X_k \cdot \mathbb{I}\{X_\mathcal{V} = x_\mathcal{V}^{(i)}\} \right] = P_W(X_j = 1, X_k = 1, X_\mathcal{V} = x_\mathcal{V}^{(i)})$$

and the denominator is:

$$\sum_{x_\mathcal{H}} \frac{\exp\left(\sum_{(a,b)\in E} w_{ab} x_a x_b\right)}{Z(W)} = P_W(X_\mathcal{V} = x_\mathcal{V}^{(i)})$$

8.30 Boltzmann machine with Hidden Nodes

$$
\begin{aligned}
\frac{\partial L}{\partial w_{jk}} &= \sum_i \left(\sum_i P(X_j = X_k = 1 \mid X_{\mathcal{V}} = x_{\mathcal{V}_i}) - P(X_j = X_k = 1) \right) \\
&= \sum_i E_W(X_j X_k | X_{\mathcal{V}_i}) - E_W(X_j X_k)
\end{aligned}
$$

8.31 Boltzmann machine parameter estimation with hidden nodes

8.31.1 Gibbs sampling

Each part of the sum can be estimated via simulation (e.g. Gibbs sampler):

- unconditionned network
- network with fixed $x_{\mathcal{V}_k}$ (clamped nodes)

The gradient is used for small steps before re-estimation

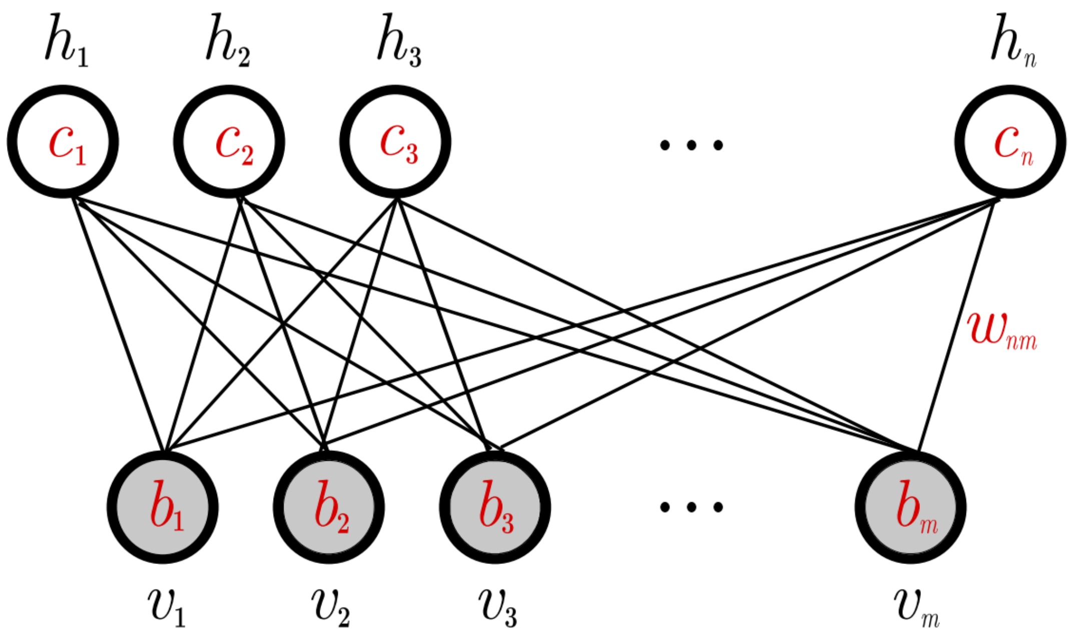
8.32 Boltzmann machine parameter estimation with hidden nodes

8.32.1 Variational approach (mean field approximation)

- Noting $\theta_k = P(X_k = 1)$ the joint distribution $P(X_j = X_k = 1)$ is approximated by $\theta_j \theta_k$
- Problem reduces in estimating θ_k
- Replacing input variables by their means
- And solving the system of non linear equations
  - $\operatorname{logit}(\theta_k) = \alpha_k + \sum_{j \neq k} w_{jk} \theta_j \theta_k$ in w_{jk}

8.33 Restricted Boltzmann machine

- one layer of visible units and one layer of hidden units with no connections within each layer.
- same generic form as a single hidden layer neural network



8.34 Restricted Boltzmann machine

Let denote $v = X_{\mathcal{V}_i}$ and h the observed value of the hidden nodes conditionnaly to v.

We have conditional independance of one layer with respect to the other

$$p(\boldsymbol{h} \mid \boldsymbol{v}) = \prod_{i=1}^{n} p(h_i \mid \boldsymbol{v}) \text{ and } p(\boldsymbol{v} \mid \boldsymbol{h}) = \prod_{j=1}^{m} p(v_j \mid \boldsymbol{h}) \ .$$

and

$$p(\boldsymbol{v}) = \sum_{\boldsymbol{h}} p(\boldsymbol{v},\boldsymbol{h}) = \frac{1}{Z}\sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v},\boldsymbol{h})} = \frac{1}{Z}\sum_{h_1}\sum_{h_2}\cdots\sum_{h_n} e^{\sum_{j=1}^{m} b_j v_j} \prod_{i=1}^{n} e^{h_i\left(c_i+\sum_{j=1}^{m} w_{ij}v_j\right)}$$

$$= \frac{1}{Z} e^{\sum_{j=1}^{m} b_j v_j} \sum_{h_1} e^{h_1\left(c_1+\sum_{j=1}^{m} w_{1j}v_j\right)} \sum_{h_2} e^{h_2\left(c_2+\sum_{j=1}^{m} w_{2j}v_j\right)} \cdots \sum_{h_n} e^{h_n\left(c_n+\sum_{j=1}^{m} w_{nj}v_j\right)}$$

$$= \frac{1}{Z} e^{\sum_{j=1}^{m} b_j v_j} \prod_{i=1}^{n}\sum_{h_i} e^{h_i\left(c_i+\sum_{j=1}^{m} w_{ij}v_j\right)} = \frac{1}{Z}\prod_{j=1}^{m} e^{b_j v_j} \prod_{i=1}^{n}\left(1+e^{c_i+\sum_{j=1}^{m} w_{ij}v_j}\right)$$

8.35 Restricted Boltzmann machine

8.35.1 Parallel Gibbs sampling

- the variables in each layer are independent of one another, given the variables in the other layers.
- Hence they can be sampled together, using the conditional probabilities

$$p(H_i = 1 \mid \boldsymbol{v}) = \mathrm{sig}\left(\sum_{j=1}^{m} w_{ij}v_j + c_i\right)$$

$$p(V_j = 1 \mid \boldsymbol{h}) = \mathrm{sig}\left(\sum_{i=1}^{n} w_{ij}h_i + b_j\right)$$

8.36 Training RBM using Gibbs sampling

The gradient of the likelihood for ℓ obs. is

$$\frac{1}{\ell}\sum_{\boldsymbol{v}\in S}\frac{\partial \ln\mathcal{L}(\boldsymbol{\theta}\,|\,\boldsymbol{v})}{\partial w_{ij}} = \frac{1}{\ell}\sum_{\boldsymbol{v}\in S}\left[-\mathbb{E}_{p(\boldsymbol{h}\,|\,\boldsymbol{v})}\left[\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial w_{ij}}\right] + \mathbb{E}_{p(\boldsymbol{h},\boldsymbol{v})}\left[\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial w_{ij}}\right]\right]$$
$$= \frac{1}{\ell}\sum_{\boldsymbol{v}\in S}\left[\mathbb{E}_{p(\boldsymbol{h}\,|\,\boldsymbol{v})}\left[v_i h_j\right] - \mathbb{E}_{p(\boldsymbol{h},\boldsymbol{v})}\left[v_i h_j\right]\right]$$

Using one Gibbs sampler round going from v_j and h_k to v'_j and h'_k:

- The first exectation could be roughly approximated by the product $v_j h_k$
- The second expectation could be approximated with one round of Gibbs sampling by the product $v'_j h'_k$

8.37 Training RBM using Gibbs Sampling

1. Take a training sample v, compute the probabilities of the hidden units and sample a hidden activation vector h from this probability distribution.
2. Compute the outer product of v and h to approximate $\mathbb{E}_{p(h|v)}[v_j h_k]$
3. From h, sample a reconstruction v' of the visible units, then resample the hidden activations h' from this. (Gibbs sampling step)
4. Compute the outer product of v' and h' to approximate the expectation $\mathbb{E}_{p(h,v)}[v_j h_k]$.
5. Update to the weight matrix and the bias

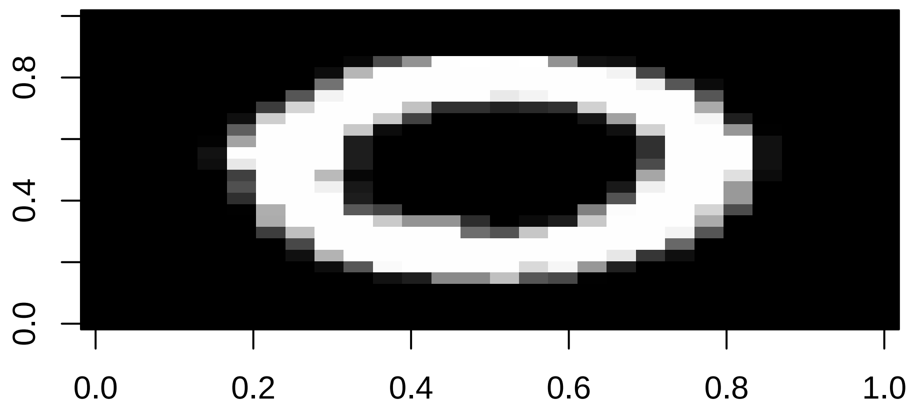
$$\Delta w_{jk} = \epsilon(v_j h_k - v'_j h'_k)$$

$$\Delta \alpha_j = \epsilon(v_j - v'_j),\ \Delta \alpha_h = \epsilon(h_k - h'_k),$$

8.38 Github example from TimoMatzen

8.38.1 The MNIST Data

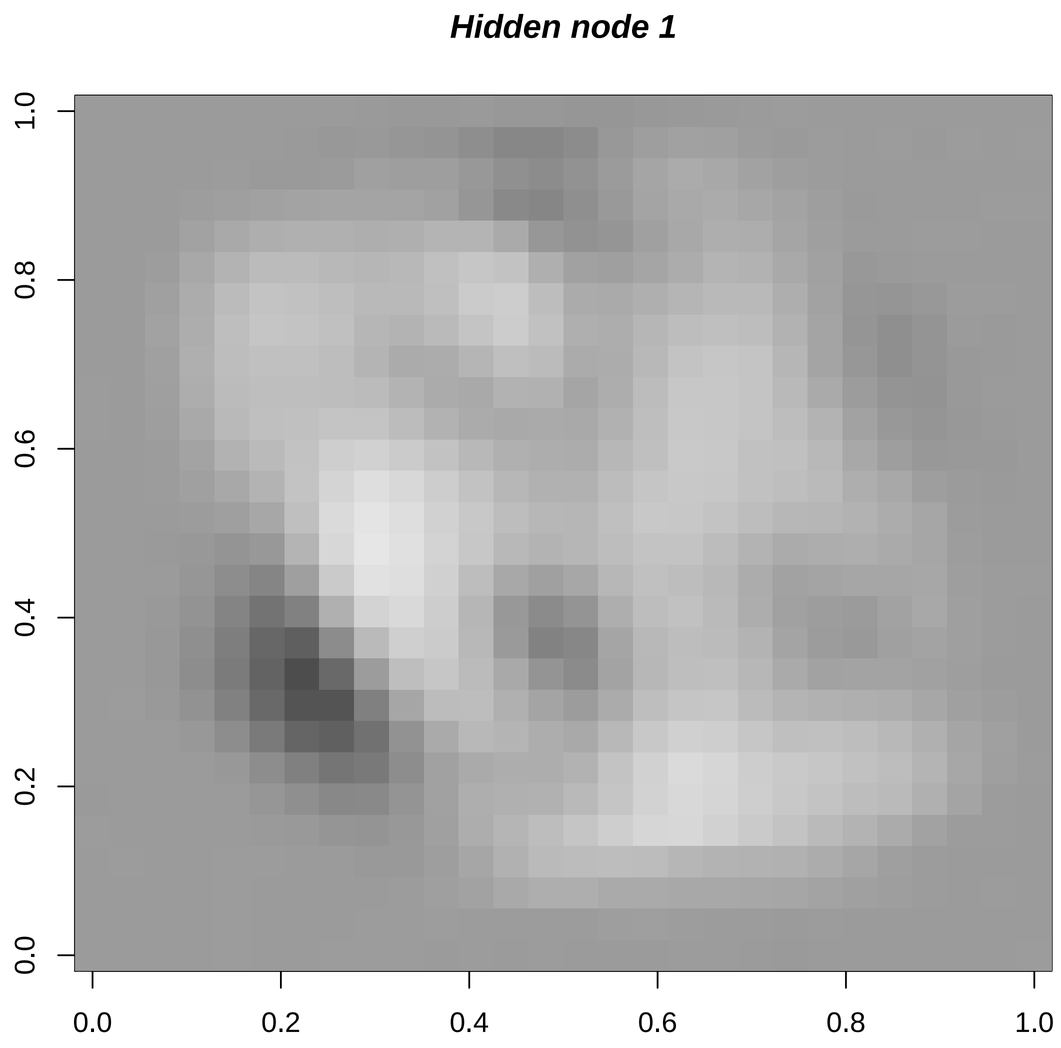
```r
# TimoMatzen
library(RBM)
# Load the MNIST data
data(MNIST)
image(matrix(MNIST$trainX[2, ], nrow = 28),
      col = grey(seq(0, 1, length = 256)))
```



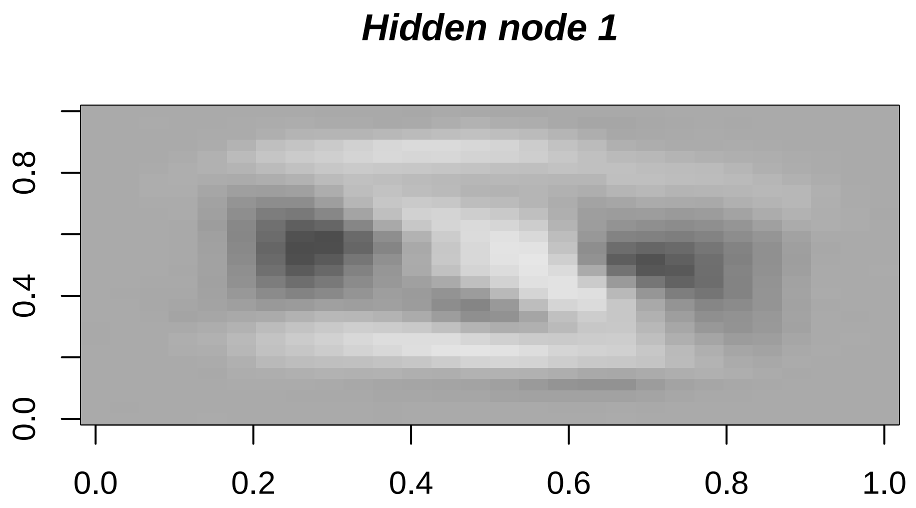
8.39 RBM: Github example from TimoMatzen: Learning

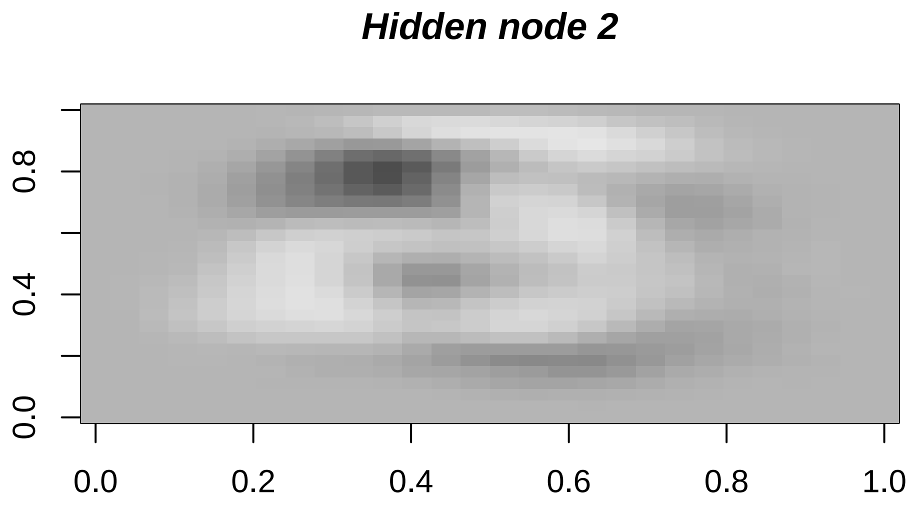
```r
train <- MNIST$trainX
nb.hidden.units<-10
modelRBM <- RBM(x = train, n.iter = 1000,
                n.hidden = nb.hidden.units,
                size.minibatch = 10)
```

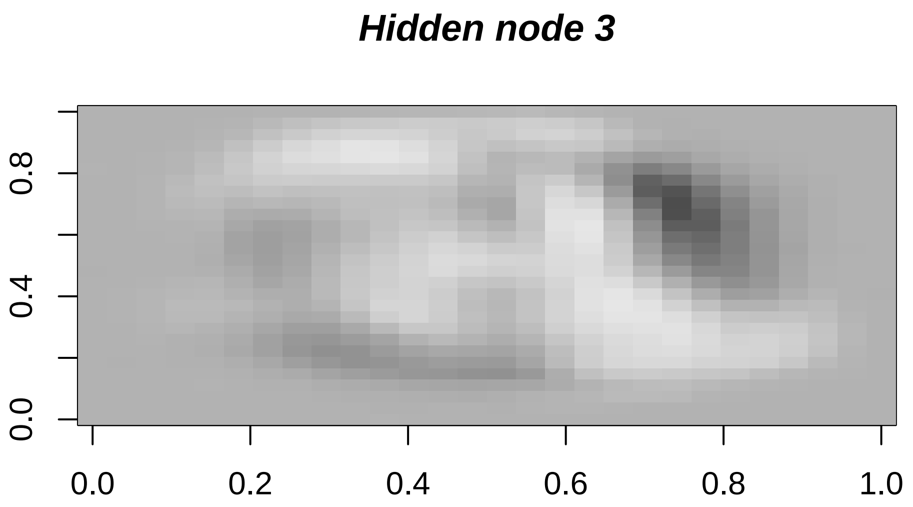
8.40 RBM: Reconstruction example from TimoMatzen Hidden units

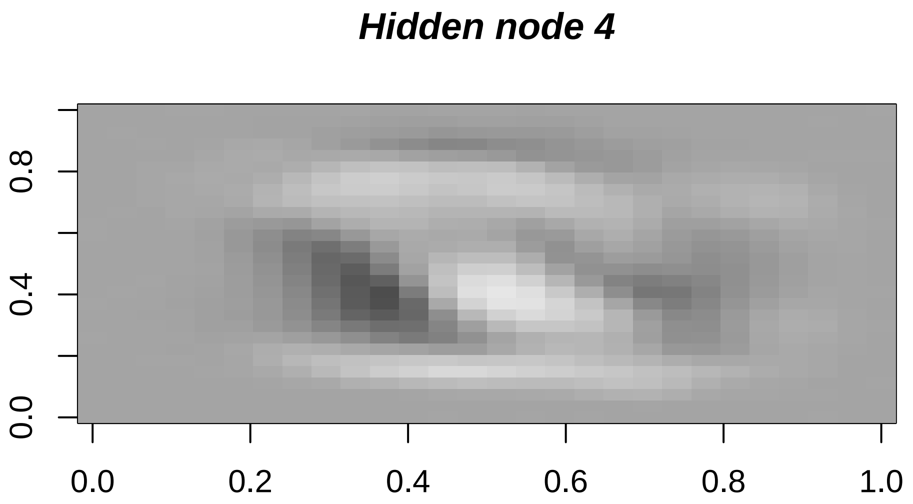


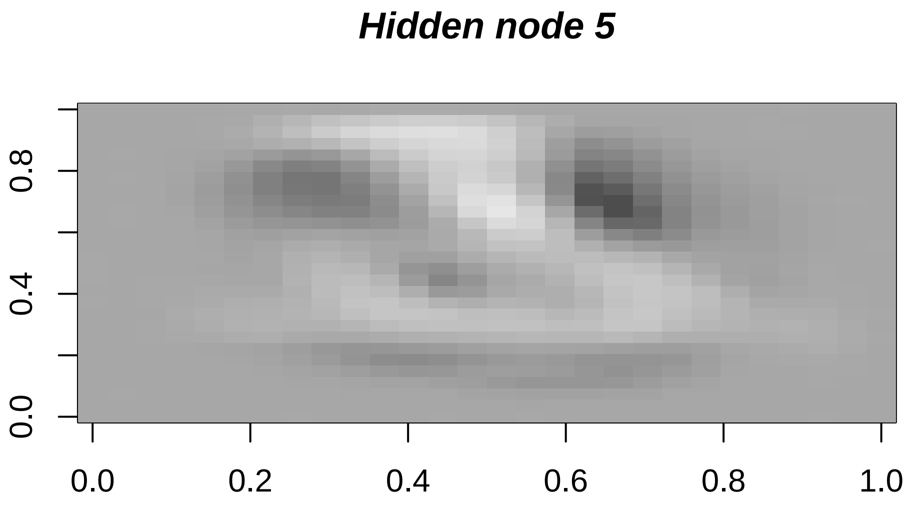
```
plot.weights<-modelRBM$trained.weights[-1,-1]
for (i in 1:nb.hidden.units) {
                image(matrix(plot.weights[,i],
                             nrow = sqrt(ncol(train))),
                      col = grey.colors(255))
                title(main = paste0("Hidden node ", i), font.main = 4)
                plot.counter <- 0
            }
```



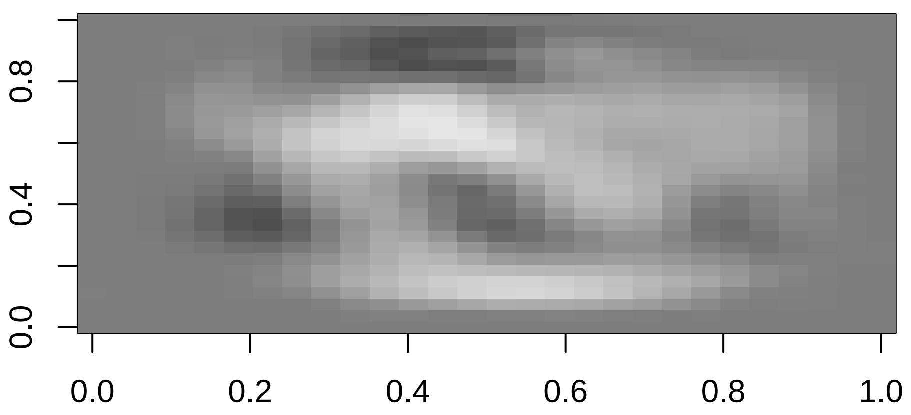



Hidden node 3
0.0
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0

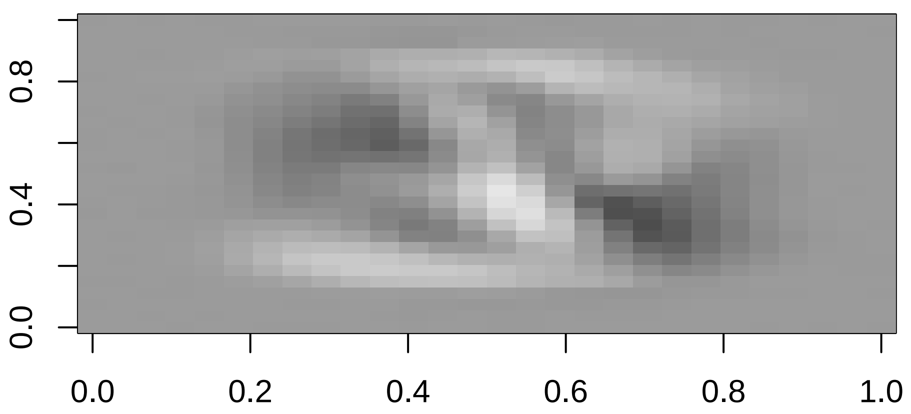

Hidden node 4
0.0
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0


Hidden node 5
0.0
0.4
0.8
0.0
0.2
0.4
0.6
0.8
1.0

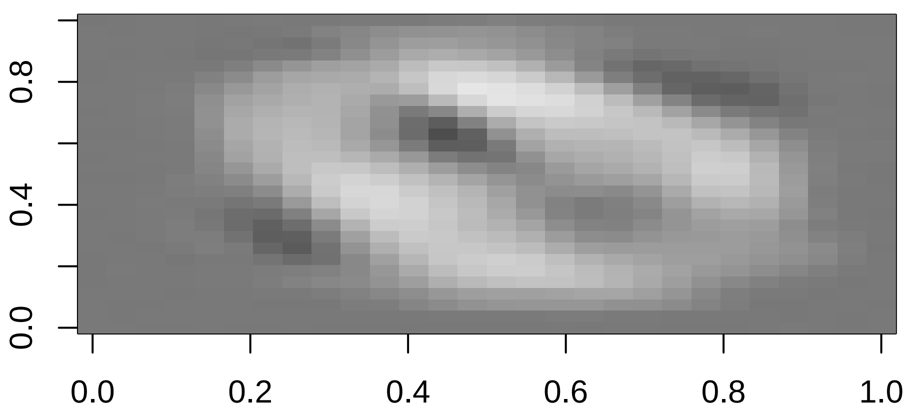
Hidden node 6

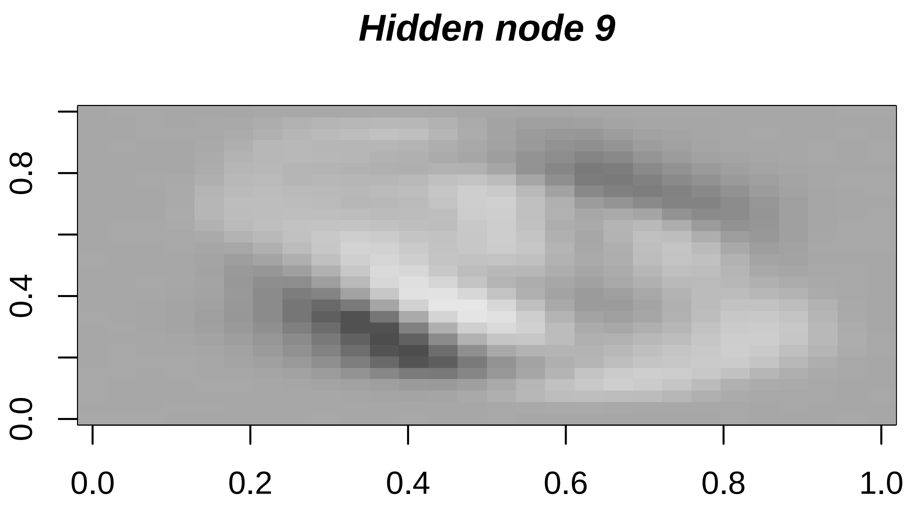


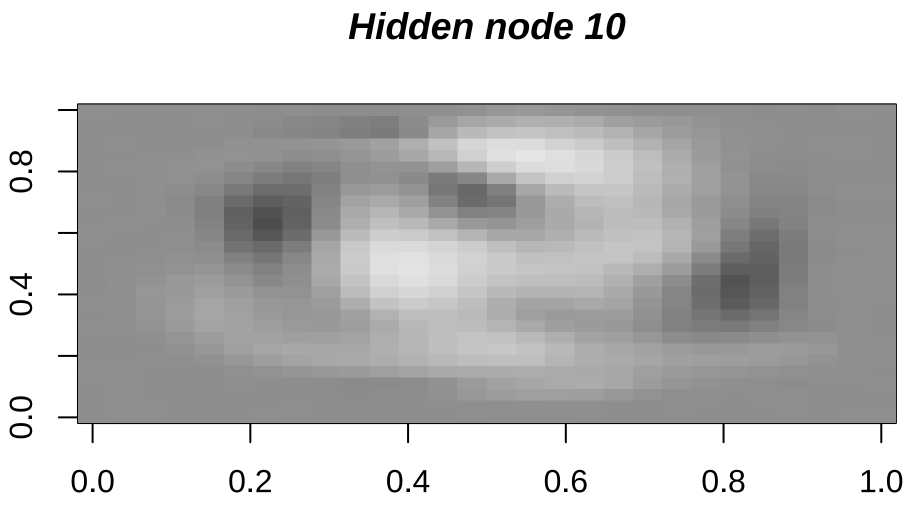
Hidden node 7



Hidden node 8

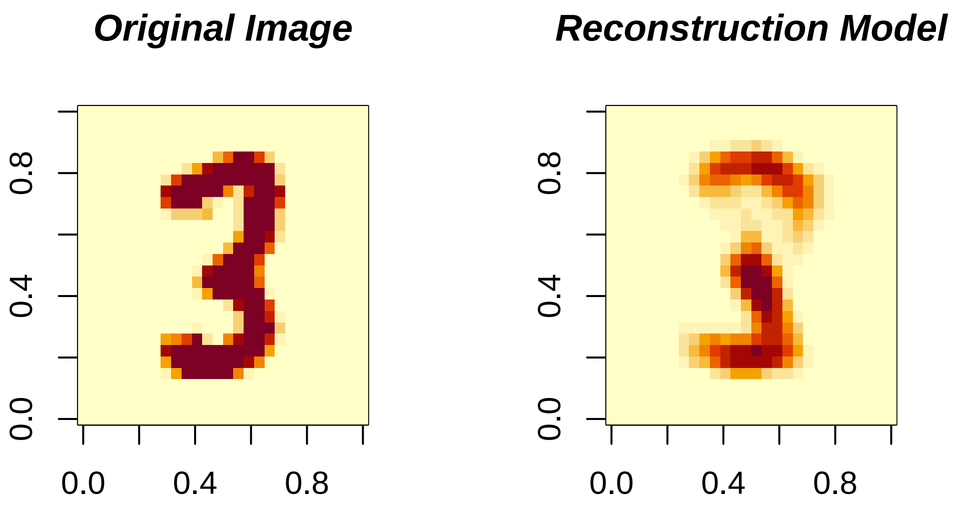






8.41 Reconstruction example from TimoMatzen

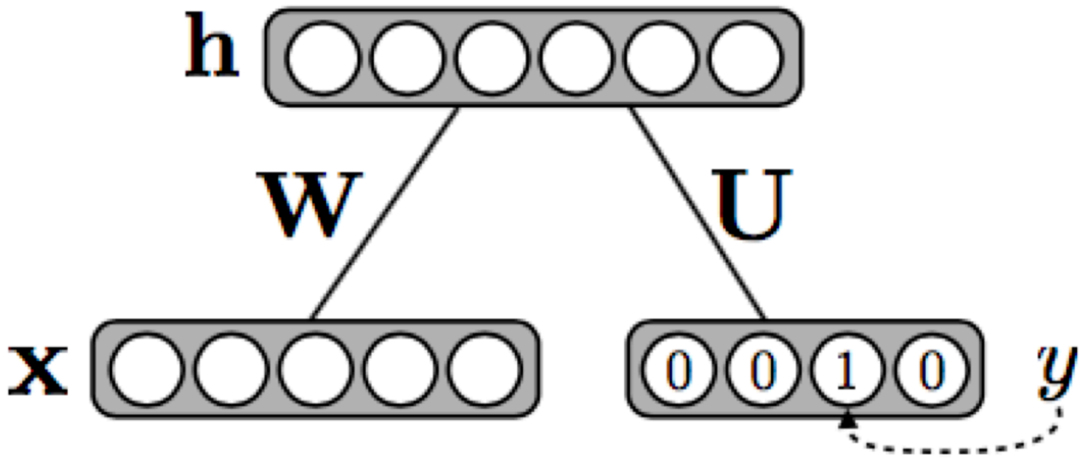
```
# Get the test data from MNIST
test <- MNIST$testX
# Reconstruct the image with modelRBM
ReconstructRBM(test = test[6, ], model = modelRBM)
```



8.42 Classification with Boltzmann Machine

Using 2 type of visible units:

- the pixel values
- the (binarized) labels.



8.43 Classification example from TimoMatzen

8.43.1 Training

```r
data(MNIST)
# First get the train labels of MNIST
TrainY <- MNIST$trainY
# This time we add the labels as the y argument
modelClassRBM <- RBM(x = train, y = TrainY,
                     n.iter = 3000, n.hidden = 200,
                     size.minibatch = 10)
```

8.44 Classification example from TimoMatzen

8.44.1 Testing

```r
# First get the test labels of MNIST
TestY <- MNIST$testY
# Give our ClassRBM model as input
PredictRBM(test = test, labels = TestY, model = modelClassRBM)$Accuracy
#[1] 0.852
```

8.45 Reference about RBM

- Hinton, G. A Practical Guide to Training Restricted Boltzmann Machines (2010) : https://www.cs.toronto.edu/~hinton/absps/guideTR.pdf

8.46 Exponential family

8.46.1 Discrete Case

$$p(x;\theta) = \exp\left(\theta^t\phi(x) - A(\theta)\right)$$

where $A(\theta) = \log(Z(\theta)) = \log\left(\sum_x \exp\left(\theta^t\phi(x)\right)\right)$

8.46.2 Examples

Gaussian, Bernoulli, Binomial, Poisson, Exponential, Weibull, Laplace, gamma, beta, multinomial, Wishart distributions

8.47 Derivatives of the log partition function

$$\frac{\partial A(\theta)}{\partial \theta} = \frac{1}{Z(\theta)} \sum_x \phi(x) \exp\left(\theta^t \phi(x)\right) = E[\phi(x)]$$

$$\begin{aligned}
\frac{\partial^2 A(\theta)}{\partial \theta \partial \theta^t} &= \frac{\partial}{\partial \theta^t} \sum_x \phi(x) \exp\left(\theta^t \phi(x) - A(\theta)\right) \\
&= \sum_x p(x;\theta)\phi(x)(\phi(x)^t - E[\phi(x)]^t) \\
&= E[\phi(x)\phi(x)^t] - E[\phi(x)]^t E[\phi(x)] \\
&= var[\phi(x)]
\end{aligned}$$

8.48 Gradient ascent for maximizing the log-likelihood

The Log-likelihood for K samples is

$$L = \sum_k \log p(x_k;\theta)$$

8.48.1 Gradient ascent

$$\begin{aligned}
\frac{\partial L}{\partial \theta} &= \sum_k \frac{\partial}{\partial \theta}(\left(\theta^t \phi(x_k) - A(\theta)\right)) \\
&= \sum_k (\phi(x_k) - E[\phi(x)]) \\
&= K\left(\hat{E}[\phi(x)] - E[\phi(x)]\right)
\end{aligned}$$

$$\theta^{q+1} = \theta^{q+1} + \epsilon \frac{\partial L}{\partial \theta}$$

At the maximum (local) $\hat{E}[\phi(x)] = E[\phi(x)]$

8.49 Undirected models are members of exponential family.

If we consider $p(x;\theta) \propto \prod_{c\in\mathcal{C}} \Psi(x_c)$

it can be rewritten as exponential family

$$p(x;\theta) = \exp\left(\theta^t\phi(x) - A(\theta)\right)$$

where

1. **The potential functions** $\Psi_C(X_C)$ can be chosen in an exponential form:

$$\Psi_C(X_C) = \exp\left(\theta_C^T\phi_C(X_C)\right),$$

which makes the joint density $p(X)$ naturally fall within the exponential family. 2. **Sufficient Statistics**: The sufficient statistics $\phi_C(X_C)$ of the cliques directly capture the local dependencies. 3. **Log-partition**: The normalization constant Z corresponds to the exponential of the log-partition function $A(\theta)$ of the exponential family.

8.49.1 Example: Ising Model

The Ising model, used in statistical physics and machine learning, is a specific case of a Markov field where:

$$p(X) = \frac{1}{Z}\exp\left(\sum_{(i,j)\in E}\theta_{ij}X_iX_j + \sum_{i\in V}\theta_iX_i\right).$$

The density clearly follows the structure of an exponential family.

9 Gaussian Graphical Model

9.1 Gaussian Graphical Model

9.1.1 Density

A random vector $x \in \mathbb{R}^p$ is distributed according to the multivariate Gaussian distribution $\mathcal{N}(\mu,\Sigma)$ where μ is the mean vector and Σ the covariance matrix is defined by

$$f(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}}exp\{-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)\}$$

9.1.2 Precision matrix

The inverse covariance matrix, also known as the precision matrix or the concentration matrix, $K = \Sigma^{-1}$

9.2 Link to Exponential family

9.2.1 Canonical parameters and sufficient statistics

$$\theta = \{-\mu^t K, K\}$$

$$\phi(x) = \{x, xx^t\}$$

9.3 Link to Markov Random Field

9.3.1 Factorization

$$f(x) \propto \prod_j \Psi(x_j) \prod_{j<k} \Psi(x_{jk})$$

where $\Psi(x_j) = (exp(-\sum_k \mu_k K_{kj})x_j$ and $\Psi(x_{jk}) = exp(x_j K_{jk} x_k)$

9.3.2 Markov property

From the factorization it is straightforward that

$$K_{ij} = 0 \Leftrightarrow X_i \perp\!\!\!\perp X_j \mid X_{V\setminus\{i,j\}}$$

Graph $G = \{V, E\}$ where $K_{ij} = 0 \Leftrightarrow \forall (i,j) \notin E$ describes the sparsity pattern of the concentration matrix.

9.4 Gaussian distribution and Conditional independence

$$\begin{bmatrix} x_A \\ x_B \end{bmatrix} \sim \mathcal{N}_p \left(\begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix} ; \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix} \right)$$

We have the following property

$$(x_A \mid x_B = b) \sim \mathcal{N}_p \left(\mu_A + \Sigma_{AB}\Sigma_{BB}^{-1}(b - \mu_B); \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA} \right).$$

The idea of the proof consists in computing the conditional density $f(x_A|x_B = b) = f_{A,B}(x_A, b)/f_B(b)$ knowning that both f_B and $f_{A,B}$ are multivariate gaussian.

9.5 Concentration matrix

The concentration matrix is $K := \Sigma^{-1}$. Using the partition of the multivariate vector in A and B, the Schur complement allows to compute

$$K_{AA} = (\Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA})^{-1}$$

which is exactly the inverse of the conditional covariance of $A|B$.

If $A = (x_1, x_2)^t$ and $B = (x_3, ..., x_p)^t$ then

$$K_{AA} = \begin{pmatrix} k_{11} & k_{12} \\ k_{12} & k_{22} \end{pmatrix} = \Sigma_{A|B}^{-1}$$

9.6 Concentration matrix

Thus the conditional covariance of $A|B$ expressed in terms of concentration becomes

$$\Sigma_{A|B} = \frac{1}{det(K_{AA})} \begin{pmatrix} k_{22} & -k_{12} \\ -k_{12} & k_{11} \end{pmatrix}$$

and the correlation of $x_1x_2|x_3...x_p$ is

$$\frac{-k_{12}}{\sqrt{k_{11}k_{22}}}.$$

9.7 Covariance selection

The task of computing the MLE for a (non-decomposable) GGM is called covariance selection (Dempster 1972).

$$\log L(K) = \log detK - tr(SK)$$

where $S = \frac{1}{N}\sum_{i=1}^{N}(xi - x)(x_i - \bar{x})^T$ is the empirical covariance matrix.

9.7.1 Exercice

Derive the equation of the log-likelihood

9.7.2 The gradient

$$\nabla \log L(K) = K^{-1} - S$$

9.8 Estimation of K when the graph structure is known

Interestingly, one can show that the MLE must satisfy the following property:

- $\Sigma_{st} = S_{st}$ if $G_{st} = 1$ or $s = t$
- $K_{st} = 0$ if $G_{st} = 0$, by definition of a GGM, i.e., the precision of a pair that are not connected must be 0.

9.8.1 Σ is a positive definite matrix completion of S

it retains as many of the entries in S as possible:

- corresponding to the edges in the graph
- subject to the required sparsity pattern on K, corresponding to the absent edges;
- the remaining entries in Σ are filled in so as to maximize the likelihood.

9.9 Estimation of K when the graph structure is known (Example)

Let us consider the example from (Hastie et al. 2009, p652) representing the cyclic structure, $X_1 - -X_2 - -X_3 - -X_4 - -X_1$, and the following empirical covariance matrix:

$$\mathbf{G} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} 10 & 1 & 5 & 4 \\ 1 & 10 & 2 & 6 \\ 5 & 2 & 10 & 3 \\ 4 & 6 & 3 & 10 \end{pmatrix}$$

The MLE is given by

$$\mathbf{\Sigma} = \begin{pmatrix} 10.00 & 1.00 & \mathbf{1.31} & 4.00 \\ 1.00 & 10.00 & 2.00 & \mathbf{0.87} \\ \mathbf{1.31} & 2.00 & 10.00 & 3.00 \\ 4.00 & \mathbf{0.87} & 3.00 & 10.00 \end{pmatrix}, \quad \mathbf{\Omega} = \begin{pmatrix} 0.12 & -0.01 & \mathbf{0} & -0.05 \\ -0.01 & 0.11 & -0.02 & \mathbf{0} \\ \mathbf{0} & -0.02 & 0.11 & -0.03 \\ -0.05 & \mathbf{0} & -0.03 & 0.13 \end{pmatrix}$$

9.10 Estimation of K when the graph is structure is unknown

By analogy to lasso one can define the following ℓ_1 penalized criterion:

$$J(K) = -\log detK + tr(SK) + \lambda \|K\|_1$$

where $\|K\|_1 = \sum_{st} |k_{st}|$

Several algorithms have been proposed for optimizing this objective (Yuan and Lin 2007; Banerjee et al. 2008; Duchi et al. 2008), although arguably the simplest is the one in (Friedman et al. 2008), which uses a coordinate descent algorithm similar to the shooting algorithm for lasso.

9.11 Graphical Lasso

The subgradient equation is

$$K^{-1} - S - \lambda Sign(K) = 0,$$

where $Sign(K_{jk}) = sign(K_{jk})$ if $K_{jk} \neq 0$, else $Sign(K_{jk}) \in [-1, 1]$ if $K_{jk} = 0$.

The graphical Lasso use regression to solve for K and its inverse $W = K^{-1}$ one row and column at a time.

9.12 Graphical Lasso

If we consider a partition of columns in two

1. $p-1$ first colums
2. last column p

we have by definition

$$\begin{pmatrix} W_{11} & w_{12} \\ w_{12}^t & w_{22} \end{pmatrix} \begin{pmatrix} K_{11} & K_{12} \\ K_{12}^t & k_{22} \end{pmatrix} = I$$

show that w_{12} can be regressed from W_{11}

$$w_{12} = -W_{11}\frac{K_{12}}{k_{22}} = W_{11}\beta$$

9.13 Graphical Lasso (from Hastie & Tibshirani)

1. Initialize $W = S + \lambda I$. The diagonal of W remains unchanged in what follows.
2. Loop through columns until convergence

   a. Partition the matrix W into part 1: all but the jth row and column, and part 2: the jth row and column.
   b. Solve the lasso type problem $W_{11}\beta - s_{12} + \lambda Sign(\beta) = 0$ using the cyclical coordinate-descent algorithm.
   c. Update $w_{12} = W_{11}\beta$

3. In the final cycle (for each j) solve for $K_{12} = -\beta K_{22}$, with $1/K_{22} = w_{22} - w_{12}^T\beta$

10 Appendix

10.1 Gibbs Sampling

Multi-stage Gibbs sampler: One step of the algorithm has p stages

1. Given $(X_1^n, \cdots, X_p^n)$ we sample X_1^{n+1} from $P(.|X_1^n, \cdots, X_p^n)$

2. Then sample X_2^{n+1} from $P(.|X_1^{n+1}, X_3^n, \cdots, X_p^n)$

j. Continuing we sample X_j^{n+1} from $P(.|X_1^{n+1}, \cdots, X_{j-1}^{n+1}, X_{j+1}^n, \cdots, X_p^n)$

p. In the last step we sample X_p^{n+1} from $P(.|X_1^{n+1}, \cdots, X_{p-1}^{n+1})$

10.2 Transition Matrix

Let A_j be the transition matrix corresponding to the j^{th} step of the multi-stage Gibbs sampler

$$A_j(x_1, x_2, \cdots, x_p; x_1', x_2', \cdots, x_p') = P(x_j'|x_1, x_2, \cdots, x_{j-1}, x_{j+1}, \cdots, x_p) \prod_{(i \neq j)} \delta(x_i - x_i')$$

The $\prod_{(i \neq j)} \delta(x_i - x_i')$ ensure that only site j can be different between the origin state $x_1, x_2, \cdots, x_p$ and the arrival state $x_1', x_2', \cdots, x_p'$.

Basically the matrix is sparse. For each configuration where x_j changes (two lines of A_j for ising model), there are two corresponding columns and thus 4 possible transitions. There are 2^p configuration, the transition matrix is $2^p \times 2^p$ and for each of the 2^{p-1} configurations without j there are 4 non zeros transitions thus $2^{2p} - 2^{p+1} = 2^{p+1}(2^{p-1} - 1)$ zeros entries in the matrix.

For a **randomized Gibbs sampler**, fix some probability distribution q_i on $\{1, 2, \cdots, p\}$. Given that we are in state $(X_1^n, \cdots, X_p^n)$, we first pick $i \in \{1, 2, \cdots, p\}$ according to this distribution. Then we sample X_i^{n+1} from $P(\cdot|X_1^n, \cdots, X_{i-1}^n, X_{i+1}^n, \cdots, Xp^n)$. The transition matrix for this algorithm is

$$A = \sum_j q_j A_j$$

10.3 Stationary distribution

Proposition: $P(x_1, x_2, \cdots, x_p)$ is the stationary distribution of the multi-stage Gibbs sampler and of the randomized Gibbs sample for any choice of the distribution q_i.

We only need to show that for all j, $A_j^T(x_1', x_2', \cdots, x_p', \bullet)P = P(x')$

$$
\begin{aligned}
A_j^T(x',\bullet)P = & \quad \cdots \sum_{x_1,x_2,\cdots,x_p} P(x_1,x_2,\cdots,x_p)A_j(x_1,x_2,\cdots,x_p;x'_1,x'_2,\cdots,x'_p) \\
= & \sum_{x_1,x_2,\cdots,x_p} P(x_1,x_2,\cdots,x_p)P(x'_j|x_1,x_2,\cdots,x_{j-1},x_{j+1},\cdots,x_p)\prod_{(i\neq j)}\delta(x_i-x'_i) \\
= & \quad P(x'_j|x'_1,x'_2,\cdots,x'_{j-1},x'_{j+1},\cdots,x'_p)\sum_{x_j}P(x'_1,x'_2,\cdots,x'_{j-1},x_j,x'_{j+1},\cdots,x'_p) \\
= & \quad P(x'_j|x'_1,x'_2,\cdots,x'_{j-1},x'_{j+1},\cdots,x'_p)P(x'_1,x'_2,\cdots,x'_{j-1},x'_{j+1},\cdots,x'_p) \\
= & \quad P(x'_1,x'_2,\cdots,x'_p)
\end{aligned}
$$

11 Exercices

11.1 Exercices Unidirected Graphical Model

11.1.1 Conditional independence

Let consider three sets of discrete variables X, Y, Z. Show that if there exist two function F and G such that

$$P(X,Y,Z) = F(X,Z)G(Y,Z)$$

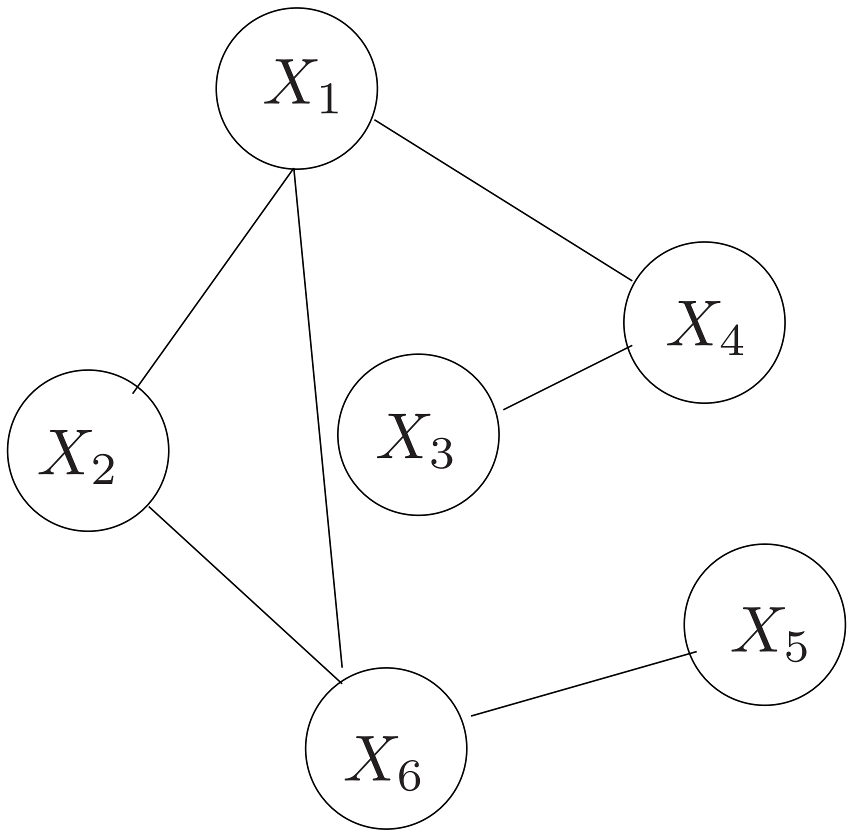
then

$$X \perp\!\!\!\perp Y \mid Z$$

11.2 Exercices Unidirected Graphical Model

11.2.1 Conditional independence (Exo 17.1 Elements of Stat)

For the Markov graph follow, list all of the implied conditional independence relations and find the maximal cliques.


X_1
X_2
X_3
X_4
X_5
X_6

11.3 Exercices Unidirected Graphical Model

11.3.1 Independences to graph (Exo 17.2 Elements of Stat)

Consider random variables X_1, X_2, X_3, X_4. In each of the following cases draw a graph that has the given independence relations:

a. $X_1 \perp\!\!\!\perp X_3 \mid X_2$, and $X_2 \perp\!\!\!\perp X_4 \mid X_3$.
b. $X_1 \perp\!\!\!\perp X_4 | X_2, X_3$ and $X_2 \perp\!\!\!\perp X_4 \mid X_1, X_3$.
c. $X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3, X_1 \perp\!\!\!\perp X_3 | X_2, X_4$ and $X_3 \perp\!\!\!\perp X_4 \mid X_1, X_2$.

12 Exercices on Directed Graphical Models

12.1 Exercice Gaussian Bayesian Network

12.1.1 Data

Let consider the following graph $x_1 \rightarrow x_2 \rightarrow x_3$ where

- $\mathbb{E}[x_1] = b_1$, $\mathbb{E}[x_2] = b_2$, $\mathbb{E}[x_3] = b_3$
- $x_1 = b_1 + z_1$
- $x_2 = b_2 + (x_1 - b_1) + z_2$
- $x_3 = b_3 + 1/2(x_2 - b_2) + z_3$
- $\sigma_1 = \sigma_2 = \sigma_3 = 1$,

12.1.2 Problem

- Write the Adajcency matrix with topological ordering
- Derive the mean vector and covariance matrix of the random vector
- Simulate Gaussian data
- Estimtate the parameters from your simulation
- What improvment could you suggest ?

12.2 Exercice Directed GGM

- $\mu^T = (0, 1, 2)$
- $diag(S) = (1, 1, 1)$
- $W = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1/2 & 0 \end{pmatrix}$

12.3 Exercice Directed GGM

We can observe that the precision matrix has the some support as W

```
n=1000
mu=c(0,1,2)
sigma=c(1,1,1)
W=matrix(c(0,1,0,0,0,1/2,0,0,0),3,3)
U=solve(diag(rep(1,3))-W)
S=diag(sigma)
Sigma=U%*%S^2%*%t(U)
solve(Sigma)
```

```
     [,1]  [,2] [,3]
[1,]    2 -1.00  0.0
[2,]   -1  1.25 -0.5
[3,]    0 -0.50  1.0
```

12.4 Exercice Directed GGM

12.4.1 First solution (direct)

```
library(mvtnorm)
Xprime=rmvnorm(n,mean=c(0,1,2),sigma=Sigma)
```

12.4.2 Second solution (constructive)

```
X=matrix(0,n,3)
Z=matrix(rnorm(n*3),n,3)
for (i in 1:n)
  for (j in 1:3)
    X[i,j]=mu[j]+sigma[j]*Z[i,j] + sum(W[j,]*(X[i,]-mu))
```

Click to go Back to Lecture

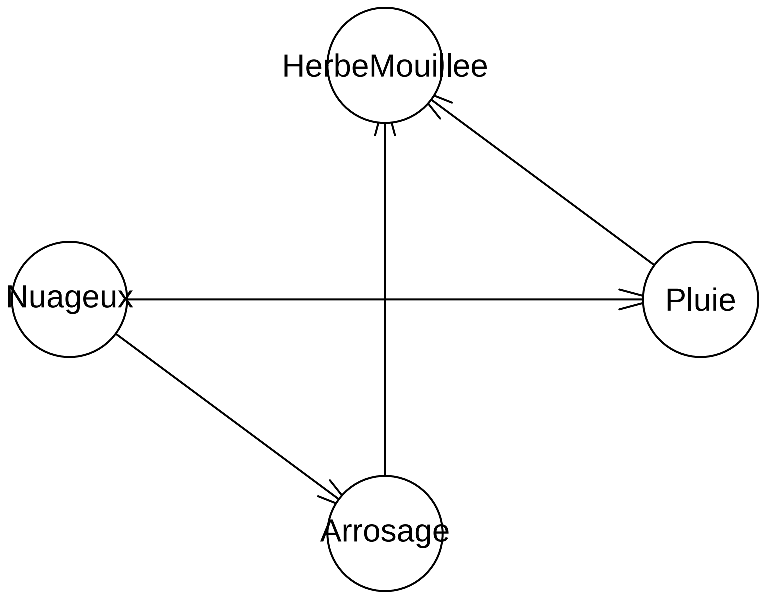
12.5 Sprinkler Exercice

Let us define the structure of the network

```
library(bnlearn)
library(visNetwork)
variables<-c("Nuageux","Arrosage","Pluie","HerbeMouillee")
net<-empty.graph(variables)
adj = matrix(0L, ncol = 4, nrow = 4, dimnames=list(variables, variables))
adj["Nuageux","Arrosage"]<-1
adj["Nuageux","Pluie"]<-1
adj["Arrosage","HerbeMouillee"]<-1
adj["Pluie","HerbeMouillee"]<-1
amat(net)=adj
```

12.6 Sprinkler Exercice

```
#plot.network(net) # for a nice html plot
plot(net)
```



12.7 Sprinkler Exercice

Simulate a sample according the model

12.8 Basic Simulation with using conditional probability tables

Function for one event (one line of dataframe)

```
NAPHM1<-function(n){
  N<-rbinom(1,size = 1,prob = 1/2)
  if (N==1)  {A<-rbinom(1,size = 1,prob = 0.1)} else {A<-rbinom(1,size =1,prob = 0.5)}
  if (N==1)  {P<-rbinom(1,size = 1,prob = 0.8)} else {P<-rbinom(1,size = ,1,prob = 0.2)}
 if (A+P==0)  HM<-rbinom(1,size = 1,prob = 0.1) else if
 (A+P==1) HM<-rbinom(1,size = 1,prob = 0.9) else
  HM<-rbinom(1,size = 1,prob = 0.99)
 X<-as.logical(c(N,A,P,HM))
}
```

12.9 Basic Simulation with using conditional probability tables

```
n<-1000
X<-data.frame(t(sapply(1:n,NAPHM1)))
names(X)<-c("Nuageux","Arrosage","Pluie","HerbeMouillee")
head(X)
```

```
  Nuageux Arrosage Pluie HerbeMouillee
1   FALSE    FALSE FALSE         FALSE
2   FALSE     TRUE  TRUE          TRUE
3    TRUE    FALSE FALSE         FALSE
4    TRUE    FALSE FALSE         FALSE
5   FALSE    FALSE FALSE         FALSE
6   FALSE    FALSE  TRUE          TRUE
```

12.10 Learning the parameters

```
mean(X$Nuageux) -> pNuageux
lapply(sousTableauxNuageux<-split(X,X$Nuageux),
       function(XsousTableau){mean(XsousTableau$Arrosage)})
lapply(sousTableauxNuageux<-split(X,X$Nuageux),
       function(XsousTableau){mean(XsousTableau$Pluie)})
lapply(sousTableauxNuageux<-split(X,X$Arrosage + X$Pluie),
       function(XsousTableau){mean(XsousTableau$HerbeMouillee)})
```

12.11 Learning the structure

```r
library(pcalg)
alpha <- 0.1  # Significance level for independence tests
pc_result <- pc(
  suffStat = list(dm = as.matrix(X), nlev = rep(2, ncol(X)), adaptDF = FALSE),
  indepTest = binCItest,  # Test for binary data
  alpha = alpha,          # Significance level
  labels = colnames(X),   # Variable names
  verbose = TRUE          # Print progress
)

# Display the inferred DAG
cat("\nInferred skeleton (undirected structure):\n")
print(pc_result@graph)

cat("\nInferred DAG (with orientations):\n")
print(pc_result@graph)

# Plot the resulting DAG
library(Rgraphviz)
plot(pc_result, main = "Inferred DAG using PC Algorithm and binCItest")
```

Back to lecture

12.12 Exercices directed Graphical Model

12.12.1 Joint distribution and graphical decomposition (Bishop 8.3)

The joint distribution over three binary variables

a	b	c	$p(a,b,c)$
0	0	0	0.192
0	0	1	0.144
0	1	0	0.048
0	1	1	0.216
1	0	0	0.192
1	0	1	0.064
1	1	0	0.048
1	1	1	0.096

12.13 Exercices directed Graphical Model

12.13.1 Bishop 8.3

Consider three binary variables $a,\ b,\ c\ \in \{0,1\}$ having the joint distribution given in Table above. Show by direct evaluation that this distribution has the property that a and b are marginally dependent, so that $p(a,b) \neq= p(a)p(b)$, but that they become independent when conditioned on c, so that $p(a,b \mid c) = p(a \mid c)p(b \mid c)$ for both $c = 0$ and $c = 1$.

12.14 Exercices directed Graphical Model

12.14.1 Bishop 8.4

Show by direct evaluation that $p(a,b,c) = p(a)p(c \mid a)p(b \mid c)$. Draw the corresponding directed graph.

12.15 Local Markov Property

12.15.1 directed local Markov property

$t \perp nd(t) \backslash pa(t) | pa(t)$ where the non-descendants of a node $nd(t)$ are all the nodes except for its descendants

We the topological ordering we have

$$p(x_t | x_1, \cdots, x_{t-1}) = p(x_t | x_{nd(t)}) = p(x_t | x_{pa(t)})$$

Thus

$$\begin{aligned} p(x_t, x_{nd(t)\backslash pa(t)} | x_{pa(t)}) &= p(x_{nd(t)\backslash pa(t)} | x_{pa(t)}) p(x_t | x_{pa(t)}, x_{nd(t)\backslash pa(t)}) \\ &= p(x_{nd(t)} | x_{pa(t)}) p(x_t | x_{pa(t)}) \end{aligned}$$

12.16 Gaussian mutual information

$$\begin{aligned}
I(s,t) =& \mathbb{E}[\log \frac{p(x_s,x_t)}{p(x_s)p(x_t)}] && (1)\\
=& -\frac{1}{2}\log \frac{|\Sigma|}{|diag(\sigma_1^2,\sigma_2^2)|} - \frac{1}{2}\mathbb{E}[z^t\Sigma^{-1}z - z^t \begin{bmatrix} 1/\sigma_1^2 & 0 \\ 0 & 1/\sigma_2^2 \end{bmatrix} z] && (2)\\
=& -\frac{1}{2}\log(1-\rho^2) - 1/2trace(E[zz^t(\Sigma^{-1} - \begin{bmatrix} 1/\sigma_1^2 & 0 \\ 0 & 1/\sigma_2^2 \end{bmatrix})]) && (3)\\
=& -\frac{1}{2}\log(1-\rho^2) - trace(I - \begin{bmatrix} 1 & \sigma_{12}/\sigma_2^2 \\ \sigma_{12}/\sigma_1^2 & 1 \end{bmatrix}) && (4)\\
=& -\frac{1}{2}\log(1-\rho^2) && (5)
\end{aligned}$$

where $z = \begin{bmatrix} x_s \\ x_t \end{bmatrix} - \begin{bmatrix} \mu_s \\ \mu_t \end{bmatrix}$ and $E[zz^t] = \Sigma$

12.17 KL-divergence

Maximizing log-likelihood is equivalent to minimizing KL-divergence

13 Projects

13.1 List 2023

Explain a concept and illustrate with an example:

- 10 minutes OBS recording
- Commented Code Notebook (not a full report).

1. Simulation of images using a Strauss model (Markov Random Field). You may use the paper "Markov Random Field Texture Models" Code for simulation Bonus : estimation of the parameters
2. Programmation of Graphical Lasso. You may use the paper "Sparse inverse covariance estimation with the graphical lasso" Original code of the algorithm illustrated with sachs data
3. Program you own Restricted Boltzmann Machine for prediction. You may use the paper "A Practical Guide to Training Restricted Boltzmann Machines" Original code of the algorithm with illustration on MNIST dataset

4. Structural equation models (SEM) using the NoTears approach. You may use the paper “DAGs with NO TEARS: Continuous Optimization for Structure Learning” Use the code from https://github.com/xunzheng/notears and illustrate with one example of your choice

5. Belief Propagation for tree. You may use chapter 20 of Murphy’s book “Machine Learning-A Probabilistic Perspective” Explain exercice Exercice 20.3

6. DAG inference using Deep Learning. Use the paper “DAG-GNN: DAG Structure Learning with Graph Neural Networks” Use the code from https://github.com/fishmoon1234/DAG-GNN to provide an example Sachs proteins (https://perso.univ-rennes1.fr/valerie.monbet/GM/Sachs.html and Tuebingen cause-effect pairs are two common benchmark datasets.