

CSE 6740 Lecture 16

The Beautiful World of Graphical Models

Guest Lecturer: Nishant Mehta

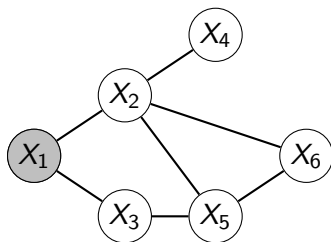
Georgia Institute of Technology

What are graphical models?

For a set of random variables, a graphical model efficiently exhibits the dependence relationships of those variables.

We define the following graph $G(V, E)$ which serves as an alternate representation of our random variables X and their conditional independence relationships:

- For each random variable $X_i \in X$, we have a node in the vertex set V .
- The existence of an edge $(X_i, X_j) \in E$ implies that X_i and X_j are dependent.



Directed and Undirected

Conditional independence relationships are in general less obvious from the graphical structure, but some simple rules can be used to discover these.

Three graphs are of particular interest

- Directed acyclic graphs - Bayesian networks
- Undirected graphs - Markov random fields
- Factor graphs - bipartite graphs of nodes and factors

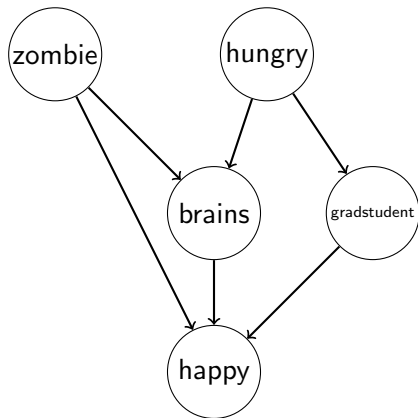
Conditional Independence

A set of random variables A is conditionally independent of a set of random variables B , given another set of random variables C , if

$$\Pr(A \mid B, C) = \Pr(A \mid C)$$

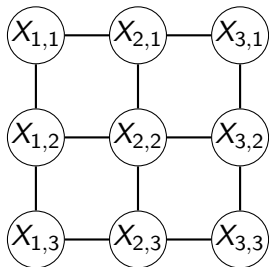
We concisely can express the above relationship as $A \perp\!\!\!\perp B \mid C$.

3 Types of Graphical Models



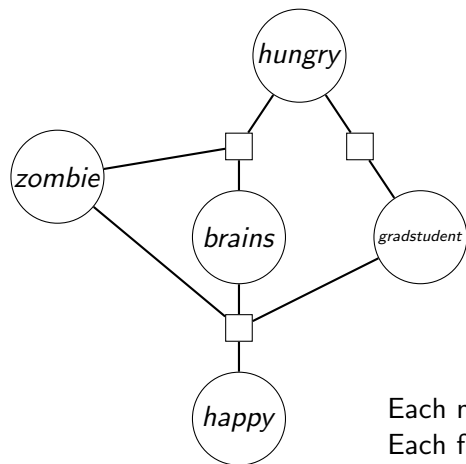
Directed acyclic graphs (DAGs)
Bayesian networks

3 Types of Graphical Models



Undirected graphs
Markov random fields

3 Types of Graphical Models



Factor graphs
bipartite graphs

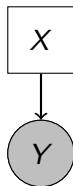
Each node represents a random variable
Each factor represents a potential function operating on the adjacent random variables

Examples of Bayesian networks



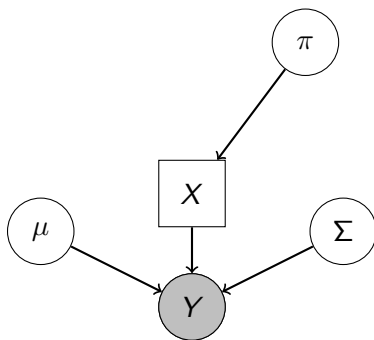
Gaussian

Examples of Bayesian networks



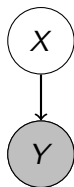
Mixture of Gaussians

Examples of Bayesian networks



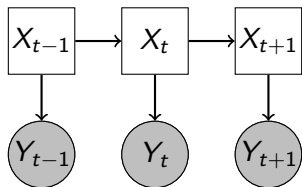
Bayesian Mixture of Gaussians

Examples of Bayesian networks



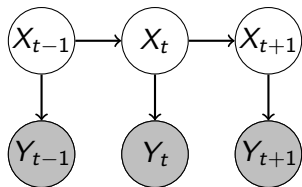
Principal Components Analysis

Examples of Bayesian networks



Hidden Markov model

Examples of Bayesian networks



Linear dynamic system

Temporal extensions

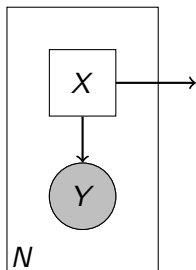


Plate notation

Conditional independence for each type of graph

Inferring conditional independence relationships is different in directed and undirected graphical models.

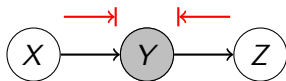
- DAGs - d-separation, Bayes ball algorithm
- Undirected graphs - if all paths from A to B pass through C, then A and B are conditionally independent given C
- Factor graphs - a variable is conditionally independent of all other variables, given its neighbors

Directed conditional independence

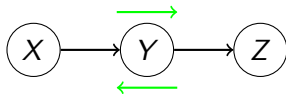
The notion of directed separation, or d-separation, comes down to a few canonical graphical submodels of three nodes:

- Chain
- V-structure
- Single parent structure

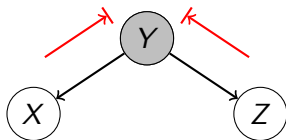
Chain - pass



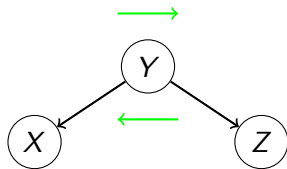
Chain - blocked



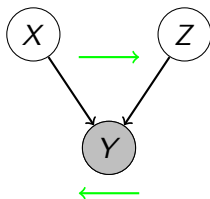
Single parent - pass



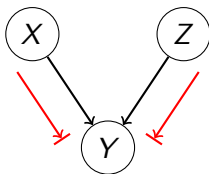
Single parent - blocked



V-structure - blocked



V-structure - pass



Undirected conditional independence

Conditional independence of a random variable set X from Y given B implies that all paths from X to Y must pass through B

This is like the graph theoretic notion of B being a graph cut that separates the graph into two components, one containing X and the other containing Y

- Markov blanket - A set is a Markov blanket of a node X if it contains all nodes that are neighbors of X
- Markov boundary - The minimal Markov blanket for a node X

Suppose B is a Markov blanket of X .

Then we are guaranteed that $X \perp\!\!\!\perp \{G \setminus \{X, B\}\} \mid B$.

Expressive power

We now have a family of all directed graphical models and all undirected graphical models. It is worth investigating if one is strictly more expressive than the other.

By more expressive, we mean that the more expressive model family is able to represent the same conditional independence relationships (including marginal independence) as the less expressive model family.

Can directed models represent the conditional independence relationships of all possible undirected models?

No - Proof: A 4-cycle.

Can undirected models represent all possible directed models?

No - Proof: A directed graph containing a V-structure.

What can we claim?

Any directed tree can be converted into an equivalent undirected graphical model or factor graph

Joint probability factorization

Suppose we have a Bayesian network with N discrete random variables, each taking on r possible values.

The joint probability distribution is

$$\Pr(X_1, X_2, \dots, X_N)$$

Naive storage $\Rightarrow O(r^N)$ space. This is bad!

Joint probability factorization

Naive storage $\Rightarrow O(r^N)$ space

We can exploit the structure of the graph to factorize the joint. Let $pa(X_i)$ refer to a set containing the parent nodes of X_i . The joint then conveniently factors as

$$\Pr(X_1, X_2, \dots, X_N) = \prod_{i=1}^N \Pr(X_i \mid pa(X_i))$$

Note that with some abuse of notation (permitted by the Geneva Convention), we use X_i to refer both to the variable X_i and its node in the graph G .

If every node has at most k parents, so that the maximum fan-in of a node is k , the factorized distribution can be stored in $O(Nr^k)$.

Hammersley-Clifford Theorem

Suppose we have an undirected graph. For a certain, highly useful class of undirected graphs, called Markov random fields, we can factorize the joint probability distribution.

The result comes from the Hammersley-Clifford Theorem (1971). First, we require 2 definitions.

Markov random field

A graphical model where two nodes are conditionally independent whenever they are separated by evidence nodes.

That is, $X_i \perp\!\!\!\perp \{X_G \setminus \{X_i \cup X_{N_i}\}\} \mid X_{N_i}$

Gibbs distribution

A probability distribution that factorizes into positive functions defined on cliques that cover all nodes and edges of graph G .

That is, $\Pr(X) = \frac{1}{Z} \prod_{c \in C_G} \phi_c(X_c)$.

Hammersley-Clifford Theorem

Markov random field

A graphical model where two nodes are conditionally independent whenever they are separated by evidence nodes.

That is, $X_i \perp\!\!\!\perp \{X_G \setminus \{X_i \cup X_{N_i}\}\} \mid X_{N_i}$

Gibbs distribution

A probability distribution that factorizes into positive functions defined on cliques that cover all nodes and edges of graph G .

That is, $\Pr(X) = \frac{1}{Z} \prod_{c \in C_G} \phi_c(X_c)$.

Hammersley-Clifford Theorem

These two definitions are equivalent.

MRF Joint probability factorization

So, let's consider Markov random fields.

Using the Markov property of the Gibbs distribution, the joint probability distribution factorizes as

$$\Pr(X) = \frac{1}{Z} \prod_{c \in C_G} \phi_c(X_c)$$

where the partition function $Z = \sum_X \prod_{c \in C_G} \phi_c(X_c)$.

This buys us massive savings in space complexity.

Storage of the potential functions requires $O(|C_G|r^k)$ space, where k is the size of the maximum clique.

Graphical Model Computations

Let's look at how to compute basic quantities in graphical models. Let (E, Q) be a partitioning of the variables into “evidence” and “query” variables respectively.

Two main quantities we desire are:

- *Marginal probabilities:*

$$\Pr(Q = q) = \sum_e \Pr(Q = q, E = e)$$

$$\Pr(E = e) = \sum_q \Pr(Q = q, E = e)$$

- *Maximum a posteriori (MAP) probabilities:*

$$\Pr^*(Q = q) = \max_e \Pr(Q = q, E = e)$$

Graphical Model Computations

From these basic quantities we can obtain other quantities such as *conditional probabilities*:

$$\Pr(Q = q|E = e) = \frac{\Pr(Q = q, E = e)}{\Pr(E = e)} = \frac{\Pr(Q = q, E = e)}{\sum_q \Pr(Q = q, E = e)}$$

In general multiple marginalizations (summations) must be performed. For example if (E, Q, H) (“evidence”, “query”, and “hidden”) is a partitioning of the variables,

$$\begin{aligned}\Pr(Q = q|E = e) &= \frac{\Pr(Q = q, E = e)}{\sum_q \Pr(Q = q, E = e)} \\ &= \frac{\sum_h \Pr(Q = q, E = e, H = h)}{\sum_q \sum_h \Pr(Q = q, E = e, H = h)}\end{aligned}$$

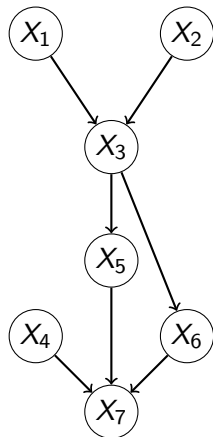
Graphical Model Computations

It would be nice if we could treat both types of graphs in the same way. Note that the directed graph equation is a special case of the undirected graph equation, except that the parent sets are not necessarily cliques. If we modified the original graph a bit, we could make it a special case on the modified graph.

The *moral graph* is an undirected graph obtained by connecting all the parents of each node in the original graph and removing the directions.

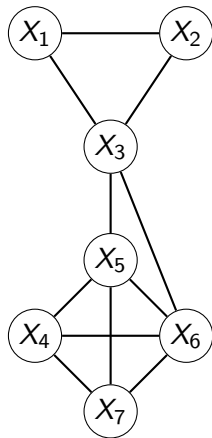
Moralization

Original, free-loving graph



moralization

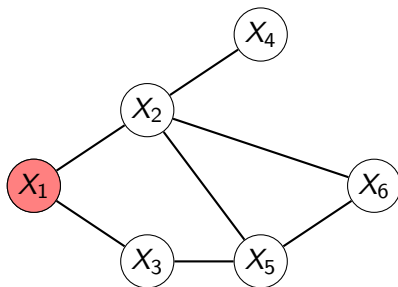
Moral graph



Inference

Suppose we want to do inference on this graph.

Our problem is to compute the marginal distribution $\Pr(X_1)$.



Inference

Suppose that each random variable X_i can take on r values. Then performing the summation below naively is $O(r^5)$:

$$\frac{1}{Z} \sum_{x_2, x_3, x_4, x_5, x_6} \psi_{1,2}(x_1, x_2) \psi_{1,3}(x_1, x_3) \psi_{2,4}(x_1, x_3) \psi_{3,5}(x_3, x_5) \psi_{2,5,6}(x_2, x_5, x_6)$$

We can rearrange the summations by pushing each summation as far right as possible without passing the variable over which it is summing:

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \sum_{x_5} \psi(x_3, x_5) \sum_{x_6} \psi(x_2, x_5, x_6)$$

Elimination algorithm

This gives rise to a simple variable elimination algorithm

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \sum_{x_5} \psi(x_3, x_5) \underbrace{\sum_{x_6} \psi(x_2, x_5, x_6)}$$

Elimination algorithm

This gives rise to a simple variable elimination algorithm

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \sum_{x_5} \psi(x_3, x_5) \underbrace{\sum_{x_6} \psi(x_2, x_5, x_6)}_{m_6(x_2, x_5)}$$
$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \underbrace{\sum_{x_5} \psi(x_3, x_5) m_6(x_2, x_5)}_{m_6(x_2, x_5)}$$

Elimination algorithm

This gives rise to a simple variable elimination algorithm

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \sum_{x_5} \psi(x_3, x_5) \underbrace{\sum_{x_6} \psi(x_2, x_5, x_6)}$$

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \underbrace{\sum_{x_5} \psi(x_3, x_5) m_6(x_2, x_5)}$$

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \underbrace{\sum_{x_4} \psi(x_1, x_3) m_5(x_2, x_3)}$$

Elimination algorithm

This gives rise to a simple variable elimination algorithm

$$\begin{aligned} & \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \sum_{x_5} \psi(x_3, x_5) \underbrace{\sum_{x_6} \psi(x_2, x_5, x_6)} \\ & \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \underbrace{\sum_{x_5} \psi(x_3, x_5)} m_6(x_2, x_5) \\ & \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \underbrace{\sum_{x_4} \psi(x_1, x_3)} m_5(x_2, x_3) \\ & \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \underbrace{\sum_{x_3} \psi(x_1, x_3)} m_4(x_2, x_3) \end{aligned}$$

Elimination algorithm

This gives rise to a simple variable elimination algorithm

$$\begin{aligned} & \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \sum_{x_5} \psi(x_3, x_5) \underbrace{\sum_{x_6} \psi(x_2, x_5, x_6)} \\ & \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \underbrace{\sum_{x_5} \psi(x_3, x_5) m_6(x_2, x_5)} \\ & \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \underbrace{\sum_{x_4} \psi(x_1, x_3) m_5(x_2, x_3)} \\ & \frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \underbrace{\sum_{x_3} \psi(x_1, x_3) m_4(x_2, x_3)} \\ & \frac{1}{Z} \underbrace{\sum_{x_2} \psi(x_1, x_2) m_3(x_1, x_2)} \end{aligned}$$

Elimination algorithm

This gives rise to a simple variable elimination algorithm

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \sum_{x_5} \psi(x_3, x_5) \underbrace{\sum_{x_6} \psi(x_2, x_5, x_6)}$$

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \sum_{x_4} \psi(x_1, x_3) \underbrace{\sum_{x_5} \psi(x_3, x_5) m_6(x_2, x_5)}$$

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \sum_{x_3} \psi(x_1, x_3) \underbrace{\sum_{x_4} \psi(x_1, x_3) m_5(x_2, x_3)}$$

$$\frac{1}{Z} \sum_{x_2} \psi(x_1, x_2) \underbrace{\sum_{x_3} \psi(x_1, x_3) m_4(x_2, x_3)}$$

$$\frac{1}{Z} \underbrace{\sum_{x_2} \psi(x_1, x_2) m_3(x_1, x_2)}$$

$$\frac{1}{Z} m_2(x_1)$$

Elimination algorithm

But what is Z ?

Well, we know that

$$\sum_X \Pr(X) = 1 \Rightarrow \sum_{x_1} \frac{1}{Z} m_2(x_1) = 1 \Rightarrow Z = \sum_{x_1} m_2(x_1)$$

So we happily get Z with one last summation over all values of X_1 .

Note that with directed graphs computing Z is not necessary, because all of our potential functions are actually conditional probabilities, which must sum to 1. This is not true in the undirected case, where we are only guaranteed that the potential functions (clique functions) are nonnegative.

Sum-Product algorithms

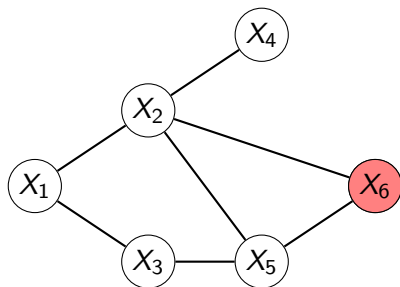
Note the message functions $m_i(x_S)$ that appear as we eliminate variables. To provide a taste of more general inference algorithms on graphs, it will be useful to think of $m_i(X_S)$ as representing a message passed from the eliminated node X_i to the each of the nodes in X_S .

This is known as a Sum-Product algorithm.

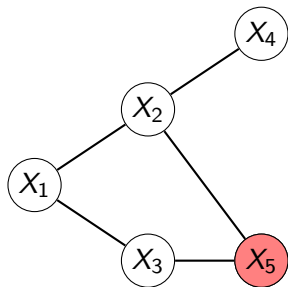
Later, we will explore a more general sum-product algorithm, called Belief Propagation, that extends the Elimination algorithm.

Belief propagation will provide an efficient way for performing exact inference on all of our variables simultaneously, given that the undirected graph (or moralized directed graph) contains at most one loop.

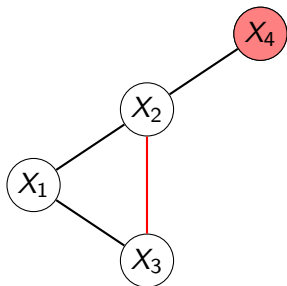
Graph Elimination



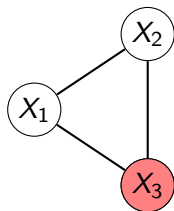
Graph Elimination



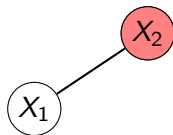
Graph Elimination



Graph Elimination



Graph Elimination



Graph Elimination



Elimination algorithm

Elimination order matters

You can see this by constructing a star graph example

- N vertices
- $N - 1$ vertices have one edge all containing to the N^{th} vertex

Removing the center vertex induces the complete graph K_{N-1} , and the computational problem is hard.

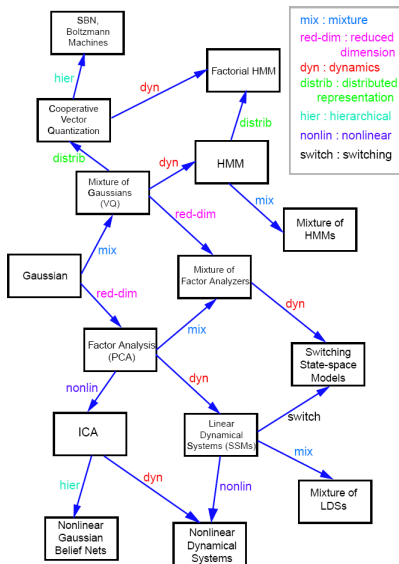
If we instead eliminate all of the vertices of degree 1 first, we are never forced to add any edges during the graph elimination algorithm and computation is easy.

Hardness result

Define the treewidth to be 1 less than the smallest achievable value of the cardinality of the largest elimination clique (by considering all possible elimination orderings).

The problem of finding an elimination ordering that minimizes the treewidth is NP-hard.

Model Model



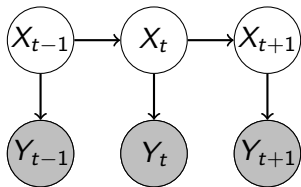
Continuous distributions

Everything we have covered so far also applies to continuous probability distributions, with a couple of important changes:

- We store (typically parametric) conditional density functions rather than conditional probability tables.
- Marginalizing over a random variable, which is a summation in the discrete case, now becomes integration. This often makes computations intractable except for in the case of the Gaussian distribution.

In general, approximations will be made in favor of computability. We will cover some of these approximate inference methods later in the course.

A simple continuous example



Let's consider a linear dynamic system with Gaussian state and observation noise:

$$\begin{aligned}\mathbf{x}_{t+1} &= \mathbf{A}\mathbf{x}_t + \mathbf{v}_t & \mathbf{v}_t &\sim \mathcal{N}(0, \mathbf{Q}) \\ \mathbf{y}_t &= \mathbf{C}\mathbf{x}_t + \mathbf{w}_t & \mathbf{w}_t &\sim \mathcal{N}(0, \mathbf{R})\end{aligned}$$

We can easily compute the conditional probability distribution (CPD) of \mathbf{x}_{t+1} given \mathbf{x}_t :

$$\Pr(\mathbf{x}_{t+1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{A}\mathbf{x}_t, \mathbf{Q})$$

and likewise for the CPD of \mathbf{y}_t given \mathbf{x}_t :

$$\Pr(\mathbf{y}_t \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{C}\mathbf{x}_t, \mathbf{R})$$

Inference for Product of Gaussians

Suppose during inference for some random variable we had to take the product of two potential functions that are Gaussian distributions. Well, the product of Gaussians is still Gaussian, due to the nice exponential form:

$$e^{\text{stuff}} e^{\text{otherstuff}} = e^{\text{stuff} + \text{otherstuff}}$$

This is good news. We will come back to this point later during a lecture on computations for graphical model computations.