AUXILIARY VARIABLE MARKOV CHAIN MONTE CARLO METHODS

MATTHEW MCKENZIE GRAHAM

Doctor of Philosophy
University of Edinburgh
2017

Matthew Mckenzie Graham: *Auxiliary variable Markov chain Monte Carlo methods,* 2017.
Submitted for the degree of Doctor of Philosophy, University of Edinburgh.

SUPERVISOR:
Dr. Amos J. Storkey

ABSTRACT

Inference, the process of drawing conclusions from evidence, is at the heart of the scientific method. Probability theory offers a consistent framework for performing inference in the presence of uncertainty, posing the inference problem as the task of computing expectations of functions of interest with respect to a probability distribution.

In complex models with a large number of unknown variables to be inferred, these expecations can be intractable to compute exactly. This has motivated the development of a large class of approximate inference methods which tradeoff a lack of exactness for greater computational tractability. Monte Carlo methods are one class of such techniques, with the integrals or summations across the whole state space approximated by summations over a finite number of randomly sampled points. This maps the inference problem to that of drawing samples from, often complex, high-dimesional, probability distributions.

LAY SUMMARY

A lay summary is intended to facilitate knowledge exchange, public engagement and outreach. It should be in simple, non-technical terms that are easily understandable by a lay audience, who may be non-professional, non-scientific and outside the research area.

Abstracts, particularly in science, engineering, medicine and veterinary medicine, may be highly technical or contain scientific language that is not easily understandable to readers outside the research area. Therefore, the lay summary is intended as supplementary to the abstract.

ACKNOWLEDGEMENTS

Put acknowledgments here.

Lorem ipsum at nusquam appellantur his, ut eos erant homero concluda turque. Albucius appellantur deterruisset id eam, vivendum partiendo dissentiet ei ius. Vis melius facilisis ea, sea id convenire referrentur, takimata adolescens ex duo. Ei harum argumentum per. Eam vidit exerci appetere ad, ut vel zzril intellegam interpretaris.

Errem omnium ea per, pro Markov chain Monte Carlo (MCMC) con populo ornatus cu, ex qui dicant nemore melius. No pri diam iriure euismod. Graecis eleifend appellantur quo id. Id corpora inimicus nam, facer nonummy ne pro, kasd repudiandae ei mei. Mea menandri mediocrem dissentiet cu, ex nominati imperdiet nec, sea odio duis vocent ei. Tempor everti appareat cu ius, ridens audiam an qui, aliquid admodum conceptam ne qui. Vis ea melius nostrum, mel alienum euripidis eu.

DECLARATION

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

MCMC	Markov chain Monte Carlo
SDE	stochastic differential equation
wrt	with respect to

1 PROBABILISTIC INFERENCE

The actual science of logic is conversant at present only with things either certain, impossible, or entirely doubtful, none of which (fortunately) we have to reason on. Therefore the true logic for this world is the calculus of probabilities
—James Clerk Maxwell

Inference is the process of drawing conclusions from evidence. Much of our lives are spent making inferences about the world given our observations of it; in particular inference is a central aspect of the scientific process. Although deductive logic offers a framework for inferring conclusions from absolute statements of truth, it does not apply to the more typical real-world setting where the information we receive is subject to uncertainty.

To make inferences under conditions of uncertainty, we must instead turn to probability theory. Probabilities offer a consistent framework for quantifying the uncertainty in our beliefs about the world and making inferences given these beliefs. The output of the inference process is itself probabilistic, reflecting that the conclusions we make given uncertain information will themselves be subject to uncertainty.

In this chapter we will first introduce the probability notation we will use in the rest of this work, and state some basic results which will be important in the later chapters. We will introduce graphical models as a compact way of visualising structure in probabilistic models. Finally we will give a concrete definition of the probabilistic inference tasks that the methods presented in the rest of this thesis are aimed at computing (approximate) solutions to, and motivate why such approximate computational methods are needed.

Probability theory is nothing but common sense reduced to calculation.
— Pierre-Simon Laplace

1.1 PROBABILITY THEORY

A *probability space* is defined as a triplet $(S, \mathcal{E}, \mathsf{P})$ where

A σ-algebra, $\mathcal{E}$, on a set S is set of subsets of S with $S \in \mathcal{E}$, $\emptyset \in \mathcal{E}$ and which is closed under complement and countable unions and intersections.

- S is the *sample space*, the set of all possible outcomes,
- $\mathcal{E}$ is the *event space*, a σ-algebra on S, defining all possible events (measurable subsets of S),
- P is the *probability measure*, a finite measure satisfying $\mathsf{P}(S) = 1$, which specifies the probabilities of events in $\mathcal{E}$.

Kolmogorov's axioms:

1. *Non-negativity:* $\mathsf{P}(E) \geq 0 \ \forall E \in \mathcal{E}$,
2. *Normalisation:* $\mathsf{P}(S) = 1$,
3. *Countable additivity: for any countable set of disjoint events* $\{E_i\}_i : E_i \in \mathcal{F} \ \forall i$, $E_i \cap E_j = \emptyset \ \forall i \neq j$, $\mathsf{P}(\cup_i E_i) = \sum_i \mathsf{P}(E_i)$.

Given this definition of a probability space, Kolmogorov's axioms [12] can be used to derive a measure-theoretic formulation of probability theory. The probability of an event $E \in \mathcal{E}$ is defined as the measure of that event $\mathsf{P}(E)$. Two events $A, B \in \mathcal{E}$ are said to be *independent* if $\mathsf{P}(A \cap B) = \mathsf{P}(A)\mathsf{P}(B)$.

A measure-theoretic approach has the advantage of providing a unified treatment for describing probabilities on both finite and infinite sample spaces. Although alternative derivations of the laws of probability from different premises such as Cox's theorem [5, 6] have been proposed, modern extensions of this work result in a calculus of probabilities that is equivalent to Kolmogorov's [17], with the differences mainly being in the philosophical interpretations of probabilities.

1.1.1 Random variables

If $(X, \mathcal{F})$ and $(Y, \mathcal{G})$ are two measurable spaces, a function $f : X \to Y$ is measurable if $f^{-1}(E) \in \mathcal{F} \ \forall E \in \mathcal{G}$.

When modelling real-world processes, rather than considering events as subsets of an abstract sample space, it is usually more helpful to consider *random variables* which represent quantities in the model of interest. A random variable $\mathsf{x} : S \to X$ is defined as a measurable function from the sample space to a measurable space $(X, \mathcal{F})$.

The Borel σ-algebra $\mathscr{B}(\mathbb{R})$ is the smallest σ-algebra on $\mathbb{R}$ which contains all open real intervals.

Often X is the reals, $\mathbb{R}$, and $\mathcal{F}$ is the Borel σ-algebra on the reals, $\mathscr{B}(\mathbb{R})$, in which case we will refer to a *real random variable*. It is also common to consider cases where X is a real vector space, $\mathbb{R}^D$, and $\mathcal{F} = \mathscr{B}(\mathbb{R}^D)$ - in this case we will term the resulting random variable a *random vector* and use the notation $\mathbf{x} : S \to X$. A final special case is when X is countable and $\mathcal{F}$ is the power set $\mathscr{P}(X)$ in which case we will refer to x as a *discrete random variable*.

If $(X, \mathcal{F})$ and $(Y, \mathcal{G})$ are two measurable spaces, μ a measure on these spaces and $f : X \to Y$ a measurable function, the pushforward measure μ_f satisfies $\mu_f(A) = \mu \circ f^{-1}(A) \ \forall A \in \mathcal{G}$.

Due to the definition of a random variable as a measurable function, we can define a pushforward measure on a random variable x

$$\mathsf{P}_{\mathsf{x}}(A) = \mathsf{P} \circ \mathsf{x}^{-1}(A) = \mathsf{P}(\{s \in S : \mathsf{x}(s) \in A\}) \quad \forall A \in \mathcal{F}. \tag{1.1}$$

The measure P_{x} specifies that the probability of the event that the random variable x takes a value in a measurable set $A \in \mathcal{F}$ is $\mathsf{P}_{\mathsf{x}}(A)$.

1.1.2 Joint and conditional probability

Often we will jointly define multiple random variables on the same probability space. Let $(S, \mathcal{E}, \mathsf{P})$ be a probability space and $\mathsf{x} : S \to X$,

$\mathsf{y} : S \to Y$ be two random variables with corresponding σ-algebras $\mathcal{F}$ and $\mathcal{G}$. Then the *joint probability* of x and y is defined as

$$\mathsf{P}_{\mathsf{x},\mathsf{y}}(A,\ B) = \mathsf{P}\Big(\mathsf{x}^{-1}(A) \cap \mathsf{y}^{-1}(B)\Big) \quad \forall A \in \mathcal{F},\ B \in \mathcal{G}. \tag{1.2}$$

The joint probability is related to the probabilities P_x and P_y by

$$\mathsf{P}_{\mathsf{x},\mathsf{y}}(A,\ Y) = \mathsf{P}_\mathsf{x}(A),\ \mathsf{P}_{\mathsf{x},\mathsf{y}}(X,\ B) = \mathsf{P}_\mathsf{y}(B) \quad \forall A \in \mathcal{F},\ B \in \mathcal{G}. \tag{1.3}$$

In this context P_x and P_y are referred to as *marginals* of the joint.

The two random variables are said to be independent if and only if

$$\mathsf{P}_{\mathsf{x},\,\mathsf{y}}(A, B) = \mathsf{P}_\mathsf{x}(A)\mathsf{P}_\mathsf{y}(B) \quad \forall A \in \mathcal{F},\ B \in \mathcal{G}. \tag{1.4}$$

Also useful is the definition of *conditional probability*

$$\mathsf{P}(A \,|\, B) = \frac{\mathsf{P}(A \cap B)}{\mathsf{P}(B)} \quad \forall A \in \mathcal{E},\ B \in \mathcal{E} : \mathsf{P}(B) \neq 0. \tag{1.5}$$

In Kolmogorov's probability theory, (1.5) is given as an additional definition distinct from the basic axioms. In alternatives such as the work of Cox [5, 6] and de Finetti [8], conditional probabilities are instead viewed as a primitive.

Correspondingly, the conditional probabilities of random variables $\mathsf{P}_{\mathsf{x}|\mathsf{y}}$ and $\mathsf{P}_{\mathsf{x}|\mathsf{y}}$ can likewise be defined as satisfying

$$\begin{aligned} \mathsf{P}_{\mathsf{x},\mathsf{y}}(A,\ B) = \mathsf{P}_{\mathsf{x}|\mathsf{y}}(A \,|\, B)\, \mathsf{P}_\mathsf{y}(B) = \mathsf{P}_{\mathsf{y}|\mathsf{x}}(B \,|\, A)\, \mathsf{P}_\mathsf{x}(A) \\ \forall A \in \mathcal{F},\ B \in \mathcal{G} : \mathsf{P}_{\mathsf{x},\mathsf{y}}(A,\ B) \neq 0, \end{aligned} \tag{1.6}$$

which is sometimes referred to as the product rule.

An implication of (1.6) is what is often termed *Bayes' theorem*

$$\mathsf{P}_{\mathsf{x}|\mathsf{y}}(A \,|\, B) = \frac{\mathsf{P}_{\mathsf{y}|\mathsf{x}}(B \,|\, A)\, \mathsf{P}_\mathsf{x}(A)}{\mathsf{P}_\mathsf{y}(B)} \quad \forall A \in \mathcal{F},\ B \in \mathcal{G} : \mathsf{P}_\mathsf{y}(B) \neq 0, \tag{1.7}$$

which will be of key importance in the later discussion of inference.

The definition in (1.2) of the joint probability of a pair of random variables can be extended to arbitarily large collections of random variables. Similarly conditional probabilities can be defined for collections of multiple jointly dependent random variables, with the product rule given in (1.6) generalising to a combinatorial number of possible factorisations of the joint probability. Graphical models offer a convenient way of representing the dependencies between large collections of random variables and any resulting factorisation structure in their joint probability, and will be discussed later in this chapter in section 1.2 .

1.1.3 Probability densities

So far we have ignored how the probability measure P is defined and by consequence the probability of a random variable.

A measure on X is σ-finite if X is a countable union of finite measure sets.

The Radon–Nikodyn theorem guarantees that for a pair of σ–finite measures μ and ν on a measurable space $(X, \mathcal{F})$ where ν is absolutely continuous with respect to μ, then there is a unique (up to μ-null sets) measurable function $f : X \to [0, \infty)$ termed a *density* such that

$$\nu(A) = \int_A f \,\mathrm{d}\mu \quad \forall A \in \mathcal{F}. \tag{1.8}$$

If μ and ν are measures on a measurable space $(X, \mathcal{F})$ then ν has absolute continuity wrt to μ if $\forall A \in \mathcal{F}$, $\mu(A) = 0 \Rightarrow \nu(A) = 0$.

The density function f is also termed the *Radon-Nikodym derivative* of ν with respect to μ, denoted $\frac{\mathrm{d}\nu}{\mathrm{d}\mu}$. Density functions therefore represent a convenient way to define a probability measure with respect to an appropriate base measure. It can also be shown that if $f = \frac{\mathrm{d}\nu}{\mathrm{d}\mu}$ and g is a measurable function that

$$\int_X g \,\mathrm{d}\nu = \int_X g\, f \,\mathrm{d}\mu, \tag{1.9}$$

which we will use later when discussing calculation of expectations.

For real random variables, an appropriate base measure is usually the *Lebesgue measure*, λ, on $\mathbb{R}$. The probability P_x of a real random variable x can then be defined via a *probability density* $\mathrm{p}_\mathrm{x} : \mathbb{R} \to [0, \infty)$ by

$$\mathrm{P}_\mathrm{x}(A) = \int_A \mathrm{p}_\mathrm{x} \,\mathrm{d}\lambda = \int_A \mathrm{p}_\mathrm{x}(x) \,\mathrm{d}x \qquad \forall A \in \mathscr{B}(\mathbb{R}). \tag{1.10}$$

Analagously for a random vector $\mathbf{x}$ with density $\mathrm{p}_\mathbf{x} : \mathbb{R}^D \to [0, \infty)$ with respect to the D-dimensional Lebesgue measure λ^D

$$\mathrm{P}_\mathbf{x}(A) = \int_A \mathrm{p}_\mathbf{x} \,\mathrm{d}\lambda^D = \int_A \mathrm{p}_\mathbf{x}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \qquad \forall A \in \mathscr{B}(\mathbb{R}^D). \tag{1.11}$$

The notation in the second equalities in (1.10) and (1.11) uses a convention that will be used throughout this thesis that integrals without an explicit measure are with respect to the Lebesgue measure.

The counting measure $\#$ is defined as $\#(A) = |A|$ for all finite A and $\#(A) = +\infty$ otherwise.

For discrete random variables, an appropriate base measure is instead the *counting measure*, $\#$. The probability of a discrete random variable is then defined via a probability density $\mathsf{p}_\mathsf{x} : X \to [0, 1]$ by

$$\mathsf{P}_\mathsf{x}(A) = \int_A \mathsf{p}_\mathsf{x} \,\mathrm{d}\# = \sum_{x \in A} \mathsf{p}_\mathsf{x}(x) \qquad \forall A \in \mathscr{P}(X). \tag{1.12}$$

The co-domain of a probability density p_x for a discrete random variable is restricted to $[0, 1]$ due to the non-negativity and normalisation requirements for the probability measure P_x, with $\sum_{x \in X} \mathsf{p}_\mathsf{x}(x) = 1$. Commonly for the case of a discrete random variable, the density p_x is instead referred to as a *probability mass function*, with density reserved for real random variables. We will however use *probability density* in both cases in keeping with the earlier definition of a density with respect to a base measure, this avoiding difficulties when definining joint probabilities on a mixture of real and discrete random variables.

The joint probability $\mathsf{P}_{\mathsf{x},\mathsf{y}}$ of a pair of random variables x and y with co-domains the measurable spaces $(X, \mathcal{F})$ and $(Y, \mathcal{G})$ respectively, can be defined via a joint probability density $\mathsf{p}_{\mathsf{x},\mathsf{y}} : X \times Y \to [0, \infty)$ by

$$\mathsf{P}_{\mathsf{x},\mathsf{y}}(A, B) = \int_{A \times B} \mathsf{p}_{\mathsf{x},\mathsf{y}} \,\mathrm{d}(\mu_\mathsf{x} \times \mu_\mathsf{y}) \quad \forall A \in \mathcal{F},\, B \in \mathcal{G}, \tag{1.13}$$

where $\mu_\mathsf{x} \times \mu_\mathsf{y}$ represents the product measure of two appropriate base measures μ_x and μ_y, e.g. $\mu_\mathsf{x} = \lambda$ and $\mu_\mathsf{y} = \#$ if x is a real random variable and y is a discrete random variable.

If $(X_1, \mathcal{F}_1, \mu_1)$ and $(X_2, \mathcal{F}_2, \mu_2)$ are two measure spaces, the product measure $\mu_1 \times \mu_2$ on a measurable space $(X_1 \times X_2, \mathcal{F}_1 \otimes \mathcal{F}_2)$ is defined as satisfying $(\mu_1 \times \mu_2)(A_1 \times A_2) = \mu_1(A_1)\mu_2(A_2)$ $\forall A_1 \in \mathcal{F}_1,\, A_2 \in \mathcal{F}_2$.

When dealing with random variables, we will often only specify the co-domain of the random variable(s) and a (joint) probability density, with the base measure being implicitly defined as the Lebesgue measure for real random variables (or vectors), counting measure for discrete random variables and an appropriate product measure for a mix of random variables. Similarly we will usually neglect to explicitly define the probability space $(S, \mathcal{E}, \mathsf{P})$ which the random variable(s) map from. In this case we will typically use the loose notation $\mathsf{x} \in X$ to mean a random variable x with co-domain X.

This less explicit but more succinct probability notation in terms of random variables and densities is common in the machine learning and computational statitistics literature and will generally be preferred to improve readability. Table 1.1 gives definitions of the densities and shorthand notation of some common parametric probability distributions that we will use in this thesis.

Name	Parameters	Shorthand	Density	Support
Normal	$\mu \in \mathbb{R}$: mean $\sigma > 0$: standard deviation	$\mathcal{N}\left(x \mid \mu, \sigma^2\right)$	$\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$	$x \in \mathbb{R}$
Log-normal	$\mu \in \mathbb{R}$: log mean $\sigma > 0$: log standard deviation	$\mathrm{LogNorm}(x \mid \mu, \sigma^2)$	$\frac{1}{x\sqrt{2\pi}\sigma} \exp\left(-\frac{(\log x-\mu)^2}{2\sigma^2}\right)$	$x > 0$
Multivariate normal	$\boldsymbol{\mu} \in \mathbb{R}^D$: mean vector $\Sigma \in S_{++}^D$: covariance matrix	$\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \Sigma)$	$\frac{1}{\sqrt{(2\pi)^D \lvert\Sigma\rvert}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\intercal \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$	$\boldsymbol{x} \in \mathbb{R}^D$
Exponential	$\lambda > 0$: rate	$\mathrm{Exp}(x \mid \lambda)$	$\lambda \exp(-\lambda x)$	$x \geq 0$
Uniform	$a \in \mathbb{R}$: minimum $b \in \mathbb{R}$: maximum, $b > a$	$\mathcal{U}(x \mid a, b)$	$\frac{1}{b-a} \mathbb{1}_{[a,b]}(x)$	$a \leq x \leq b$
Half-Cauchy	$\gamma > 0$: scale	$\mathcal{C}_{\geq 0}(x \mid \gamma)$	$\frac{2}{\pi\gamma}\left(1 + \frac{x^2}{\gamma^2}\right)^{-1}$	$x \geq 0$
Gamma	$\alpha > 0$: shape $\beta > 0$: rate	$\mathrm{Gamma}(x \mid \alpha, \beta)$	$\frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} \exp(-\beta x)$	$x \geq 0$
Logistic	$m \in \mathbb{R}$: location $s > 0$: scale	$\mathrm{Logistic}(x \mid m, s)$	$\frac{1}{4s} \cosh\left(\frac{x-m}{2s}\right)^{-2}$	$x \in \mathbb{R}$
Inverse cosh	$m \in \mathbb{R}$: location $s > 0$: scale	$\mathrm{InvCosh}(x \mid m, s)$	$\frac{1}{2s} \cosh\left(\frac{\pi(x-m)}{2s}\right)^{-1}$	$x \in \mathbb{R}$

Table 1.1: Definitions of densities of some common parameteric distributions which will be used in this thesis.

1.1.4 Transforms of random variables

It is common to define a random variable via a transform of another. Let x be a random variable with co-domain the measurable space $(X, \mathcal{F})$. Further let $(Y, \mathcal{G})$ be a second measurable space and $\phi : X \to Y$ a measurable function between the two spaces. If we define $\mathsf{y} = \phi \circ \mathsf{x}$ then analagously to our original definition of $\mathsf{P_x}$ as the pushforward measure of P under the measurable function defining x, we can define $\mathsf{P_y}$ in terms of $\mathsf{P_x}$ as

$$\mathsf{P_y}(A) = \mathsf{P_x} \circ \phi^{-1}(A) = \mathsf{P_x}(\{x \in X : \phi(x) \in A\}) \quad \forall A \in \mathcal{G}, \tag{1.14}$$

i.e. the probability of the event $\mathsf{y} \in A$ is equal to the probability of x being in the pre-image under ϕ of A. To calculate probabilities of transformed random variables therefore we will therefore need to be able to find the pre-images of values of the transformed variable.

If the probability $\mathsf{P_x}$ is defined by a probability density $\mathsf{p_x}$ with respect to a measure μ_x, we can also in some cases find a density $\mathsf{p_y}$ on the transformed variable $\mathsf{y} = \phi(\mathsf{x})$ with respect to a (potentially different) measure μ_y which can be used to calculate the probability $\mathsf{P_y}$,

$$\mathsf{P_y}(A) = \int_{\phi^{-1}(A)} \mathsf{p_x} \,\mathrm{d}\mu_\mathsf{x} = \int_A \mathsf{p_y} \,\mathrm{d}\mu_\mathsf{y} \quad \forall A \in \mathcal{G}. \tag{1.15}$$

For random variables with countable co-domains where the integral in (1.15) corresponds to a sum, a $\mathsf{p_y}$ satisfying (1.15) is simple to identify. If x is a discrete random variable with probability density $\mathsf{p_x}$ with respect to the counting measure, then $\mathsf{y} = \phi(\mathsf{x})$ will necessarily also be a discrete random variable. Applying (1.15) for $\mathsf{p_x} = \frac{\mathrm{d}\mathsf{P_x}}{\mathrm{d}\#}$ we have that

$$\begin{aligned}\int_{\phi^{-1}(A)} \mathsf{p_x}(x)\,\mathrm{d}\#(x) &= \sum_{x \in \phi^{-1}(A)} \mathsf{p_x}(x) = \sum_{y \in A} \sum_{x \in \phi^{-1}(y)} \mathsf{p_x}(x) \\ &= \int_A \sum_{x \in \phi^{-1}(y)} \mathsf{p_x}(x)\,\mathrm{d}\#(y) \quad \forall A \in \mathcal{G}. \end{aligned}\tag{1.16}$$

We can therefore define $\mathsf{p_y} = \frac{\mathrm{d}\mathsf{P_y}}{\mathrm{d}\#}$ in terms of $\mathsf{p_x}$ as

$$\mathsf{p_y}(y) = \sum_{x \in \phi^{-1}(y)} \mathsf{p_x}(x) \quad \forall y \in Y. \tag{1.17}$$

In the special case that ϕ is bijective we have that

$$\mathsf{p}_\mathsf{y}(y) = \mathsf{p}_\mathsf{x} \circ \phi^{-1}(y) \quad \forall y \in Y. \tag{1.18}$$

For transformations of real random variables and vectors, the situation is more complicated as we need to account for any local contraction or expansion of space by the map ϕ. Let $X = \mathbb{R}^M$ and $Y = \mathbb{R}^N$ with $N \leq M$, $N, M \in \mathbb{N}$. We will need a result from geometric measure theory, the *co-area formula* [7]. Let g be an L^1 integrable function and $\boldsymbol{\phi} : X \to Y$ a Lipschitz map. Then the co-area formula states that

$$\int_X g(\boldsymbol{x})\, J_{\boldsymbol{\phi}}(\boldsymbol{x})\, \mathrm{d}\lambda^M(\boldsymbol{x}) = \int_Y \int_{\boldsymbol{\phi}^{-1}(\boldsymbol{y})} g(\boldsymbol{x})\, \mathrm{d}\mathcal{H}^{M-N}(\boldsymbol{x})\, \mathrm{d}\lambda^N(\boldsymbol{y}) \tag{1.19}$$

The D-dimensional Hausdorff measure $\mathcal{H}^D$ on $\mathbb{R}^N$ for $D \in \mathbb{N}, 0 < D < N$ formalises a measure of the 'volume' of D-dimensional submanifolds of $\mathbb{R}^N$ - e.g. for $D = 1$ it corresponds to the length of a curve in $\mathbb{R}^N$. Additionally $\mathcal{H}^N = \lambda^N$ and $\mathcal{H}^0 = \#$.

where $\mathcal{H}^D$ is the D-dimensional *Hausdorff measure* and $J_{\boldsymbol{\phi}} : X \to [0, \infty)$ is the *Jacobian determinant* defined as

$$J_{\boldsymbol{\phi}}(\boldsymbol{x}) = \left| \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}}^{\mathsf{T}} \right|^{\frac{1}{2}} \quad \forall \boldsymbol{x} \in X. \tag{1.20}$$

Now let $\mathbf{x}$ be a random vector with co-domain the measurable space $(X, \mathscr{B}(\mathbb{R}^M))$ and define $\mathbf{y} = \boldsymbol{\phi} \circ \mathbf{x}$ as a random vector with co-domain the measurable space $(Y, \mathscr{B}(\mathbb{R}^N))$ with $\boldsymbol{\phi} : X \to Y$ a Lipschitz map as above. Let $Z = \left\{ \boldsymbol{x} \in X : J_{\boldsymbol{\phi}}(\boldsymbol{x}) = 0 \right\}$ and require that $\mathsf{P}_\mathsf{x}(Z) = 0$. Then for $A \in \mathscr{B}(\mathbb{R}^N)$ define an L^1 integrable function g as

$$g(\boldsymbol{x}) = \begin{cases} \mathbb{1}_A \circ \boldsymbol{\phi}(\boldsymbol{x})\, \mathsf{p}_\mathbf{x}(\boldsymbol{x})\, J_{\boldsymbol{\phi}}(\boldsymbol{x})^{-1} & \forall \boldsymbol{x} \in X \setminus Z \\ 0 & \forall \boldsymbol{x} \in Z \end{cases}. \tag{1.21}$$

Integrating $g(\boldsymbol{x})\, J_{\boldsymbol{\phi}}(\boldsymbol{x})$ over $\boldsymbol{x} \in X$ we have that

$$\int_X g(\boldsymbol{x})\, J_{\boldsymbol{\phi}}(\boldsymbol{x})\, \mathrm{d}\lambda^M(\boldsymbol{x}) = \int_{X \setminus Z} \mathbb{1}_A \circ \boldsymbol{\phi}(\boldsymbol{x})\, \mathsf{p}_\mathbf{x}(\boldsymbol{x})\, \mathrm{d}\lambda^M(\boldsymbol{x}) \tag{1.22}$$

$$= \int_X \mathbb{1}_A \circ \boldsymbol{\phi}(\boldsymbol{x})\, \mathrm{d}\mathsf{P}_\mathsf{x}(\boldsymbol{x}) \tag{1.23}$$

$$= \int_{\phi^{-1}(A)} \mathrm{d}\mathsf{P}_\mathsf{x}(\boldsymbol{x}) = \mathsf{P}_\mathsf{y}(A). \tag{1.24}$$

The equality between first and second lines comes from the requirement $\mathsf{P}_\mathsf{x}(Z) = 0$, with the Lebesgue integrals of a function over two sets which differ by only a zero-measure set equal. Now applying the co-area formula (1.19) to the left-hand side gives

$$\int_Y \int_{\boldsymbol{\phi}^{-1}(\boldsymbol{y})} g(\boldsymbol{x})\,\mathrm{d}\mathcal{H}^{M-N}(\boldsymbol{x})\,\mathrm{d}\lambda^N(\boldsymbol{y}) = \mathrm{P}_{\mathrm{y}}(A). \tag{1.25}$$

Therefore we can define a density $\mathrm{p}_{\mathrm{y}} = \frac{\mathrm{d}\mathrm{P}_{\mathrm{y}}}{\mathrm{d}\lambda^N}$ satisfying (1.15) as

$$\mathrm{p}_{\mathrm{y}}(\boldsymbol{y}) = \int_{\boldsymbol{\phi}^{-1}(\boldsymbol{y})} \mathrm{p}_{\mathbf{x}}(\boldsymbol{x})\,J_{\boldsymbol{\phi}}(\boldsymbol{x})^{-1}\,\mathrm{d}\mathcal{H}^{M-N}(\boldsymbol{x}) \quad \forall \boldsymbol{y} \notin \boldsymbol{\phi}(Z). \tag{1.26}$$

For the special case of a dimension-preserving map $\boldsymbol{\phi}$ with $N = M$ the integral in (1.26) is with respect to $\mathcal{H}^0$ which is equivalent to the counting measure #. In this case $J_{\boldsymbol{\phi}}(\boldsymbol{x}) = \left|\frac{\partial\boldsymbol{\phi}}{\partial\boldsymbol{x}}\right|$ and we therefore get

$$\mathrm{p}_{\mathrm{y}}(\boldsymbol{y}) = \sum_{\boldsymbol{x}\in\boldsymbol{\phi}^{-1}(\boldsymbol{y})} \mathrm{p}_{\mathbf{x}}(\boldsymbol{x}) \left|\frac{\partial\boldsymbol{\phi}}{\partial\boldsymbol{x}}\right|^{-1} \quad \forall \boldsymbol{y} \notin \boldsymbol{\phi}(Z). \tag{1.27}$$

Under the further restriction that $\boldsymbol{\phi}$ is bi-Lipschitz, i.e. it is bijective and Lipschitz in both directions, we recover the more commonly presented multidimensional change of variables formula

$$\mathrm{p}_{\mathrm{y}}(\boldsymbol{y}) = \mathrm{p}_{\mathbf{x}} \circ \boldsymbol{\phi}^{-1}(\boldsymbol{y}) \left|\frac{\partial\boldsymbol{\phi}^{-1}}{\partial\boldsymbol{y}}\right| \quad \forall \boldsymbol{y} \in Y. \tag{1.28}$$

In both of the cases considered, we have seen that if the function ϕ the random variable x is mapped through is bijective, the resulting expression for the density on the mapped random variable y is simpler in the sense that the pre-image $\phi^{-1}(y)$ of a point $y \in Y$ is itself a point and so we do not need to integrate or sum over points in the pre-image which will often be difficult to do analytically.

Bijectivity is a very limiting condition however, with many models involving non-bijective transformations of random variables. Later in this thesis we will see that methods used for defining the more general forms for calculating the density of a transformed variable are key to proposed methods for performing inference in generative models defined by complex, non-dimension preserving and non-bijective transformations of random variables.

1.1.5 Expectations

A fundamental operation when working with probabilistic models is computing expectations of random variables. Let $(S, \mathcal{E}, \mathrm{P})$ be a probability space, and $\mathrm{x} : S \to X$ a random variable on this space. Then the *expected value of* x is defined as

$$\mathbb{E}[\mathrm{x}] = \int_S \mathrm{x}(s)\,\mathrm{dP}(s). \tag{1.29}$$

Often it will be more convenient to express expectations in terms of the probability $\mathrm{P_x}$ instead. If $f : S \to X$ is a measurable function and μ a measure on S then the integral with respect to the pushforward measure μ_f of an integrable function g satisfies

$$\int_X g(x)\,\mathrm{d}\mu_f(x) = \int_S g \circ f(s)\,\mathrm{d}\mu(s). \tag{1.30}$$

If we take g as the identity map we therefore have that

$$\mathbb{E}[\mathrm{x}] = \int_X x\,\mathrm{dP_x}(x). \tag{1.31}$$

If $\mathrm{P_x}$ is given by a density $\mathrm{p_x} = \frac{\mathrm{dP_x}}{\mathrm{d}\mu}$ then using (1.9) we also have

$$\mathbb{E}[\mathrm{x}] = \int_X x\,\mathrm{p_x}(x)\,\mathrm{d}\mu(x), \tag{1.32}$$

which is often the form used for computation.

A further useful implication of (1.30) is what is sometimes termed the *Law of the unconscious statistician*. Let $\mathrm{x} : S \to X$ be a random variable, $\phi : X \to Y$ a measurable function and define $\mathrm{y} = \phi \circ \mathrm{x}$. Then the expected value of y is

$$\mathbb{E}[\mathrm{y}] = \int_S \mathrm{y}(s)\,\mathrm{dP}(s) = \int_S \phi \circ \mathrm{x}(s)\,\mathrm{dP}(s) = \int_X \phi(x)\,\mathrm{dP_x}(x), \tag{1.33}$$

i.e. it can be calculated by integrating ϕ with respect to $\mathrm{P_x}$. This means we can calculate expectations of a transformed random variable $\mathrm{y} = \phi(\mathrm{x})$ without needing to use the change of variables formulae from Section 1.1.4 to explicitly calculate the probability $\mathrm{P_y}$ (or density $\mathrm{p_y}$) and with a relatively weak condition of measurability on ϕ.

1.1.6 Conditional expectations and densities

A related concept, and one which will be key in our discussion of inference, is conditional expectation. Let $(S, \mathcal{E}, \mathsf{P})$ be a probability space, $(X, \mathcal{F})$ and $(Y, \mathcal{G})$ two measurable spaces and $\mathsf{x} : S \to X$ and $\mathsf{y} : S \to Y$ two random variables. Then the *conditional expectation of* x *given* y, is defined as a measurable function $\mathbb{E}[\mathsf{x}\,|\,\mathsf{y}] : Y \to X$ satisfying

$$\int_{\mathsf{y}^{-1}(A)} \mathsf{x}(s)\,\mathrm{d}\mathsf{P}(s) = \int_A \mathbb{E}[\mathsf{x}\,|\,\mathsf{y}](y)\,\mathrm{d}\mathsf{P}_{\mathsf{y}}(y) \quad \forall A \in \mathcal{G}. \tag{1.34}$$

$\mathbb{E}[\mathsf{x}\,|\,\mathsf{y}]$ is guaranteed to be uniquely defined almost everywhere in Y by (1.34), i.e. up to P_{y}-null sets. As a particular case where $A = Y$ we recover what is sometimes termed the *Law of total expectation*

$$\int_S \mathsf{x}\,\mathrm{d}\mathsf{P} = \int_S \mathbb{E}[\mathsf{x}\,|\,\mathsf{y}] \circ \mathsf{y}\,\mathrm{d}\mathsf{P} \implies \mathbb{E}[\mathsf{x}] = \mathbb{E}[\mathbb{E}[\mathsf{x}\,|\,\mathsf{y}] \circ \mathsf{y}]. \tag{1.35}$$

We can also motivate a definition of conditional density in terms of conditional expectation. Assume a joint density $\mathsf{p}_{\mathsf{x},\mathsf{y}} = \frac{\mathrm{d}\mathsf{P}_{\mathsf{x},\mathsf{y}}}{\mathrm{d}(\mu_{\mathsf{x}} \times \mu_{\mathsf{y}})}$ exists and has marginal density $\mathsf{p}_{\mathsf{y}} = \frac{\mathrm{d}\mathsf{P}_{\mathsf{y}}}{\mathrm{d}\mu_{\mathsf{y}}}$. Then for all $A \in \mathcal{G}$

$$\int_{\mathsf{y}^{-1}(A)} \mathsf{x}(s)\,\mathrm{d}\mathsf{P}(s) = \int_S \mathsf{x}(s)\,\mathbb{1}_A \circ \mathsf{y}(s)\,\mathrm{d}\mathsf{P}(s) \tag{1.36}$$

$$= \int_{X \times Y} x\,\mathbb{1}_A(y)\,\mathrm{d}\mathsf{P}_{\mathsf{x},\mathsf{y}}(x, y) \tag{1.37}$$

$$= \int_A \int_X x\,\mathsf{p}_{\mathsf{x},\mathsf{y}}(x, y)\,\mathrm{d}\mu_{\mathsf{x}}(x)\,\mathrm{d}\mu_{\mathsf{y}}(y). \tag{1.38}$$

Define $g : Y \to X$ as

$$g(y) = \begin{cases} \int_X x\,\frac{\mathsf{p}_{\mathsf{x},\mathsf{y}}(x,y)}{\mathsf{p}_{\mathsf{y}}(y)}\,\mathrm{d}\mu_{\mathsf{x}}(x) & \forall y \in Y : \mathsf{p}_{\mathsf{y}}(y) > 0 \\ 0 & \forall y \in Y : \mathsf{p}_{\mathsf{y}}(y) = 0. \end{cases} \tag{1.39}$$

Then from (1.38) we have that for all $A \in \mathcal{G}$

$$\int_{\mathsf{y}^{-1}(A)} \mathsf{x}(s)\,\mathrm{d}\mathsf{P}(s) = \int_A g(y)\,\mathsf{p}_{\mathsf{y}}(y)\,\mathrm{d}\mu_{\mathsf{y}}(y) = \int_A g(y)\,\mathrm{d}\mathsf{P}_{\mathsf{y}}(y). \tag{1.40}$$

The definition of g in (1.39) therefore satisfies the definition of conditional expectation in (1.34) and is uniquely defined up to a P_{y}-null set. Therefore if $\mathsf{p}_{\mathsf{x},\mathsf{y}}$ and p_{y} can be defined we have that

$$\mathbb{E}[\mathsf{x} \mid \mathsf{y}](y) = \int_X x \, \mathsf{p}_{\mathsf{x}|\mathsf{y}}(x \mid y) \, \mathrm{d}\mu_{\mathsf{x}}(x) \quad \forall y \in Y : \mathsf{p}_{\mathsf{y}}(y) > 0 \tag{1.41}$$

where the *conditional density of* x *given* y, $\mathsf{p}_{\mathsf{x}|\mathsf{y}}$, is defined as

$$\mathsf{p}_{\mathsf{x}|\mathsf{y}}(x \mid y) = \frac{\mathsf{p}_{\mathsf{x},\mathsf{y}}(x, y)}{\mathsf{p}_{\mathsf{y}}(y)} \quad \forall x \in X,\ y \in Y : \mathsf{p}_{\mathsf{y}}(y) > 0 \tag{1.42}$$

which can be seen to be analagous to the definition of conditional probability in (1.5). Note the definition of conditional expectation in (1.34) was not dependent on a joint density $\mathsf{p}_{\mathsf{x},\mathsf{y}}$ being defined and so is more general than (1.41).

1.2 GRAPHICAL MODELS

Graphical models = statistics × graph theory × computer science
—Zoubin Ghahramani

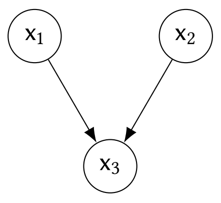
When working with probabilistic models involving large numbers of random variables, it will often be the case that not all the variables are jointly dependent on each other but that instead there are more local conditional relationships between them. Graphical models, which use graphs to describe the dependencies between random variables, are a useful framework for visualising the structure in complex probabilistic models and for giving a graph-theoretic basis for establishing the dependence between sets of random variables.

Central to all graphical models is the concept of conditional independence. Let $(S, \mathcal{E}, \mathsf{P})$ be a probability space and $\mathsf{x} : S \to X$, $\mathsf{y} : S \to Y$ and $\mathsf{z} : S \to Z$ be three random variables with corresponding σ-algebras, $\mathcal{F}_{\mathsf{x}}$, $\mathcal{F}_{\mathsf{y}}$ and $\mathcal{F}_{\mathsf{z}}$ respectively. Following from our earlier definition of (unconditional) independence of random variables in (1.4), we say that x *and* y *are conditionally independent given* z, denoted $\mathsf{x} \perp \mathsf{y} \mid \mathsf{z}$, if

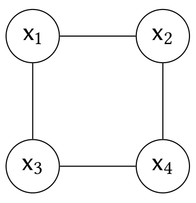
$$\mathbb{E}[\mathbb{1}_A \circ \mathsf{x}\ \mathbb{1}_B \circ \mathsf{y} \mid \mathsf{z}] = \mathbb{E}[\mathbb{1}_A \circ \mathsf{x} \mid \mathsf{z}]\,\mathbb{E}[\mathbb{1}_B \circ \mathsf{y} \mid \mathsf{z}] \ \ \forall A \in \mathcal{F}_{\mathsf{x}},\ B \in \mathcal{F}_{\mathsf{y}}, \tag{1.43}$$

holds almost everywhere with respect to P_{z}. If a joint density on the random variables exists, a sufficient condition for $\mathsf{x} \perp \mathsf{y} \mid \mathsf{z}$ is that the conditional density $\mathsf{p}_{\mathsf{x},\mathsf{y}|\mathsf{z}}$ factorises as

$$\mathsf{p}_{\mathsf{x},\mathsf{y}|\mathsf{z}}(x, y \mid z) = \mathsf{p}_{\mathsf{x}|\mathsf{z}}(x \mid z)\mathsf{p}_{\mathsf{y}|\mathsf{z}}(y \mid z) \quad \forall x \in X,\ y \in Y,\ z \in Z. \tag{1.44}$$



(a) Directed graphical model.



(b) Undirected graphical model.

Figure 1.1: Examples of directed and undirected graphical models. Circular nodes represent random variables in the model, with edges between them indicating dependencies between variables.

This definition can be naturally extended to conditional independence when conditioning on more than one random variable, for example

$$\mathrm{v} \perp \mathrm{x} \mid \mathrm{y}, \mathrm{z} \implies \mathrm{p}_{\mathrm{v},\mathrm{x}|\mathrm{y},\mathrm{z}}(v, x \mid y, z) = \mathrm{p}_{\mathrm{v}|\mathrm{y},\mathrm{z}}(v \mid y, z)\mathrm{p}_{\mathrm{x}|\mathrm{y},\mathrm{z}}(x \mid y, z) \quad (1.45)$$

1.2.1 Directed and undirected graphical models

Several different graphical frameworks have been proposed for representing conditional independency relationships (and other information) in probabilistic models.

Directed graphical models [14], also known as *Bayesian networks*, represent probabilistic models as *directed acyclic graphs* (i.e. a directed graph in which there are no directed cycles), with the nodes in the graph representing random variables in the model and the edges of the graph defining a factorisation of the joint density over these variables into a product of conditional and marginal densities. In particular a conditional density factor is included for each node with parents (on the node random variable value given the parent variable values) and a marginal density factor for each root node without any parents.

An example directed graphical model for three random variables, x_1, x_2 and x_3, is shown in Figure 1.1a. The graph implies that the joint density can be factorised as

$$\mathrm{p}_{\mathrm{x}_1,\mathrm{x}_2,\mathrm{x}_3}(x_1, x_2, x_3) = \mathrm{p}_{\mathrm{x}_3|\mathrm{x}_1,\mathrm{x}_2}(x_3 \mid x_1, x_2)\, \mathrm{p}_{\mathrm{x}_1}(x_1)\, \mathrm{p}_{\mathrm{x}_2}(x_2). \quad (1.46)$$

Note that this factorisation would not be valid for all joint densities on the three variables; in particular we have that x_1 and x_2 are (unconditionally) independent and so that the joint density $\mathrm{p}_{\mathrm{x}_1,\mathrm{x}_2}$ can be written as the product of the two marginals $\mathrm{p}_{\mathrm{x}_1}$ and $\mathrm{p}_{\mathrm{x}_2}$.

Directed graphical models are a natural way of specifying *generative models* - i.e. probabilistic models which can be used to generate simulated observable quantities. Typically the factorisation specified by a directed graphical model gives a straightforward method to generate values from the joint density via *ancestral sampling*.

Ancestral sampling in a directed graphical model corresponds to first sampling values from all the root nodes from their marginal densities, then iteratively sampling from the conditional densities on each node for which all the parents nodes already have sampled values to condition on.

An alternative formalism for graphically representing probabilistic models is that of *undirected graphical models* [11], which are also known as *Markov random fields*. As with directed graphical models, each node in the graph represents a random variable, but here the edges connecting nodes are undirected. Rather than describing a factorisation of a joint density into conditional and marginal densities, an undirected graphical model indicates the factorisation of a joint density into a product of *clique potentials* on each of the *maximal cliques* in the graph.

A *clique* is a fully connected component of the graph - i.e. a subset of nodes in the graph such that all pairs of nodes in the subset are connected by an edge. A *maximal clique* is a clique which is not a strict subset of any other clique. A *clique potential* is a non-negative function of the values of the random variables in the clique; it does necessarily correspond to any conditional or marginal probabilty density.

An example undirected graphical model on four random variables, x_1, x_2, x_3 and x_4, is shown in Figure 1.1b. Here the (maximal) cliques correspond to all the connected pairs of nodes. If $\psi_{a,b}$ denotes the clique potential on the pair (a, b) then the graphical model implies the joint density can be factorised as

$$\begin{gathered}\mathrm{p}_{\mathrm{x}_1,\mathrm{x}_2,\mathrm{x}_3,\mathrm{x}_4}(x_1, x_2, x_3, x_4) = \\ \frac{1}{Z}\psi_{\mathrm{x}_1,\mathrm{x}_2}(x_1, x_2)\psi_{\mathrm{x}_1,\mathrm{x}_3}(x_1, x_3)\psi_{\mathrm{x}_2,\mathrm{x}_4}(x_2, x_4)\psi_{\mathrm{x}_3,\mathrm{x}_4}(x_3, x_4),\end{gathered} \tag{1.47}$$

with Z a normalising constant such that the density integrates to 1 and so defines a valid probability measure.

Undirected graphical models are a natural representation for models of systems of mutually interacting components. For example they are commonly used in models of images to represent dependencies between pixel values and models of ferromagnetism to represent interactions between lattices of particles.

Unlike directed models, generating joint configurations of the random variables in an undirected graphical model from the implied joint distribution is typically a non-trivial task, with no general equivalent to ancestral sampling. Further the joint density can typically only be evaluated up to an unknown normalising constant, with the integral needed to evaluate this constant often intractable for models involving a large number of variables or complex potentials. These properties mean that inference in distributions defined by undirected graphical models is often particularly challenging.

As suggested at the start of this section, both directed and undirected graphical models encode conditional independence properties of probabilistic models. In particular the rules of *D-separation* for directed graphical models and *U-separation* for undirected model give graph-based algorithmic descriptions of how to determine whether a pair of random variables are conditionally independent for a given conditioning set of random variables.

D-separation: $\mathrm{x} \perp \mathrm{y} \,|\, C \iff$ *all paths in the graph between* x *and* y *are blocked. A path is blocked if at least one of the following holds: 1. The path includes a* $\rightarrow\!\bigcirc\!\rightarrow$ *node or a* $\leftarrow\!\bigcirc\!\rightarrow$ *node in C. 2. The path includes a* $\rightarrow\!\bigcirc\!\leftarrow$ *node and neither the node or its descendants are in C.*

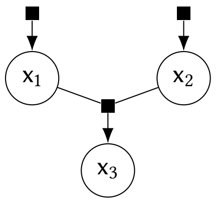
For example the directed graphical model in Figure 1.1a encodes the (un)conditional independence property $\mathrm{x}_1 \perp \mathrm{x}_2 \,|\, \emptyset = \mathrm{x}_1 \perp \mathrm{x}_2$ i.e. that x_1 and x_2 are independent if the value of x_3 is *not* conditioned on. The undirected graphical model in Figure 1.1b encodes the conditional independence properties $\mathrm{x}_1 \perp \mathrm{x}_4 \,|\, \mathrm{x}_2, \mathrm{x}_3$ and $\mathrm{x}_2 \perp \mathrm{x}_3 \,|\, \mathrm{x}_1, \mathrm{x}_4$.

U-separation: x *and* y *in the model and a conditioning set of random variables* C, $\mathrm{x} \perp \mathrm{y} \,|\, C \iff$ *at least one random variable node on every path between* x *and* y *is in C.*

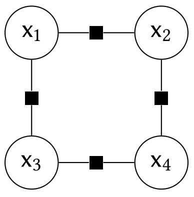
Although there are methods to convert a directed graphical model to an undirected one and vice versa, in general these transformations are lossy - not all of the conditional independence relationships encoded in the original graph will necessarily be maintained in the transformed graph. For example there is no undirected graphical model which will represent the exact set of conditional independence properties represented by the directed graphical model in Figure 1.1a. Likewise there is no directed graphical model which will represent the exact set of conditional independence properties represented by the undirected graphical model in Figure 1.1b. Further there are distributions with dependency structures and factorisations which cannot be uniquely represented by either directed or undirected graphical models [9].

1.2.2 Factor graphs

An alternative graphical model formalism which overcomes some of the limitations of directed and undirected graphical models is that of factor graphs [9, 10]. In factor graphs, in addition to nodes representing random variables, represented as in directed and undirected graphical



(a) A factor graph equivalent of the directed model in Figure 1.1a.



(b) A factor graph equivalent of the undirected model in Figure 1.1b.

Figure 1.2: Examples of factor graphs corresponding to the directed and undirected graphical models in Figure 1.1. Square black nodes correspond to individual factors depending on the connected variables (represented by circular nodes) in the joint density.

models by circular nodes, a second class of nodes, denoted by filled squares (■), are introduced which represent individual factors in the joint density across the random variables represented in the model.

Factors may be either directed or undirected. Undirected factors, denoted by factor nodes in which all edges connecting to variable nodes are undirected, correspond to a factor in the joint density which depends on all of the variables with nodes connected to the factor, but without any requirement that the factor corresponds to a conditional or marginal probability density. Directed factors, denoted by factor nodes in which at least one edge from the factor node to a variable node is directed, correspond to a conditional density on the variables pointed to by directed edges given the values of the variables connected to the the factor node by undirected edges (if there are no such variables then the factor instead corresponds to a marginal density).

Edges between nodes in a factor graph are always between nodes of disparate types i.e. between factor and variable nodes, but never between factor and factor or variable and variable nodes. As with directed graphical models, factor graphs with directed factors must not contain any directed cycles (i.e. a connected loop of edges in which one of every pair of edges connected to a factor on the loop is directed and all of the directed edges point in the same sense around the loop).

In the original extension of undirected factor graphs [10] to include directivity [9], it was proposed to allow multiple directed factors to connect via directed edges to the same variable node, representing multiple factors in a conditional density on that node. This generalisation introduces extra normalisation requirements and looses the interpretation

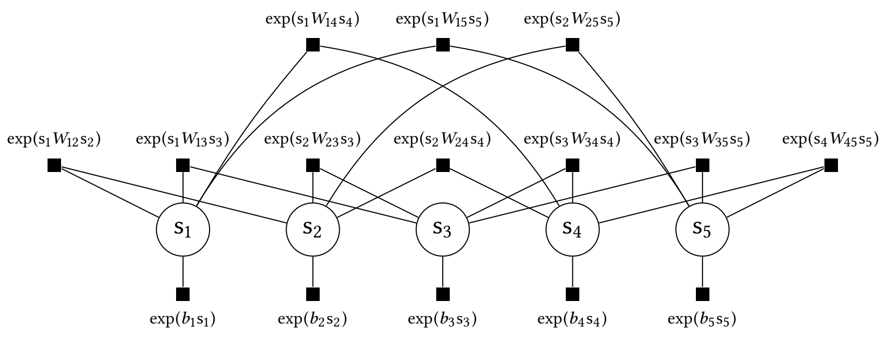


Figure 1.3: Five unit Boltzmann machine factor graph showing explicit factorisation of distribution into pairwise and single variable potentials.

of a directed factor as directly representing a conditional density, and so we will here only use directed factor graphs in which there is at most one directed edge connecting from a factor to a node.

Whether two variables are conditionally independent given a set of other variables can be checked from a factor graph by checking if all paths (i.e. connected series of edges and nodes) between the two corresponding variables nodes in the factor graph are *blocked*. A path is blocked if at least one of the following conditions is satisfied [9]

1. One of the variable nodes in the path is in the conditioning set.
2. One of the directed factor nodes in the path has two connected undirected edges in the path and there is no second directed path from the node to a variable node in the conditioning set.

Both directed and undirected graphical models can always be losslessly converted to a factor graph, i.e. such that by applying the above blocking rules after the transformation we obtain exactly the same set of conditional independency properties as present in the original graph, and thus they have a superset of the capacity to represent conditional indendence properties as either of these two alternative frameworks. For example, factor graph equivalents of the directed and undirected graphical model examples in Figure 1.1 are shown in Figure 1.2.

As well as allowing representations of mixed graphs with both directed and undirected factors which cannot be represented with either directed or undirected graphical models, factor graphs are also able to include finer-grained information about the factorisation of the joint density than either of the other two model types by explicitly indicating the presence of individual factors. For instance Figure 1.3 shows the

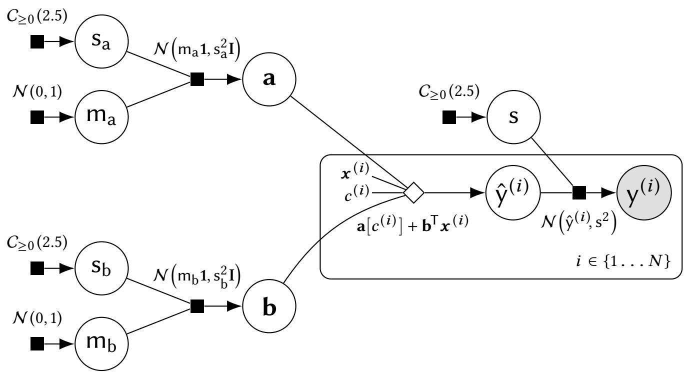


Figure 1.4: Hierarchical linear regression model factor graph showing examples of extended factor graph notation.

factor graph for a *Boltzmann machine* distribution, sometimes called a *pairwise binary Markov random field* or *Ising model*, on five binary random variables $\{s_i\}_{i=1}^5$. A Boltzmann machine distribution can be factored in to a product of pairwise weighted interactions $\exp(s_i W_{ij} s_j)$ and single variable bias potentials $\exp(b_i s_i)$, each of which are explicitly represented by labelled factors in Figure 1.3. A corresponding undirected graphical model representation would have a single clique involving all five variables, and so would not indicate any information about the factorisation of the joint density.

In Figure 1.4 we illustrate some additional useful factor graph notation we will use in this thesis. We use a factor graph corresponding to a hierarchical linear regression model which will be discussed in more detail later in the thesis as a motivating example. The exact meaning of the model and its various factors are unimportant to the discussion of notation here so will be skipped for now.

It will often be useful to be able to explicitly represent deterministic functions applied to the random variables in a factor graph. For this purpose we introduce an additional node type denoted by an unfilled diamond (◇). The semantics of this node type are very similar to standard directed factor nodes. Variables acting as inputs to the function are connected to the node by undirected edges and the variable corresponding to the function output indicated by a directed edge from the node to the relevant variable. Like standard factor nodes, the deterministic factor nodes only ever connect to variable nodes. The operations performed by the function on the inputs will usually be included as a label adjacent to the node as illustrated by the example in Figure 1.4.

The deterministic factor nodes are directly equivalent to a standard directed factor node corresponding to a degenerate Dirac delta conditional density on the output variable which concentrates all the probability mass at the output value of the function applied to inputs (conditioning variables). From this perspective it can be seen that the previously discussed rules for evaluating conditional independency properties in factor graphs can be directly extended to account for the new node type by just considering it as a (specialised) directed factor node.

Optionally constant values used in a model may be included in a factor graph as plain nodes indicated only by a label. The $\boldsymbol{x}^{(i)}$ and $c^{(i)}$ nodes in Figure 1.4 are an example of this notation.

A commonly used convention in both factor graphs and directed graphical models is *plate notation* [4], with an example of a plate shown by the rounded rectangle bounding some of the nodes in Figure 1.4. Plates are used to indicate a subgraph in the model which is replicated multiple times (with the number of replications and an index over the replications typically indicated somewhere on the plate, for example in the lower right corner in the plate in Figure 1.4). The subgraph entirely contained on the plate is assumed to be replicated the relevant number of times, with any edges crossing into the plate from variable nodes outside of the plate being repeated once for each subgraph replication. Plates are commonly used to represent a model component repeated across multiple data items.

Each of the factors in Figure 1.4 is labelled with a shorthand for a probability density function corresponding to the conditional or marginal density factor associated with the node. Definitions for the shorthand notations that are used for densities in this thesis are given in Table 1.1. The dependence of the factors on the value of the random variable the density is defined on is omitted in the factor labels for brevity.

A final additional notation used in Figure 1.4 is the use of a shaded variable node (corresponding to $\mathrm{y}^{(i)}$) to indicate a random variable corresponding to an observable quantity in the model.

1.2.3 Computation graphs

A final graph based tool we will make use of in this theis is that of *computation graphs* [1]. In particular computation graphs (via associated software frameworks [16]) will be used to allow automatic differenti-

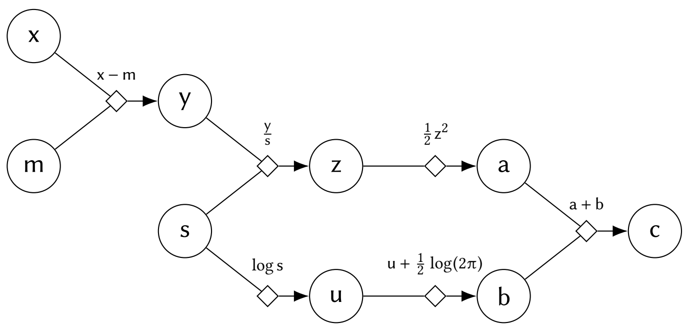


Figure 1.5: Example computation graph corresponding to calculation of the negative log density of a univariate Gaussian distribution.

ation of complex probabilistic models used in later chapters. Computation graphs are not typically considered in the context of probabilistic graphical models, but they share many of the same features and as we will see are closely related to directed factor graphs.

A *computation graph*, sometimes insead termed a *computational graph* or *data flow graph*, represents the computations involved in evaluating a mathematical expression. In this thesis we will distinguish between two types of nodes in a computation graph. *Variable nodes* correspond to variables which hold either inputs to the computation or intermediate results corresponding to the outputs of sub-expressions. *Operation nodes* describe how non-input variable nodes are computed as functions of other variable nodes. In other presentations of computation graphs often the operation nodes are instead implicitly represented by directed edges between variable nodes. However analagously to the more explicit factorisation afforded by directed factor graphs compared to directed graphical models, directly representing operations as nodes allows finer grained information about the decomposition of the operations associated with a computation graph to be included.

As with directed graphical models and directed factor graphs, computation graphs cannot contain directed cycles. This does not preclude recursive and recurrent computations however as these can always be unrolled to form a directed acyclic graph. The 'mathematical expressions' a computation graph is constructed to evaluate can be arbitarily complex - a computation graph corresponding to the evaluation of any numerical algorithm can always be constructed including use of arbitrary nested flow control and branching statements.

An example of a computation graph representing the calculation of the negative log density of a univariate Gaussian distribution, i.e.

$$c = \frac{1}{2}\left(\frac{x - m}{s}\right)^2 + \log s + \frac{1}{2}\log(2\pi) \tag{1.48}$$

is shown in Figure 1.5. The graph inputs have chosen to be the value of the random variable (x) to evaluate the density at and the mean (m) and the standard deviation (s) parameters of the density.

Variable nodes in the computation graph have been represented by labelled circles and operation nodes with labelled diamonds. Undirected edges connecting from a variable node to an operation node correspond to the inputs to the operation, and directed edges from an operation node to variable nodes to the outputs of the operation.

The computation graph associated with a given expression is not uniquely defined. There will usually be multiple possible orderings in which operations can be applied to achieve the same result (up to differences due to non-exact floating point computation). Similarly what should considered a single operation to be represented by a node in the computation graph as opposed to being split up into a sub-graph of multiple operations is a matter of choice. For example in Figure 1.5 the addition of the constant $\frac{1}{2}\log(2\pi)$ could have been included at various other points in the graph and the operation $\frac{1}{2}z^2$ could have been split in to separate multiplication and exponentation operations.

The main motivation for representing expressions as computation graphs is to formalise an efficient general procedure for automatically calculating derivatives of the output of an expression with respect to its inputs termed automatic differentiation [3, 13]. The key ideas in automatic differentiation are to use the chain rule to decompose the derivatives into products and sums of the partial derivatives of the output of each individual operation in the expression with respect to its input, and to use an efficient recursive accumulation of these partial derivative sum-products corresponding to a traversal of the computation graph such that multiple derivatives can be efficiently calculated together.

Depending on how the computation graph is traversed to accumulate the derivative terms, different modes of automatic differentiation are possible. Of most use in this thesis will be *reverse-mode accumulation* [15], in which the derivatives of an output node with respect to all in-

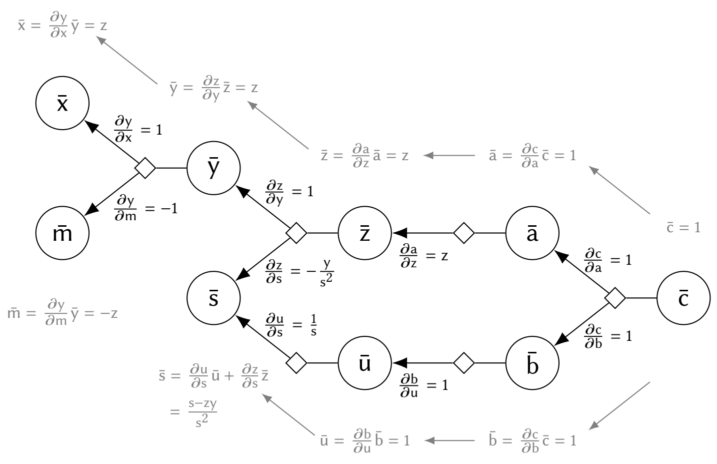

$\bar{x} = \frac{\partial y}{\partial x}\bar{y} = z$
$\bar{y} = \frac{\partial z}{\partial y}\bar{z} = z$
$\bar{z} = \frac{\partial a}{\partial z}\bar{a} = z$
$\bar{a} = \frac{\partial c}{\partial a}\bar{c} = 1$
$\bar{c} = 1$
$\bar{x}$
$\frac{\partial y}{\partial x} = 1$
$\bar{y}$
$\frac{\partial z}{\partial y} = 1$
$\bar{m}$
$\frac{\partial y}{\partial m} = -1$
$\bar{z}$
$\bar{a}$
$\frac{\partial a}{\partial z} = z$
$\frac{\partial c}{\partial a} = 1$
$\frac{\partial z}{\partial s} = -\frac{y}{s^2}$
$\bar{s}$
$\bar{c}$
$\bar{m} = \frac{\partial y}{\partial m}\bar{y} = -z$
$\frac{\partial u}{\partial s} = \frac{1}{s}$
$\frac{\partial c}{\partial b} = 1$
$\bar{u}$
$\bar{b}$
$\frac{\partial b}{\partial u} = 1$
$\bar{s} = \frac{\partial u}{\partial s}\bar{u} + \frac{\partial z}{\partial s}\bar{z}$
$= \frac{s-zy}{s^2}$
$\bar{u} = \frac{\partial b}{\partial u}\bar{b} = 1$
$\bar{b} = \frac{\partial c}{\partial b}\bar{c} = 1$

Figure 1.6: Visualisation of applying reverse-mode automatic differentiation to the computation graph in Figure 1.5 to calculate the derivatives of the negative log density of a univariate Gaussian distribution.

put nodes are accumulated by a reverse pass through the computation graph from the output node to inputs.

As an example the partial derivatives of the expression for univariate Gaussian log density given in (1.48) with respect to x, m and s can be decomposed using the chain rule in terms of the intermediate variables in the computation graph shown in Figure 1.5 as

$$\frac{\partial c}{\partial x} = \frac{\partial c}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial y}\frac{\partial y}{\partial x}, \tag{1.49}$$

$$\frac{\partial c}{\partial m} = \frac{\partial c}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial y}\frac{\partial y}{\partial m}, \tag{1.50}$$

$$\frac{\partial c}{\partial s} = \frac{\partial c}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial s} + \frac{\partial c}{\partial b}\frac{\partial b}{\partial u}\frac{\partial u}{\partial s}. \tag{1.51}$$

We can immediately see that some of the chains of products of partial derivatives are repeated in the different derivative expressions - for example $\frac{\partial c}{\partial a}\frac{\partial a}{\partial z}$ appears in the expressions for all three derivatives. Reverse-mode accumulation is effectively an automatic way of exploiting these possibilities for reusing calculations.

Figure 1.6 shows a visualisation of reverse-mode accumulation applied to the computation graph in Figure 1.5. The first step is for a *forward pass* through the graph to be performed, i.e. values are provided for each of the input variables and then each of the intermediate and output variables calculated from the incoming operation applied to their parent values. Importantly the values of all variables in the graph calculated during the forward pass must be maintained in memory.

The *reverse pass* recursively calculates the values of the partial derivatives of the relevant output node with respect to each variable node in the graph - we will term these intermediate derivatives *accumulators* denoted with barred symbols in Figure 1.6 e.g. $\bar{\mathrm{a}} = \frac{\partial \mathrm{c}}{\partial \mathrm{a}}$. The reverse pass begins by seeding an accumulator for the output node to one (i.e. $\bar{\mathrm{c}} = \frac{\partial \mathrm{c}}{\partial \mathrm{c}} = 1$ in Figure 1.6). Accumulators for the input variables of an operation are calculated by multiplying the accumulator for the operation output by the partial derivatives of the operation output with respect to each input variable. For non-linear operations multiplying by the operator partial derivatives will require access to the value of the input variables calculated in the forward pass. If a variable is an input to multiple operations, the derivative terms from each operation are added together in the relevant accumulator, as for example shown for $\bar{\mathrm{s}}$ in Figure 1.6. By recursively applying these product and sum operations, the derivatives of the output with respect to all variables in the graph can be calculated.

This reverse accumulation method allows computation of numerically exact (up to floating point error) derivatives of a single output variable in a computation graph with respect to *all input variables* with a computational cost, in terms of the number of atomic operations which need to be performed, that is a constant factor of the cost of the evaluation of the original expression represented by the computation graph in the forward pass. The constant factor is typically two to three and at most six [2]. This efficient computational cost is balanced by the requirement that the values of all intermediate variables in the computation graph evaluated in the forward pass through the graph must be stored in memory for the derivative accumulation in a reverse pass, which for large computational graphs can become a bottleneck.

To calculate the full Jacobian from a computation graph representing a function with M inputs $\{\mathrm{x}_i\}_{i=1}^{M}$ and N outputs $\{\mathrm{y}_i\}_{i=1}^{N}$, i.e. the $N \times M$ matrix $\boldsymbol{J}$ with entries $J_{i,j} = \frac{\partial \mathrm{y}_i}{\partial \mathrm{x}_j}$, we can do a single forward pass and N reverse passes each time accumulating the derivatives of one output variable with respect to all inputs. This leads to an overall computational cost that is $\mathcal{O}(N)$ times the cost of a single (forward) function evalaution to evaluate the full Jacobian. As each of the reverse passes can trivially be run in parallel (in addition to any parallelisation of the operations in the forward and reverse passes themselves), this $\mathcal{O}(N)$

factor in the operation count need not corresponds to an equivalent increase in compute time.

An alternative to reverse-mode accumulation is *forward-mode accumulation* [18], which insteads accumulates partial derivatives with respect to a single input variable alongside the forward pass through the graph. In contrast to reverse-mode, this allows calculation of the partial derivatives of all output variables with respect to a single input variable at a computational cost that is a constant factor of the cost of the evaluation of the original expression in the forward pass. Forward-mode accumulation therefore allows evaluation of the Jacobian of a function with M inputs and N outputs at an overall computational cost that is $\mathcal{O}(M)$ times the cost of a single function evaluation.

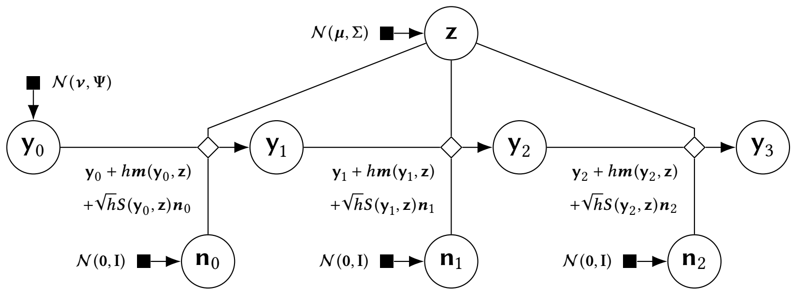
For functions with $M \gg N$, e.g. scalar valued functions of multiple inputs, reverse-mode accumulation is generally therefore signficantly more efficient at computing the Jacobian. Forward-mode accumulation is however useful for evaluating the Jacobian of functions with $N \gg M$, and also has the advantage over reverse-mode accumulation of avoiding the requirement to store the values of intermediate variables from the forward pass for the reverse pass(es).

The direct overlap in our notation to represent variable and operation nodes in computation graphs and that used to represent (random) variable nodes and deterministic factor nodes in factor graphs is intentional. Although often the operations associated with a deterministic node in a factor graph will be more complex than the operations usually represented by nodes in a computation graph, this is only a matter of granularity of reprensentation - fundamentally they perform the same role. Importantly this means we can treat subgraphs of a factor graphs consisting of only variable and deterministic factor nodes as computation graphs and if the operations performed by the deterministic nodes are differentiable, use reverse-mode automatic differentiation to efficiently propagate derivatives through these sub-graphs.

Like directed graphical models, a directed factor graph naturally specifies a generative process via ancestral sampling, with values for the random variables in the graph successively calculated in a forward pass consisting of a combination of deterministic and stochastic operations on the values of parent variables. A computation graph likewise specifies a generative process, how to compute the expression outputs

```
$\mathbf{y}_0 \leftarrow \mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Psi})$
$\mathbf{z} \leftarrow \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
for $t \in \{1 \dots T\}$ do
    $\mathbf{n}_{t-1} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $\mathbf{y}_t \leftarrow \mathbf{y}_{t-1} + h\,\boldsymbol{m}(\mathbf{y}_{t-1}, \mathbf{z}) + \sqrt{h}\,\boldsymbol{S}(\mathbf{y}_{t-1}, \mathbf{z})\mathbf{n}_{t-1}$
```

(a) Pseudo-code for Euler-Maruyama simulation of SDE model.


$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
$\mathbf{z}$
$\mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Psi})$
$\mathbf{y}_0$
$\mathbf{y}_1$
$\mathbf{y}_2$
$\mathbf{y}_3$
$\mathbf{y}_0 + h\boldsymbol{m}(\mathbf{y}_0, \mathbf{z})$
$+\sqrt{h}\boldsymbol{S}(\mathbf{y}_0, \mathbf{z})\boldsymbol{n}_0$
$\mathbf{y}_1 + h\boldsymbol{m}(\mathbf{y}_1, \mathbf{z})$
$+\sqrt{h}\boldsymbol{S}(\mathbf{y}_1, \mathbf{z})\boldsymbol{n}_1$
$\mathbf{y}_2 + h\boldsymbol{m}(\mathbf{y}_2, \mathbf{z})$
$+\sqrt{h}\boldsymbol{S}(\mathbf{y}_2, \mathbf{z})\boldsymbol{n}_2$
$\mathcal{N}(\mathbf{0}, \mathbf{I})$
$\mathbf{n}_0$
$\mathcal{N}(\mathbf{0}, \mathbf{I})$
$\mathbf{n}_1$
$\mathcal{N}(\mathbf{0}, \mathbf{I})$
$\mathbf{n}_2$

(b) Directed factor graph of 3 time steps of SDE simulation.

Figure 1.7: Example of a simulator model corresponding to Euler-Maruyma integration of a set of stochastic differential equations (SDEs), $\mathrm{d}\mathbf{y}(t) = \boldsymbol{m}(\mathbf{y}(t), \mathbf{z})\,\mathrm{d}t + \boldsymbol{S}(\mathbf{y}(t), \mathbf{z})\,\mathrm{d}\mathbf{n}(t)$, specified as pseudo-code in (a) and a directed factor graph in (b).

given inputs, computed via a forward pass through the graph with the main differences being here that the inputs to that process are assumed to be given rather than sampled from marginal densities and the intermediate operations are all deterministic.

Rather than specifying a generative model via a directed factor graph (or graphical model), it is common for complex models to instead be specified procedurally in code as a *simulator*. Often such simulators may involve a mechanistic model of a physical process for example described by a set of stochastic differential equations (SDEs). Any stochasticity in a simulator model will be introduced via draws from a (pseudo-)random number generator in the programming language used to specify the model. Given these random inputs, the output of the simulator is then calculated as a series of determinstic operations performed to the inputs and so can be described by a computation graph. The overall composition of directed factor nodes specifying the generation of random inputs from known densities by the random number generator and computation graph describing the operations performed by the simulator code together therefore define a directed factor graph from which we can extract a joint density on all the variables in the models as the product of all factors. An example of a simulator model corresponding

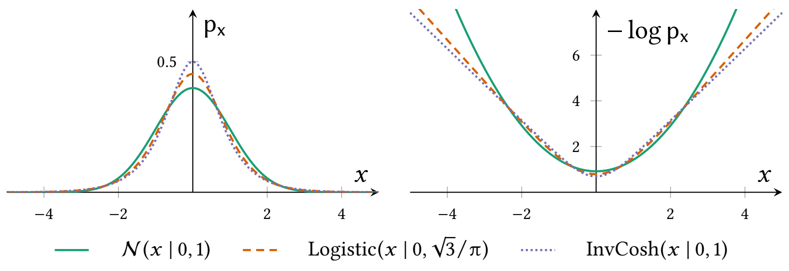


Figure 1.8: Unit variance densities with unbounded support.

to Euler-Maruyma approximate integration of a set of SDEs is shown as both pseudo-code and a corresponding factor graph in Figure 1.7.

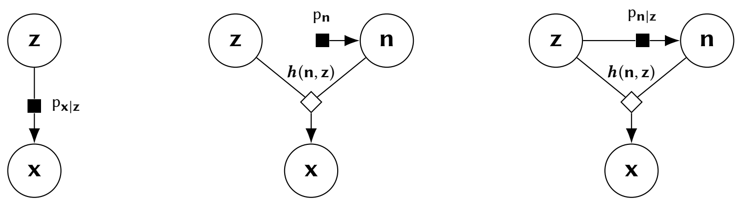
A key difference of simulator models from more typical probabilistic models is that the variables corresponding to observables in the factor graph a simulator model may be the output of deterministic factors (operations) rather than a probabilistic directed factor. As we will see later in the thesis this can complicate inference in such models.

Just as there are multiple ways to formulate a computation graph depending on what are used as the intermediate variables and operations, there is flexibility in how we parameterise probabilistic factors in a factor graph. One immediate example of this comes from the above discussion of simulator models. Typically all random number generator routines in a numerical computing library for densities on real random variables or vectors will be implemented by generating a set of standard uniform random variables using a base pseudo-random number generator and then performing a series of deterministic operations to the uniform variates to produce random variables with the required density. Therefore rather than directly representing a non-uniform directed factor with output $\mathbf{x}$ in a factor graph, we could instead choose to *reparametrise* the factor graph in terms of the the uniform random variables $\mathbf{u}$ which are used to generate $\mathbf{x}$, with $\mathbf{x}$ now the output of a deterministic factor with input $\mathbf{u}$ corresponding to the deterministic transformation used to produce $\mathbf{x}$ with the required density - pictorally $\blacksquare \rightarrow \textcircled{\mathbf{x}}$ is transformed to $\blacksquare \rightarrow \textcircled{\mathbf{u}} - \diamond \rightarrow \textcircled{\mathbf{x}}$.

A common motivation we will have for reparameterising a probabilistic model is to 'standardise' the joint density prior to conditioning on any observed values. In particular it will often be helpful to parameterise models as far as possible in terms independent unit-variance random variables with unbounded support, for instance $\mathcal{N}(0, 1)$, $\mathrm{InvCosh}(0, 1)$

Original factor	Reparametrisation
$\mathcal{N}(\mu, \sigma^2)$: $\blacksquare \longrightarrow \mathrm{x}$	$\mathcal{N}(0,1)$: $\blacksquare \longrightarrow \mathrm{u} \xrightarrow{\mu + \sigma \mathrm{u}} \mathrm{x}$
$\mathrm{LogNorm}(\mu, \sigma^2)$: $\blacksquare \longrightarrow \mathrm{x}$	$\mathcal{N}(0,1)$: $\blacksquare \longrightarrow \mathrm{u} \xrightarrow{\exp(\mu + \sigma \mathrm{u})} \mathrm{x}$
$\mathcal{N}(\boldsymbol{\mu}, \Sigma)$: $\blacksquare \longrightarrow \mathbf{x}$	$\mathcal{N}(\mathbf{0}, \mathbf{I})$: $\blacksquare \longrightarrow \mathbf{u} \xrightarrow{\boldsymbol{\mu} + \mathrm{chol}(\Sigma)\mathbf{u}} \mathbf{x}$
$\mathrm{Exp}(\lambda)$: $\blacksquare \longrightarrow \mathrm{x}$	$\mathrm{Logistic}\left(0, \frac{\sqrt{3}}{\pi}\right)$: $\blacksquare \longrightarrow \mathrm{u} \xrightarrow{\frac{1}{\lambda} \log\left(1 + \exp\left(\frac{\pi \mathrm{u}}{\sqrt{3}}\right)\right)} \mathrm{x}$
$\mathcal{U}(a, b)$: $\blacksquare \longrightarrow \mathrm{x}$	$\mathrm{Logistic}\left(0, \frac{\sqrt{3}}{\pi}\right)$: $\blacksquare \longrightarrow \mathrm{u} \xrightarrow{a + (b - a)\left(1 + \exp\left(\frac{\pi \mathrm{u}}{\sqrt{3}}\right)\right)^{-1}} \mathrm{x}$
$\mathcal{C}_{\geq 0}(\gamma)$: $\blacksquare \longrightarrow \mathrm{x}$	$\mathrm{InvCosh}(0,1)$: $\blacksquare \longrightarrow \mathrm{u} \xrightarrow{\gamma \exp\left(\frac{\pi \mathrm{u}}{2}\right)} \mathrm{x}$

Table 1.2: Standardisation reparametrisations



(a) Original factor (b) Fully reparameterised (c) Partially reparameterised

Figure 1.9: Full and partial auxiliary reparameterisation transforms of a directed factor node representing a conditional density $\mathsf{p}_{\mathbf{x}|\mathbf{z}}$. In full auxiliary reparameterisation an auxiliary random variable $\mathbf{n}$ is introduced which is (unconditionally) independent of $\mathbf{z}$ such that the output of a deterministic transformation $\mathbf{x} = \boldsymbol{h}(\mathbf{n}, \mathbf{z})$ has the same conditional density given $\mathbf{z}$ as the original factor output. In partial auxiliary reparameterisation the auxiliary random variable $\mathbf{n}$ is instead conditionally dependent on $\mathbf{z}$.

or Logistic$(0, \sqrt{3}/\pi)$. The densities for all three shown for comparison in Figure 1.8. Transforming variables to have unbounded support is usually helpful as its avoids the difficulties of working with constrained spaces. Setting all variables to have unit-variance a-priori helps to normalise the scale of variables, which will often for example simplify choosing step size parameters. Parameterising in terms of independent random variables removes any dependencies prior to conditioning on observations, with the resulting joint densities typically having geometries which are easier for inference algorithms to handle.

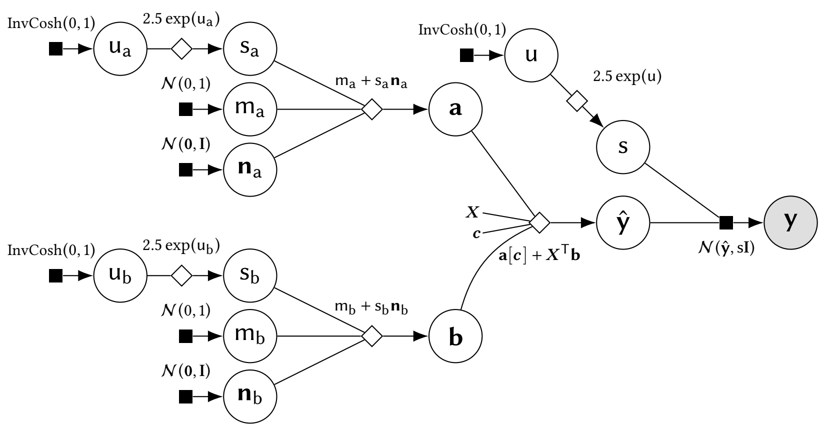


Figure 1.10: Hierarchical linear regression model stochastic computation graph.

1.3 INFERENCE

Having now introduced the underlying theoretical tools we use to construct probabilistic models, we will now describe more concretely the problem of probabilistic inference.

You cannot do inference without making assumptions —David Mackay

The starting point of any inference problem is a model (or models). The model codifies our assumptions about the world. In particular we will assume the model proposes a probabilistic relationship between observable quantities, which we will denote with the random vector $\mathbf{y}$, and unobserved quantities which we wish to infer, which we will denote as the random vector $\mathbf{z}$.

Implicitly or explicitly the model defines the joint probability $\mathsf{P}_{\mathsf{y},\mathsf{z}}$; this might be by defining a joint probability density $\mathsf{p}_{\mathsf{y},\mathsf{z}}$ which specifies $\mathsf{P}_{\mathsf{y},\mathsf{z}}$, or by construction of a *simulator* or *generative model* which allows to generate values for $\mathbf{y}$ and $\mathbf{z}$ without the resulting joint probability $\mathsf{P}_{\mathsf{y},\mathsf{z}}$ being directly specified.

1.3.1 Posterior expectations

1.3.2 Model evidence

2 APPROXIMATE INFERENCE

2.1 DETERMINISTIC APPROACHES

2.1.1 Laplace's method

2.1.2 Variational inference

2.1.3 Expectation propagation

2.2 STOCHASTIC APPROACHES

2.2.1 Monte Carlo method

2.2.2 Rejection sampling

2.2.3 Importance sampling

2.2.4 Markov chain Monte Carlo

3 MARKOV CHAIN MONTE CARLO

3.1 METROPOLIS–HASTINGS

3.2 GIBBS SAMPLING

3.3 SLICE SAMPLING

3.4 HAMILTONIAN MONTE CARLO

4 THERMODYNAMIC METHODS

4.1 SIMULATED TEMPERING

4.2 PARALLEL TEMPERING

4.3 TEMPERED TRANSITIONS

4.4 ANNEALED IMPORTANCE SAMPLING

4.5 PATH SAMPLING

4.6 ADIABATIC MONTE CARLO

BIBLIOGRAPHY

[1] Friedrich L Bauer. ‘Computational graphs and rounding error’. In: *SIAM Journal on Numerical Analysis* 11.1 (1974), pp. 87–96.

[2] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul and Jeffrey Mark Siskind. ‘Automatic differentiation in machine learning: a survey’. In: *arXiv preprint arXiv:1502.05767* (2015).

[3] L. M. Beda, L. N. Korolev, N. V. Sukkikh and T. S. Frolova. *Programs for automatic differentiation for the machine BESM*. Technical Report. (In Russian). Moscow, USSR: Institute for Precise Mechanics and Computation Techniques, Academy of Science, 1959.

[4] Wray L Buntine. ‘Operations for learning with graphical models’. In: *Journal of artificial intelligence research* (1994).

[5] Richard T Cox. ‘Probability, frequency and reasonable expectation’. In: *American Journal of Physics* 14.1 (1946), pp. 1–13. URL: `http://dx.doi.org/10.1119/1.1990764`.

[6] Richard T Cox. ‘The algebra of probable inference’. In: *American Journal of Physics* 31.1 (1963), pp. 66–67. URL: `http://dx.doi.org/10.1119/1.1969248`.

[7] Herbert Federer. *Geometric measure theory*. Springer, 1969.

[8] Bruno de Finetti. ‘Foresight: its logical laws, its subjective sources’. In: *Studies in Subjective Probability*. Ed. by H. E. Kyburg. English translation of original 1937 French article *La Prévision: ses lois logiques, ses sources subjectives*. Springer, 1992, pp. 134–174.

[9] Brendan J Frey. ‘Extending factor graphs so as to unify directed and undirected graphical models’. In: *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc. 2002, pp. 257–264. URL: `https://arxiv.org/abs/1212.2486`.

[10] Brendan J Frey, Frank R Kschischang, Hans-Andrea Loeliger and Niclas Wiberg. ‘Factor graphs and algorithms’. In: *Proceedings of the 35th Annual Allerton Conference on Communication Control and Computing*. 1997.

[11] Ross Kindermann and Laurie Snell. *Markov random fields and their applications*. American Mathematical Society, 1980.

[12] Andreĭ Nikolaevich Kolmogorov. *Foundations of the Theory of Probability*. Ed. by Nathan Morrison. 2nd English Edition. English translation of original 1933 German monograph, *Grundbegriffe der Wahrscheinlichkeitrechnung*. Chelsea Publishing Company, 1956. URL: https://pdfs.semanticscholar.org/c3e1/51f71168a5f348bdebfde11752ca603fa6d0.pdf.

[13] John F Nolan. 'Analytical differentiation on a digital computer'. PhD thesis. Massachusetts Institute of Technology, 1953.

[14] Judea Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann, 1988.

[15] Bert Speelpenning. 'Compiling Fast Partial Derivatives of Functions Given by Algorithms'. PhD thesis. University of Illinois at Urbana-Champaign, 1980.

[16] Theano Development Team et al. 'Theano: A Python framework for fast computation of mathematical expressions'. In: *arXiv e-prints* abs/1605.02688 (May 2016). URL: http://arxiv.org/abs/1605.02688.

[17] Alexander Terenin and David Draper. 'Cox's Theorem and the Jaynesian Interpretation of Probability'. arXiv preprint. 2015. URL: https://arxiv.org/abs/1507.06597v2.

[18] Robert Edwin Wengert. 'A simple automatic derivative evaluation program'. In: *Communications of the ACM* 7.8 (1964), pp. 463–464.