INFERENCE AND LEARNING IN COMPLEX STOCHASTIC PROCESSES

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Xavier Boyen
December 2002

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Daphne Koller
(Principal Adviser)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

John C. Mitchell

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Craig Boutilier, University of Toronto

Approved for the University Committee on Graduate Studies:

To my parents and my brother.

Abstract

Systems with stochastic dynamics and complex structures are pervasive in the real world. Examples abound in a variety of domains, and include the flow of wealth in the stock market, and the ecology of evolving ecosystems. Such systems are composed of many variables that tend to interact in a structured fashion.

The ability to perform inference in such systems is key to many applications, such as tracking or learning based on observed behavior over time. The main obstacle to scalability is the need to reason with probability distributions, which quickly become unmanageable with more than a few variables. In the static case, Bayesian Networks are known to provide an appropriate solution for structured systems. In the dynamic case, inference is almost always intractable, even for nicely structured processes. Intuitively, this is because correlations propagate over time from one variable to another, resulting in all variables becoming fully entangled.

These correlations are shown to be typically weak, and, for inference purposes, to have a limited lifetime. These phenomena are exploited in an efficient approximate inference algorithm for stochastic processes represented as Dynamic Bayesian Networks (DBNs), by factoring out provably weak correlations. Under certain assumptions, it is shown that, even if approximate inference goes on forever, the total approximation error remains small and bounded on expectation. Novel notions of "weak" and "sparse" interaction are introduced, to capture quantitative aspects of the interaction strength between subprocesses. These notions are useful both to refine the estimation of inference errors, and to guide the design of approximations. Experimental performance and accuracy evaluations are provided for tracking complex systems borrowed from real life.

With regard to learning the structure and parameters of partially observable processes, it is shown how approximate inference can greatly accelerate structural and parametric learning, with minimal accuracy penalty. An additional temporal approximation based on similar ideas is also shown to produce high-performance online learners. Finally, it is suggested how to discover hidden state variables simply by identifying unexplained temporal correlations. Experimental results are given in which DBN structures and parameters for several real-life stochastic processes are learned from raw time series data.

Acknowledgements

I am indebted to a number of people, at Stanford and elsewhere, who knowingly or unwittingly had a profound impact on the advent of this dissertation.

First, I would like to thank Daphne Koller, whose unrelenting drive to perfection both for herself and her students has had a profound influence on my *épanouissement* as a researcher. My gratitude also goes to Craig Boutilier and John Mitchell, whose suggestions and critique proved invaluable in the redaction of this thesis, and to Nils Nilsson and Ross Shachter for sharing their insights while serving on my orals.

I feel exceedingly privileged to have had the opportunity to bounce ideas off a number of researchers in the field, both within Daphne's research group and outside of it. I particularly wish to acknowledge Drago Anguelov, Eric Bauer, Urszula Chajewska, Adnan Darwiche, Raya Fratkina, Nir Friedman, Lise Getoor, Carlos Guestrin, Moises Goldszmidt, Joe Halpern, David Heckerman, Eric Horvitz, Michael Jordan, Leslie Kaelbling, Alex Kozlov, Uri Lerner, Brian Milch, Kevin Murphy, Andrew Ng, Uri Nodelman, Tim Paek, Ron Parr, Avi Pfeffer, Andres Rodriguez, Mehran Sahami, Eran Segal, Ben Taskar, Simon Tong, and Leon Wong. Thanks also to Tim Huang for providing the BAT model used in many experiments.

My warmest appreciation goes to a number of faculty and fellow graduate students I had the fortune of knowing during my years at Stanford. I thank Eyal Amir, Guido Appenzeller, Nikolaj Bjørner, Dan Boneh, Edouard Bugnion, Tom Costello, Tom Cover, Patrick Doyle, Bernd Finkbeiner, Piero La Mura, Denis Leroy, Philippe Manne, Pedrito Maynard-Reid II, John McCarthy, Rajeev Motwani, Aarati Parmar, Anna Patterson, Mark Pichora, Vaughan Pratt, Yoav Shoham, Dan Teodosiu, Carlo Tomasi, Tomás Uribe, and Ben Van Roy.

I am very grateful as well to the many friends and colleagues, not named above, who enabled me to grow as a researcher and scholar throughout my graduate career. To all, thanks!

I would also like to acknowledge the generous gift of a Graduate Fellowship from Microsoft Research.

Above all, I would like to express my deep gratitude to my parents, my brother, and Liadan, for their unwavering support during this very special period of my life.

Contents

List of Tables

List of Figures

Part I

Preliminaries

Chapter 1

Introduction

Complex stochastic systems are pervasive in real life. From the changing patterns of air masses that define our climate, to the delicate equilibrium of populations in an ecosystem, to an orchestrated immune response against a bacterial invasion, to the chaotic motion of pedestrians across a large city—in all these examples, complex and highly structured systems seem to obey hidden laws, whereby a patient order emerges from the volatility of the instant. In all these examples, a large stochastic system, composed of myriad hidden random processes interacting with each other in precise ways, evolves with utter unpredictability, yet can be reliably tracked, explained, anticipated, and learned.

In this thesis, we study inference and learning algorithms for complex stochastic processes, *i.e.*, evolving systems composed of a number of subprocesses which interact with each other in structured ways, albeit non-deterministically. In structured processes, variables tend not to interact indiscriminately with all the other variables; rather, interactions are often limited to small subsets of neighboring variables, forming sparse dependency graphs with complex structures. The study of such processes is very important, if only because many processes found in real life are similarly patterned.

1.1 Motivation

Although well-known and effective inference methods exist for many classes of stochastic processes, such as Kalman filters and related techniques for Gaussian processes, reasoning with complex processes remains an unsolved problem, due to the general lack of scalability of those methods. For example, roughly speaking, the mere representation of the state distribution in a discrete multi-variate system takes exponential space in the number of variables. The scalability problem is even more acute when complex processes are richly and compactly modeled as dynamic Bayesian networks, which do not afford the same scalability of inference as provided by regular Bayesian networks in the atemporal case. As it turns out, there are very fundamental reasons why scaling up inference in stochastic dynamic systems is essentially infeasible, in stark contrast with the static case, where adequate process structure goes a very long way to make inference efficient.

The ability to reason effectively about large processes, however, is essential for many purposes. In the natural sciences, for example, it is common investigative practice to conduct extensive simulations of a phenomenon of interest, using handcrafted models that are successively refined based on the results of previous simulations. Since the vast majority of systems studied this way are—at once—dynamic, stochastic, and complex, it is clear that any simulation technique based on rigorous probabilistic inference will soon meet some fundamental tractability limitations.

The tractability problem is even more acute when one wishes to induce explanatory models directly from measurement data. Learning a model from data is, in a sense, very similar to the manual investigative process above; the widely used Expectation-Maximization algorithm for learning under uncertainty is actually based on an alternation between inference and model refinement. One of the consequences of this dependence on inference, is that whatever scalability barriers were faced by probabilistic inference methods will be encountered many times over upon attempting probabilistic learning.

Even more challenging than our metaphorical scientist's successive adjustments to his handcrafted model, is the task of coming up with a working model without relying on much if any prior knowledge about the system. In the world of machine learning, this is what structure discovery is to parametric tuning. Here, the computational complexity is even greater, as the learner must now search a large discrete space of competing structures, in addition to optimizing values in a continuous space of probabilities.

1.2 Summary of contributions

In this dissertation, we show that probabilistic inference in complex dynamic systems does not always need to be intractable, if only we tolerate some approximations in the results. More precisely, we show that, just as the structure of a static model played a dominant role in rendering Bayesian inference feasible, the structure of a dynamic model can be exploited to greatly reduce the apparent complexity of approximate inference in dynamic Bayesian networks, depending on the structure of the model. Based on those insights, we propose a truly scalable approximate inference algorithm for dynamic Bayesian networks. We also provide a thorough analysis of the errors incurred by the approximations, and discuss the accuracy-*vs.*-complexity trade-offs in relation to the structure of the model.

Building on those results, and in view of the central role that inference plays in learning, we then naturally address the problem of learning structured dynamic models under uncertainty. We first present a scalable and efficient algorithm for learning the numerical parameters of a model given its graphical structure, both in an off-line and in an on-line fashion. We provide ample experimental evidence that our learning algorithm hardly suffers at all from using approximate inference as its core, while being significantly faster and much more scalable than competing techniques.

Lastly, we explore the structural learning problem. Using a similar approximation framework as before, we develop a learning algorithm able not only to learn the structure and parameters of the model, but also to populate it with any necessary hidden variables. We illustrate the behavior of this algorithm on a few synthetic and real-life learning tasks.

1.3 Overview of the thesis

The rest of this thesis is organized as follows. The core of the exposition is articulated in two main threads: an Inference part, from Chapter 3 through 6, and a Learning part, from Chapter 7 through 9.

First, in Chapter 2, we briefly recapitulate some important notions of probability and information theory.

Turning to inference, in Chapter 3, we recapitulate a series of important notions relating to stochastic processes, Bayesian networks, and dynamic Bayesian Networks, including a reasonably detailed description of the classic BN inference algorithm.

In Chapter 4, we first study why the classic inference methods, which were so successful for BNs, are bound to fail when applied to DBNs due to an irremediable intractability caused by the temporal dimension. From the insights of this analysis, we propose an alternative solution based on approximate inference. We also provide some empirical validation for our algorithm.

In Chapter 5, we analyze the properties of our algorithm, focusing on the long term effects of repeated approximation. In particular, we show that, due to a stochastic contraction phenomenon, accumulated errors remain nicely bounded rather than diverge to infinity. We also show that the contraction benefits can be maximized by selecting approximations that match the structure of the process, thereby reducing the error.

In Chapter 6, we refine our study of the approximation error, this time focusing on the incremental error incurred at each approximation step. This perspective leads us to introduce quantitative measures of the weakness and sparsity of interactions in complex hierarchical systems, bringing us a refined understanding of how information propagates and correlations arise in such systems.

Shifting our focus to learning, in Chapter 7, we recapitulate a number of the most relevant learning algorithms for partially observable processes, noting that they all rely heavily on multiple probabilistic inference to cope with partial observability.

In Chapter 8, we show how our approximation techniques can be used in parametric learning to break the inference bottleneck. We also propose a theoretically sound algorithm for on-line learning, amd provide experimental validation.

In Chapter 9, we address the problem of structural learning, and propose a novel method for discovering hidden state variables by identifying temporal correlations in the raw data. We present an algorithm that allows us to learn complete structured DBN models from raw time series data. We illustrate the working of the algorithm in a few experiments with synthetic and real-life data.

We devote Chapter 10 to a discussion of related approaches from the literature.

Finally, in Chapter 11, we give a brief conclusion and outline a couple of future research directions.

1.4 Original publications

This dissertation covers the results of research performed during the years 1997–1999. Chapters 2, 3, and 7 summarize some relevant background material known in the art, while Chapters 4, 5, 6, 8, and 9 present our contributions.

The results of Chapters 4 and 5 were originally published in [BK98b]. The contents of Chapter 6 first appeared in [BK99], Chapter 8 in [BK98a], and Chapter 9 in [BFK99].

Chapter 2

Background

Before delving into stochastic processes and Bayesian inference, it is useful to recapitulate a few useful concepts from probability theory and information theory.

2.1 Notions of probability theory

2.1.1 Probability spaces

Mathematically, a *probability space* is a triple $\langle \Omega, \Sigma, \mathbf{P} \rangle$, where Ω is a set of *outcomes*, Σ is a set of *events*, and $\mathbf{P}$ is a unit measure on the measurable space $\langle \Omega, \Sigma \rangle$.

The set Ω may be viewed as the set of all possible states of the world, where each distinguished state ω exhaustively encompasses information relevant to all aspects of the randomness present in the system; in other words, there is no random variable or quantity, observed or not, that is not uniquely determined by the value of Ω. By itself, however, specifying Ω is typically not enough to ascribe meaningful probabilities to its elements, especially in the case of uncountable and continuous spaces, where the probability of any particular outcome will most typically have to be zero. The standard definition of a probability space is therefore measure-theoretic, whereby probabilities are not defined on outcomes but on events, which are measurable sets of outcomes. The set of measurable events, denoted Σ, is a non-empty collection of subsets of Ω with the structure of a *σ-field*, *i.e.*, closed under complementation and under countable union and intersection of its elements. The pair $\langle \Omega, \Sigma \rangle$ is called a *measurable space*. To make a probability space, the measurable space $\langle \Omega, \Sigma \rangle$ is equipped with a probability measure $\mathbf{P}$, *i.e.*, a non-negative countably additive set function $\mathbf{P} : \Sigma \to [0 \cdot\cdot 1]$ that assigns probabilities[1] to events, such that $\mathbf{P}[\Omega] = 1$.

2.1.2 Random variables

The notion of *random variable* is useful when one seeks to restrict attention to only a portion of the information contained in ω, the random state of the world. More precisely, a random variable is defined as a function $X : \Omega \to dom[X]$ from the state space into any set of interest.

As the values taken by a random variable X are mapped to subsets of Ω, or events, they also define a transformation of the probability measure $\mathbf{P} : \Sigma \to [0 \cdot\cdot 1]$ into another probability measure $\mathbf{P}_X : \Sigma_X \to [0 \cdot\cdot 1]$, called the *probability distribution*[2], or, simply, *distribution* of X. The distribution $\mathbf{P}_X$ is defined the natural way: if A is a subset of $dom[X]$ such that the event $(X \in A)$ is measurable, *i.e.*, $A \subseteq dom[X]$, and $\{\omega : X \in A\} \in \Sigma$, then $\mathbf{P}_X[A] = \mathbf{P}[\{\omega : X \in A\}]$.

For simplicity, the notation $\mathbf{P}_X$ is never used: $\mathbf{P}_X[A]$ is conveniently written $\mathbf{P}[X \in A]$; similarly, if $x \in dom[X]$, $\mathbf{P}_X[\{x\}]$ is abbreviated as $\mathbf{P}[X = x]$.

2.1.3 Discrete spaces

The general definitions introduced above may be simplified in the case where the outcome space Ω is finite or countably infinite, and each outcome $\omega \in \Omega$ has a well-defined probability $\mathbf{P}[\{\omega\}]$. In this case, the event set Σ is the set of all subsets of Ω,

[1] Some authors use the term "probability" as an abbreviation for "probability measure". In our convention, a probability is the numerical value assigned by a probability measure to an event of interest.

[2] It is noted that, often in the field of statistics, random variables are specifically required to take values on the real line; correspondingly, statisticians often refer to "distributions" as probability measures over measurable subsets of the reals. We do not impose such restrictions; indeed, in the probability space $\langle \Omega, \Sigma, \mathbf{P} \rangle$, Ω itself may be seen as a random variable (via the identity map) whose distribution is $\mathbf{P}$.

i.e., $\Sigma = 2^\Omega$, and the probability of any event $E \in \Sigma$ is given by:

$$\mathbf{P}[E] \ = \ \sum_{\omega \in E} \mathbf{P}[\{\omega\}] \ ,$$

where $\mathbf{P}[\{\omega\}] \geq 0$ for all $\omega \in E$ and $\mathbf{P}[\Omega] = 1$.

Since we focus almost entirely on the discrete case, this will be the default assumption, unless explicitly specified otherwise. In addition, when talking about a discrete space Ω, we will often omit mention of the event set, which is implicitly taken as $\Sigma = 2^\Omega$.

2.1.4 Joints and marginals

Let $V = \{X_1, ..., X_n\}$ be a set of random variables over some discrete space Ω, on which a probability measure $\mathbf{P}$ is assumed. Let also $1_\wp$ denote the characteristic function of the predicate $\wp$. The *joint distribution* over V is defined as follows:

$$\begin{aligned} &\forall x_1 \in dom[X_1], ..., x_n \in dom[X_n] \\ &\quad \mathbf{P}[X_1 = x_1, ..., X_n = x_n] \ = \ \sum_{\omega} 1_{X_1[\omega]=x_1} \ldots 1_{X_n[\omega]=x_n} \, \mathbf{P}[\omega] \ . \end{aligned}$$

The *marginal distribution* over a subset $V' = \{X_{i_1}, ..., X_{i_k}\} \subseteq V$ is defined in a similar way, substituting the variables in V' for those in V. Although they share a definition, joint and marginal distributions convey different meanings: a distribution is called joint if the emphasis is placed on the plurality of random variables; it is referred to as marginal if the emphasis is placed on the restriction to some variables of interest.

2.1.5 Operations on measures

It is easy to see that the marginal $\mathbf{P}[V']$ from the previous section can be obtained by integrating the joint $\mathbf{P}[V]$ over the domains of all the variables in the difference $V \setminus V'$. More generally, let V and V' be sets of variables over a discrete space Ω, such that $V' \subseteq V$; let also $\bar{\mathbf{P}}[V]$ be any measure over V (not necessarily a distribution).

The *marginalization* of $\bar{\mathbf{P}}[V]$ over V' is written:

$$\bar{\mathbf{P}}[V'] \;=\; \sum_{V\setminus V'} \bar{\mathbf{P}}[V] \;,$$

and satisfies, for every $v' \in dom[V']$:

$$\bar{\mathbf{P}}[V'=v'] \;=\; \sum_{\omega} 1_{V'[\omega]=v'}\, \bar{\mathbf{P}}[\omega] \;.$$

Now, let $\bar{\mathbf{P}}_1[X]$ and $\bar{\mathbf{P}}_2[Y]$ be two measures over the variables or sets of variables X and Y, defined over a common state space Ω. The *product* of $\bar{\mathbf{P}}_1[X]$ and $\bar{\mathbf{P}}_2[Y]$ is the joint measure $\bar{\mathbf{P}}[X,Y]$, written:

$$\bar{\mathbf{P}}[X,Y] \;=\; \bar{\mathbf{P}}_1[X] \otimes \bar{\mathbf{P}}_2[Y] \;,$$

and defined as:

$$\bar{\mathbf{P}}[X=x,Y=y] \;=\; \sum_{\omega} \left(1_{X[\omega]=x}\, \bar{\mathbf{P}}_1[X=x]\right)\left(1_{Y[\omega]=y}\, \bar{\mathbf{P}}_2[Y=y]\right) \;.$$

The product $\bar{\mathbf{P}}_1[X]\otimes\bar{\mathbf{P}}_2[Y]$ is sometimes called the *outer product* of $\bar{\mathbf{P}}_1[X]$ and $\bar{\mathbf{P}}_2[Y]$. The above definition still applies when X and Y are sets of variables with a non-empty intersection.

2.1.6 Conditional distributions

Let $\langle \Omega, 2^{\Omega}, \mathbf{P}\rangle$ be a discrete probability space, and X and Y be two variables or sets of variables in that space. The *conditional distribution* of X given Y, denoted $\mathbf{P}[X|Y]$, is a measure over X and Y, such that $\mathbf{P}[X,Y] = \mathbf{P}[X|Y] \otimes \mathbf{P}[Y]$.

The result of *conditioning* a distribution $\mathbf{P}[X,Y]$ on an event $Y=y$ is the distribution $\mathbf{P}[X|Y=y]$, given by:

$$\mathbf{P}[X=x|Y=y] \;=\; \frac{\mathbf{P}[X=x,Y=y]}{\sum_{y'} \mathbf{P}[X=x,Y=y']} \;.$$

2.1.7 Expectation

Let $f : dom[X] \to \mathbb{R}$ be a real-valued function of a discrete random variable X with probability distribution $\mathbf{P}[X]$. The *expectation* of $f[X]$ is defined as:

$$\mathbb{E}_{\mathbf{P}}[f[X]] = \sum_{x \in dom[X]} \mathbf{P}[X = x]\, f[x] \; .$$

2.2 Notions of information theory

We now recapitulate a few of the central notions of information theory, and refer the reader to the excellent treatise by Cover and Thomas [CT91] for more information.

2.2.1 Entropy

Entropy is the most fundamental information theoretic notion. It expresses the amount of randomness or uncertainty of a distribution or random variable.

Definition 2.1 Let Ω be a discrete space, and φ be a probability distribution defined over it. The *entropy* of the discrete distribution φ is defined as:

$$\begin{aligned} \mathbb{H}[\varphi] &= \mathbb{E}_{\varphi}[log_2 \frac{1}{\varphi}] \\ &= \sum_{\omega \in \Omega} \varphi[\omega]\, log_2 \frac{1}{\varphi[\omega]} \; , \end{aligned}$$

where $(\lambda x.\; x\, log_2 \frac{1}{x})[0] = 0$, to ensure continuity at the limit when $x \to 0$.

The entropy of a discrete distribution is always non-negative, and is zero if and only if all the probability mass is concentrated on a single element of the space. When computed using the logarithm in base 2, the unit of entropy is the *bit*.

The entropy of a discrete random variable X in some probability space $\langle \Omega, 2^{\Omega}, \mathbf{P} \rangle$, is defined as the entropy of the probability distribution induced over the discrete space formed by the domain of the variable, *i.e.*, $\mathbb{H}[X] = \mathbb{H}[\mathbf{P}[X]]$. It is noted that the notation $\mathbb{H}[X]$ implies without mentioning it the probability space in which X is defined.

2.2.2 Conditional entropy

Conditional entropy expresses the amount of randomness expected to remain in a random variable or distribution after the value of a second variable becomes known.

Definition 2.2 Let $\boldsymbol{\varphi}[X,Y]$ be a discrete probability distribution defined over the product domain of two variables X and Y. The *conditional entropy* of X given Y is defined as:

$$\boldsymbol{H}[X|Y] \quad = \quad \boldsymbol{E}_Y[\boldsymbol{E}_{X|Y}[log_2 \frac{1}{\boldsymbol{\varphi}[X|Y]}]] \ .$$

The conditional entropy of a discrete variable is always bounded by its unconditional entropy, *i.e.*, $0 \leq \boldsymbol{H}[X|Y] \leq \boldsymbol{H}[X]$, for any given distribution $\boldsymbol{\varphi}[X,Y]$.

2.2.3 Relative entropy

Introduced by Kullback and Leibler [KL51], the discrete *relative entropy*, also called *KL divergence*, is a measure of the difference between a given distribution $\boldsymbol{\psi}$ and a reference distribution $\boldsymbol{\varphi}$ over the same discrete space. As detailed by Cover and Thomas [CT91, chap. 2], it can be thought of quantifying the information theoretic loss or inefficiency incurred by using the distribution $\boldsymbol{\psi}$ when the true distribution is $\boldsymbol{\varphi}$. It is defined as follows.

Definition 2.3 Letting $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ be two distributions over the same discrete space Ω, the *relative entropy* or *Kullback-Leibler divergence* of $\boldsymbol{\psi}$ with respect to $\boldsymbol{\varphi}$ is given by:

$$\begin{aligned} \boldsymbol{D}_{KL}[\boldsymbol{\varphi} \,||\, \boldsymbol{\psi}] \quad &= \quad \boldsymbol{E}_{\boldsymbol{\varphi}}[log_2 \frac{\boldsymbol{\varphi}}{\boldsymbol{\psi}}] \\ &= \quad \sum_{\omega_i \in \Omega} \boldsymbol{\varphi}[\omega_i] \, log_2 \frac{\boldsymbol{\varphi}[\omega_i]}{\boldsymbol{\psi}[\omega_i]} \ , \end{aligned}$$

where $(0 \, log_2 \frac{0}{x}) = 0$ for all x.

Discrete relative entropy is always non-negative, and is zero when and only when the two distributions are equal. However, it is not a distance metric, as it does not obey the triangle inequality. It is also asymmetric, since, in general, $\mathbb{D}_{KL}[\varphi \,\|\, \psi] \neq \mathbb{D}_{KL}[\psi \,\|\, \varphi]$. When the logarithm is computed in base 2, relative entropy is expressed in *bits*.

2.2.4 Mutual information

Mutual information is a measure of the correlation between two variables, or sets of variables, in some probability space, here assumed discrete. Intuitively, it captures how much information about one variable can be derived from knowledge about the other.

Definition 2.4 Let φ be a probability distribution defined over a discrete space Ω. Let X and Y be two random variables, or sets of random variables, defined in that space. The *mutual information* between X and Y is defined as:

$$\mathbb{I}[X;Y] \quad = \quad \mathbb{D}_{KL}[\varphi[X,Y] \,\|\, \varphi[X] \otimes \varphi[Y]] \; .$$

It is symmetric and non-negative, and is null whenever the pairs of variables are mutually independent.

2.2.5 Conditional mutual information

The related concept of conditional mutual information measures the correlation between two variables, in the expected context provided by a third variable.

Definition 2.5 With φ and Ω as above, let X, Y, and Z be three random variables or sets of random variables. The *conditional mutual information* between X and Y given Z is defined as:

$$\mathbb{I}[X;Y|Z] \quad = \quad \mathbb{E}_{\varphi[Z]}[\mathbb{D}_{KL}[\varphi[X,Y|Z] \,\|\, \varphi[X|Z] \otimes \varphi[Y|Z]]] \; .$$

Again, it is noted that $\mathbb{I}[X;Y|Z]$ depends on φ, which shall be clear from context.

Part II

Inference

Chapter 3

Reasoning under uncertainty

Our focus in this thesis, is reasoning with discrete-time finite-state Markov processes, and especially those featuring extra structure. In the simple case, discrete Markov processes are ideally represented using the well-known *hidden Markov model* (HMM) formalism, which does little more than specifying a flat state space and an explicit stochastic transition law, usually conveyed as a large matrix. Here, we are mainly interested in *structured* processes, *i.e.*, processes whose state space is not flat, but factored in a number of *state variables*, which may have structured ways of interacting with each other. Such systems are more succinctly and usefully modeled using *dynamic Bayesian networks* (DBN), which will most of this thesis is concerned.

3.1 Markov processes

3.1.1 Basic framework

The focus of this thesis is on discrete-time, finite-state, partially observable, stationary, time-invariant, complex stochastic processes.

A *stochastic process* is a dynamic system which evolves non-deterministically. In other words, the state of the system changes over time, in a manner that is at least partially affected by randomness. It follows that the future state of a stochastic process cannot be predicted exactly, even with perfect knowledge of the trajectory of the system up to the present time. We say that a system is *factored* if it is described in terms of a number of state variables, rather than a flat space of enumerated states. We use the words *structured* and *complex* to convey the added connotation that the variables interact with each other in some structured way. The system is *stationary* if the laws that govern its evolution over time are themselves time invariant.

A *partially observable* process is a process in which the state variables are not directly accessible to observation; they are hidden, and can only be observed indirectly via the influence they exert on response variables, often tainted with noise and other distortions. Due to the incurred loss of information, it is typically not possible to know the state of such a system with perfect accuracy at any point in time (except perhaps at some origin at which the process may have started in a determined state).

Under the *discrete time* assumption, time is assumed to range over a discrete set, which we conventionally take as the set of integers. The notation $S^{(t)} = s$ is used to denote that the state of S at time t is s, where $s \in \mathbf{S} = \{s_1, ..., s_n\}$, and $\mathbf{S}$ is the finite set of possible states of the system.

3.1.2 The Markov assumption

Since the state of a stochastic process is rarely known with certainty, reasoning about this state will require consideration of collections of alternate realities. Probability theory provides the appropriate framework, allowing us to treat the uncertain state as a random variable with an associated probability distribution.

At first sight, reasoning with probability distributions may seem a daunting task—and even more so in dynamic systems, since we would presumably not only deal with distributions of states at given points in time, but with distributions of trajectories covering the entire life of the system. In principle, this level of generality is required for reasoning with arbitrary stochastic processes.

The situation becomes significantly easier, however, if the process is *Markovian*. A Markovian process is a process that satisfies the Markov assumption, which states that the future trajectory is independent of the past given the present state of the system. In other words, the Markov assumption asserts that the present state of the system contains enough information to make its future independent from its past, *i.e.*, that $\mathbb{P}[S^{(t+1)}|S^{(0)},...,S^{(t)}] = \mathbb{P}[S^{(t+1)}|S^{(t)}]$.

Unfortunately, in a partially observable system, perfect knowledge of the present state is unattainable. However, as Aström [Ast65] shows, a probability distribution over the present state plays essentially the same role of decoupling past and future, for Markovian systems. Intuitively, to each distinct state in the distribution, corresponds a possible world with that state in which the Markov assumption holds, and to which we have simply assigned a probability. As they reflect the belief of the observer, such state distributions are called *belief states*.

Provided the Markov assumption applies, it follows that many reasoning tasks can be performed exclusively using instantaneous belief states, as opposed to distributions over complete trajectories. As shown in [Ast65], examples of such tasks include the estimation of the current state of the process, the prediction of its future evolution, the explanation of past observations, as well as the synthesis of an optimal control input.

More precisely, it follows from the Markov property that a belief state at some time t captures all information about the past (prior to t), that is relevant to the future evolution of the process (posterior to t), and, in particular, predicting future state distributions and making optimal decisions.

Not only is the Markov assumption very useful, it is also very reasonable. In particular, the laws of classical physics are Markovian, so it should be expected that any system that is accurately described by those laws should have a Markovian model. For this reason, the Markov property is almost universally relied upon when reasoning about stochastic dynamic models—although we hasten to add that it is sometimes deemed useful to depart from this requirement in specific applications.

For mathematical convenience, we also assume our stochastic processes to be *time invariant*. This means that the transition probability from state s_i at time $t-1$ to state s_j at time t is independent of the value of t, *i.e.*, $\forall t, t'$:

$$\mathbb{P}[S^{(t-1)} = s_j | S^{(t-1)} = s_i] \quad = \quad \mathbb{P}[S^{(t'-1)} = s_j | S^{(t'-1)} = s_i] \ .$$

3.1.3 Formalism

Using $s_i^{(t)}$ to denote the event $S^{(t)} = s_i$, it follows from the above assumptions that the process can be described via a *transition model* $\mathcal{S}$, defined as follows: $\forall t$:

$$\mathcal{S}[s_i \stackrel{trn}{\rightarrow} s_j] \quad = \quad \mathbb{P}[s_j^{(t)} | s_i^{(t-1)}] \ .$$

In the case of an HMM, $\mathcal{S}$ is often described explicitly as an $n \times n$ matrix T such that $T_{i,j} = \mathcal{S}[s_i \stackrel{trn}{\rightarrow} s_j]$.

In a partially observable Markov process, the state is not directly observable. Rather, a *response* $R^{(t)}$ is observed, where R is assumed drawn from a finite set $\mathbf{R} = \{r_1, ..., r_m\}$ of possible responses. The response at a given time slice depends stochastically and exclusively on the state at the same time slice; *i.e.*, $R^{(t)}$ is conditionally independent of any $S^{(t')}$ and $R^{(t')}$ given $S^{(t)}$. Using $r_k^{(t)}$ to denote the event $R^{(t)} = r_k$, the response of the process can be described via an *observation model* $\mathcal{R}$, defined as, for any t:

$$\mathcal{R}[s_i \stackrel{obs}{\rightarrow} r_k] \quad = \quad \mathbb{P}[r_k^{(t)} | s_i^{(t)}] \ .$$

The Markov assumption implies that all the historical information needed to monitor or predict the evolution of the process is contained in its present state, or more exactly the partial knowledge of the present state that is theoretically available. This partial knowledge can be summarized in a *belief state*, or probability distribution over the possible states.

Since the knowledge contained in the belief state is to be updated every time a new observation is available, it is necessary to distinguish between the belief states before and after such an update.

Definition 3.1 The *prior belief state* at time t, denoted $\boldsymbol{\sigma}^{(\bullet t)}$, is the distribution over the process state at time t, given the response history from the origin of time up to but not including time t. Letting $r_{k_l}^{(l)}$ denote the response observed at time l, for any

l, the prior belief state is defined as:

$$\boldsymbol{\sigma}^{(\bullet t)}[s_i] \quad = \quad \mathbb{P}[s_i^{(t)} | r_{k_0}^{(0)}, \ldots, r_{k_{t-1}}^{(t-1)}] \ .$$

Definition 3.2 The *posterior belief state* at time t, denoted $\boldsymbol{\sigma}^{(t\bullet)}$, is the distribution over the process state at time t, given the response history from the origin of time up to and including time t. With $r_{k_l}^{(l)}$ as above:

$$\boldsymbol{\sigma}^{(t\bullet)}[s_i] \quad = \quad \mathbb{P}[s_i^{(t)} | r_{k_0}^{(0)}, \ldots, r_{k_{t-1}}^{(t-1)}, r_{k_t}^{(t)}] \ .$$

Unless specified otherwise, an unqualified belief state at some time t is always understood to be the posterior belief state at time t.

3.2 Filtering

Monitoring or *filtering* a stochastic process is the operation of maintaining an up-to-date belief state, as time advances and new process responses are observed. The original optimal filtering algorithm is the celebrated Kalman filter [Kal60], whose invention arguably marks the era of effective reasoning with partially observable stochastic systems.

In principle, the procedure is quite straightforward if the transition and observation models $\mathcal{S}$ and $\mathcal{R}$ are known. The current posterior belief state $\boldsymbol{\sigma}^{(t\bullet)}$ can be derived from the previous posterior belief state $\boldsymbol{\sigma}^{(t-1\bullet)}$ and the current observed response $R^{(t)}$, in a two-state computation, as follows:

1. The prior belief state $\boldsymbol{\sigma}^{(\bullet t)}$ for the current time slice is obtained by propagating $\boldsymbol{\sigma}^{(t-1\bullet)}$ through the stochastic transition model $\mathcal{S}$:

$$\boldsymbol{\sigma}^{(\bullet t)}[s_j] \quad = \quad \sum_{i=1}^{n} \boldsymbol{\sigma}^{(t\bullet)}[s_i] \, \mathcal{S}[s_i \stackrel{trn}{\rightarrow} s_j] \ .$$

2. The posterior belief state $\boldsymbol{\sigma}^{(t\bullet)}$ is then obtained by Bayesian conditioning of $\boldsymbol{\sigma}^{(\bullet t)}$

on the observed response $R^{(t)}$, under the observation model $\mathcal{R}$:

$$\boldsymbol{\sigma}^{(t\bullet)}[s_i] \quad = \quad \frac{\boldsymbol{\sigma}^{(\bullet t)}[s_i]\, \mathcal{R}[s_i \overset{obs}{\rightarrow} R^{(t)}]}{\sum_{i'=1}^{n} \boldsymbol{\sigma}^{(\bullet t)}[i']\, \mathcal{R}[s_{i'} \overset{obs}{\rightarrow} R^{(t)}]} \ .$$

Abstractly, the transition model $\mathcal{S}$ can be viewed as a function mapping the probability distribution $\boldsymbol{\sigma}^{(t-1\bullet)}$ to the probability distribution $\boldsymbol{\sigma}^{(\bullet t)}$, which is also the expected belief state at time t, where the expectation is taken over the distribution of responses at time t given the belief state at time $t-1$. Similarly, to the set $\mathbf{R} = \{r_1, ..., r_m\}$ of possible responses at time t, corresponds a family of functions $\mathcal{R}_{r_k}$, mapping $\boldsymbol{\sigma}^{(\bullet t)}$ to $\boldsymbol{\sigma}^{(t\bullet)}$ in the event that $R^{(t)} = r_k$. Therefore, assuming that the response at time t is $R^{(t)}$, the two-stage update rule for belief states is given by the diagram:

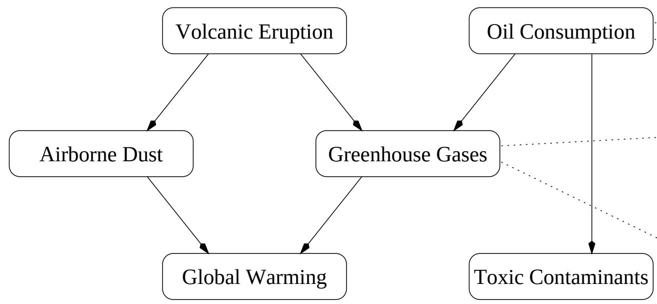
$$\boldsymbol{\sigma}^{(t-1\bullet)} \overset{\mathcal{S}}{\mapsto} \boldsymbol{\sigma}^{(\bullet t)} \overset{\mathcal{R}_{R^{(t)}}}{\mapsto} \boldsymbol{\sigma}^{(t\bullet)} \ ,$$

where the second map is parameterized by $R^{(t)}$.

The actual methods needed to carry the above two steps depend on the chosen representation. In the case of HMMs, as illustrated above, the computation reduces to straightforward vector and matrix operations. Conversely, structured representations require more sophisticated algorithms. A detailed account of probabilistic inference using HMMs and variants thereof can be found in [SHJ96].

3.3 Bayesian networks

Bayesian Networks (BNs), named after Reverend Thomas Bayes (1702–1761), were first devised as a compact and expressive representation of structured multivariate probability distributions over discrete domains [Pea88]. At first an expressive language with limited inference capabilities, this formalism quickly became widely used as the method of choice for reasoning under uncertainty. This success is due in large part to the development of powerful inference algorithms, which could exploit the model structure to great computational advantage. A number of extensions to the formalism have subsequently been proposed, including Gaussian and Hybrid BNs for



P(OC)	false	true
	0.2	0.8

P(GG | VE, OC)	false	true
VE=false, OC=false	0.99	0.01
VE=false, OC=true	0.25	0.75
VE=true, OC=false	0.50	0.50
VE=true, OC=true	0.10	0.90

Figure 3.1: Example of Bayesian network, specified as a directed acyclic graph, where each node is associated with the conditional probability distribution of the random variable it represents (shown for OC and GG only). The model and numbers shown are fictitious.

modeling continuous systems [SK89, Mur98, LP01], as well as, more in line with our subject, Dynamic BNs for modeling systems that evolve over time [DK89].

Example 3.3 Consider the Bayesian network represented in Figure 3.1, which defines a joint probability distribution $\mathcal{P}[\mathsf{VE}, \mathsf{OC}, \mathsf{AD}, \mathsf{GG}, \mathsf{GW}, \mathsf{TC}]$ over the six random variables shown on the figure. The BN is composed of a directed acyclic graph over the six variables (represented as nodes), and an annotation which specifies the conditional probability distribution of each variable given its immediate parents in the graph. The joint probability distribution is then defined as the product of the six conditional probability distributions:

$$\begin{aligned}
&\mathbf{P}[\mathsf{VE}, \mathsf{OC}, \mathsf{AD}, \mathsf{GG}, \mathsf{GW}, \mathsf{TC}] \; = \\
&\quad \mathbf{P}[\mathsf{VE}] \otimes \mathbf{P}[\mathsf{OC}] \otimes \mathbf{P}[\mathsf{AD}|\mathsf{VE}] \otimes \mathbf{P}[\mathsf{GG}|\mathsf{VE}, \mathsf{OC}] \otimes \mathbf{P}[\mathsf{GW}|\mathsf{AD}, \mathsf{GG}] \otimes \mathbf{P}[\mathsf{TC}|\mathsf{OC}]
\end{aligned}$$

Intuitively, each random variable is influenced only by the value of its direct parents, or, more precisely, it is not directly influenced by any variable but its parents. For example, $\mathsf{GlobalWarming}$ is directly influenced by $\mathsf{AirborneDust}$ and $\mathsf{GreenhouseGases}$, but not by $\mathsf{VolcanicEruption}$ or $\mathsf{OilConsumption}$. It follows that one easy way to sample from this distribution is to sample each variable according to the probability distribution that corresponds to the values of its parents. (The variables need to be sampled in topological order for the values of the parents to be available when needed.)

∎

We now give a rigorous definition of the formalism.

3.3.1 Probabilistic semantics

Definition 3.4 A (static) Bayesian network over a set of variables $\mathbf{U} = \{U_1, ..., U_n\}$ is a tuple $\mathcal{B} = \langle \mathcal{G}, \Theta \rangle$ consisting of the following two components:

1. A *directed acyclic graph* $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$, where $\mathbf{V}$ and $\mathbf{E}$ are the set of vertices and edges, respectively. The vertices in $\mathbf{V}$ are in a one-to-one correspondence with the variables in $\mathbf{U}$, which they represent. The set of edges $\mathbf{E}$ defines a binary, ordered, acyclic, irreflexive relation over the set of vertices $\mathbf{V}$.

2. A *parameterization* or *quantification* Θ of $\mathcal{G}$, which associates to each vertex V of $\mathbf{V}$ a *conditional probability distribution* $\mathbb{P}[V|Pa[V]]$, where $Pa[V] = \{V' : \langle V', V \rangle \in \mathbf{E}\}$ is the set of all vertices of $\mathbf{V}$ which are the origin of an edge in $\mathbf{E}$ whose extremity is V.

It can be shown that, under regularity assumptions which are always satisfied in the discrete case, a Bayesian network $\mathcal{B}$ induces a joint probability distribution over the Cartesian product of the respective domains of all variables in $\mathbf{U}$. This distribution is given by the outer product:

$$\mathbb{P}[\mathbf{U}] = \prod_{i=1}^{n} \mathbb{P}[U_i|Pa[U_i]] ,$$

where $\mathbb{P}[X]$ denotes the entire probability distribution of X, and $\mathbb{P}[X|Y]$ denotes the collection of probability distributions of X given all instantiations of Y, indexed by the instantiations of Y.

Equivalently, the individual probabilities of instantiations of $\mathbf{U}$ are given by:

$$\mathbb{P}[\mathbf{u}] = \prod_{i=1}^{n} \mathbb{P}[u_i|\{u_j : U_j \in Pa[U_i]\}] .$$

3.3.2 Structural independence assumptions

It follows from the above definition that the network graph $\mathcal{G}$ encodes a number of conditional independence assertions that the induced probability will necessarily satisfy, regardless of the quantification Θ. To emphasize this point, we sometimes qualify them as *structural*.

An independence assertion is a statement of the form "X and Y are conditionally independent given Z", usually written $X \perp Y|Z$, and which formally means that, for all possible tuples $\langle x, y, z\rangle \in dom[X] \times dom[Y] \times dom[Z]$, the equalities $\boldsymbol{P}[x|y, z] = \boldsymbol{P}[x|z]$ and $\boldsymbol{P}[y|x, z] = \boldsymbol{P}[y|z]$ hold. Informally, given the knowledge that $Z = z$, the distribution of X would not be affected by any additional information about Y. The above holds even if X, Y, and Z are sets of variables, which need not be necessarily disjoint.

As shown in [Pea88, SS90], the structure of the graph encodes a number of conditional independence assertions, which can be logically derived from a set of axioms known as the *graphoid axioms*. The complete set of independence assertions that can be derived from the graph is captured by a graph-theoretic relation known as *d-separation*. Specifically, $\mathbf{Z}$ d-separates $\mathbf{X}$ and $\mathbf{Y}$, if and only if $\mathbf{X} \perp \mathbf{Y}|\mathbf{Z}$ derives from the network graph and the graphoid axioms, *i.e.*, if and only if $\mathbf{X} \perp \mathbf{Y}|\mathbf{Z}$ structurally.

A complete characterization of d-separation is provided by the notion of *active path*. Intuitively, an active path between two variables X and Y reflects a chain of structural dependences between neighboring variables along the path. Active paths are structural properties of a Bayesian network, and depend exclusively on the structure of $\mathcal{G}$, and on which nodes of $\mathbf{V}$ correspond to *evidence variables*, *i.e.*, variables whose value is externally observed.

Definition 3.5 Let $\mathcal{G} = \langle \mathbf{V}, \mathbf{E}\rangle$ be the graph of a Bayesian network, and let $\mathbf{V}^* \subseteq \mathbf{V}$ be the subset of evidence nodes in $\mathbf{V}$. Let us say that a node V on an undirected path in $\mathcal{G}$ is the focus of a v-structure, or *v-node* for short, if the path contains two edges of $\mathcal{G}$ that are directed toward V. Then, the *active paths* between elements of $\mathbf{V} \setminus \mathbf{V}^*$ are exactly the undirected paths in $\mathcal{G}$ that satisfy the two conditions:

1. every v-node on the path either is in $\mathbf{V}^*$ or has at least one descendent in $\mathbf{V}^*$;

2. every non-v-node node on the path is in $\mathbf{V} \setminus \mathbf{V}^*$ (including the two extremities of the path).

The following fact links the concept of active path to the structural independence semantics of Bayesian networks for individual variables.

Fact 3.6 *In a Bayesian network $\mathcal{B} = \langle \mathcal{G}, \Theta \rangle$, two nodes X and Y are structurally independent given a set of nodes $\mathbf{Z}$, i.e., $X \perp Y | \mathbf{Z}$, if and only if there is no active path between X and Y when $\mathbf{Z}$ is taken as the evidence set.*

The following fact allows us to generalize structural independence semantics to sets of variables.

Fact 3.7 *Two sets of nodes $\mathbf{X}$ and $\mathbf{Y}$ are independent given a third set $\mathbf{Z}$, denoted $\mathbf{X} \perp \mathbf{Y} | \mathbf{Z}$, if and only if $X \perp Y | Z$ for all $X \in \mathbf{X}$ and $Y \in \mathbf{Y}$.*

We also have the following corollary, which conveniently summarizes some of the structural independence assertions encoded by a Bayesian network.

Corollary 3.8 *In a Bayesian network $\mathcal{B} = \langle \mathcal{G}, \Theta \rangle$, any variable is structurally independent of its non-descendents given its parents in $\mathcal{G}$, irrespective of Θ.*

To illustrate how those results apply, we now return to our previous example in Figure 3.1.

Example 3.9 Going back to the example of Figure 3.1, it is easy to see that $\mathsf{GW} \not\perp \mathsf{VE}$ and $\mathsf{GW} \not\perp \mathsf{OC}$, since each of VolcanicEruption and OilConsumption is an ancestor of GlobalWarming, and there is no evidence node along the undirected path from VolcanicEruption to OilConsumption via GlobalWarming.

More interestingly, we also have $\mathsf{GW} \not\perp \mathsf{TC}$, even though GlobalWarming and ToxicContaminant do not influence each other in this model; they are nonetheless dependent, as both have OilConsumption as common ancestor. However, we have $\mathsf{GW} \perp \mathsf{TC} | \mathsf{OC}$, since ToxicContaminant is no longer relevant to GlobalWarming once OilConsumption is observed.

Conversely, we have $\mathsf{VE} \perp \mathsf{OC}$, since VolcanicEruption and OilConsumption are *a priori* independent in the model. However, we have $\mathsf{VE} \not\perp \mathsf{OC}|\mathsf{GG}$, since VolcanicEruption and OilConsumption become competing explanations of a known outcome, once GreenhouseGases is observed. Simlarly, we have $\mathsf{VE} \not\perp \mathsf{OC}|\mathsf{GW}$, since the observation of GlobalWarming opens an active path between VolcanicEruption and OilConsumption.

3.3.3 Causal interpretation

Another useful intuition is given by the *causal interpretation* of Bayesian networks. Under this interpretation, the network topology directly models the causal relationships between the network variables, so that each directed edge between a pair of variables represents a direct causal dependency from the origin of the edge toward its extremity. Thus, each variable in the network jointly and stochastically depends on its parent variables, and none other.

Although the causal interpretation is responsible for most of the intuitive appeal of the formalism, and was, indeed, one of the early motivations for its creation, it should be noted that Bayesian networks need not obey the causal interpretation. In particular, the same probability distribution can often be represented by many Bayesian networks with distinct topologies, most of which are causally incompatible with each other. In general, however, for distributions that are modeled after causal physical systems, causally faithful networks will typically provide the most compact representation—an observation which has been exploited for learning causal relations from data [HMC97] (see also [HS94]).

3.4 Inference

Bayesian inference is the process by which we answer questions such as: "Given that $A = a_1$ and $B = b_2$, what is the joint probability that $C = c_3$ and $D = d_4$"? More precisely, a Bayesian network inference algorithm computes probability distributions of the form $\mathbf{P}[C, D|A = a_1, B = b_2]$, where $A = a_1$ and $B = b_2$ are the evidence, C

and D are the *query*, and $\mathbf{P}$ is a probability distribution given as a Bayesian network over the variables A, B, C, D, and possibly others.

One of the most successful and widely used inference algorithms for (static) Bayesian network is known as the *junction tree* or *join tree* algorithm, originally due to Lauritzen and Spiegelhalter [LS88], and later extended in [JLO90]. A detailed procedural presentation may be found in [HD94], of which the following overview is a summary.

3.4.1 Junction trees

The algorithm is composed of two major steps. The first step consists of a preprocessing operation, in which the Bayesian network is transformed into a related structure called a *junction tree*. The second step is the actual inference step, in which the distribution over the query variables is calculated from the junction tree and the supplied evidence.

Definition 3.10 A *junction tree*, also called *join tree*, *clique tree*, or *cluster tree*, defined over a set of variables $\mathbf{U}$, is a tuple $\Upsilon = \langle \mathcal{F}, \Phi \rangle$, consisting of the following two components:

1. An *undirected tree* $\mathcal{F}$, where with each node of $\mathcal{F}$ is associated a *cluster* of variables, *i.e.*, a non-empty subset of $\mathbf{U}$. The clusters in $\mathcal{F}$ are required to satisfy the *running intersection property*, which requires that all clusters in $\mathcal{F}$ on the path between any two clusters $\mathbf{X}$ and $\mathbf{Y}$ contain all the variables in $\mathbf{X} \cap \mathbf{Y}$. It is also required that, for each variable $U \in \mathbf{U}$, the *family* of U, which is defined as $\{U\} \cup Pa[U]$, be contained in at least one of the clusters.

2. A set Φ of *belief potentials*, which are defined as functions mapping each instantiation of a set of variables to a non-negative real number. Φ contains the following potentials:

   (a) *cluster potentials*: each cluster of $\mathcal{F}$ is attributed a potential over the variables it contains;

(b) *sepset potentials*: each *sepset* of $\mathcal{F}$ is associated with a potential over its variables, where a sepset, or *separator set*, is defined as the intersection of two adjacent clusters in $\mathcal{F}$.

The junction tree is said to be *locally consistent* or *calibrated*, if, for each cluster $\mathbf{X}$ and adjacent sepset $\mathbf{S}$, the potential of $\mathbf{S}$ is identical to the potential of $\mathbf{X}$ marginalized over the variables in $\mathbf{S}$:

$$\sum_{\mathbf{X} \setminus \mathbf{S}} \boldsymbol{\phi}_{\mathbf{X}} \quad = \quad \boldsymbol{\phi}_{\mathbf{S}} \ .$$

A junction tree Υ encodes a real-valued function over its set of variables $\mathbf{U}$, according to:

$$\mathbb{P}[\mathbf{U}] \quad = \quad \frac{\prod_i \boldsymbol{\phi}_{\mathbf{X}_i}}{\prod_j \boldsymbol{\phi}_{\mathbf{S}_j}} \ ,$$

where $\boldsymbol{\phi}_{\mathbf{X}_i}$ and $\boldsymbol{\phi}_{\mathbf{S}_j}$ respectively denote the cluster and sepset potentials of Υ. In the normalized case, where $\sum_{\mathbf{U}} \mathbb{P}[\mathbf{U}] = 1$, the function encoded by Υ is a joint probability distribution over its set of variables $\mathbf{U}$.

For any calibrated junction tree satisfying the above constraints, it can be shown that the marginal probability over any cluster of Υ is equal to the cluster potential. In other words, $\boldsymbol{\phi}_{\mathbf{X}} = \mathbb{P}[\mathbf{X}]$ for any cluster $\mathbf{X}$. It follows that the probability distribution over any variable or set of variables Z can be computed from any cluster or sepset $\mathbf{X}$ that contains Z, by simple marginalization, as follows:

$$\mathbb{P}[Z] \quad = \quad \sum_{\mathbf{X} \setminus \{Z\}} \boldsymbol{\phi}_{\mathbf{X}} \ .$$

3.4.2 Graphical construction

The construction of a junction tree from a given Bayesian network involves a sequence of graph transformations, whose steps are described next.

Moralization

The first step is the construction of the *moral graph* $\mathcal{G}_M$, which is an undirected graph obtained by "marrying the parents" of each variable in the original Bayesian network [LS88, LDLL90]. More precisely, $\mathcal{G}_M$ is an undirected graph $\langle \mathcal{V}_M, \mathcal{E}_M \rangle$, where $\mathcal{V}_M = \mathcal{V}$ as defined in the given Bayesian network. $\mathcal{E}_M$ contains one undirected edge for each directed edge in $\mathcal{E}$, and one undirected edge between each pair of vertices that are the parents of the same vertex in $\mathcal{G}$.

Triangulation

The goal of the second transformation is to ensure that the moral graph is *triangulated*, or *chordal*, that is, every cycle of length four or greater contains a pair of vertices connected by an edge not included in the cycle. The transformation consists of adding edges to the moral graph until it achieves chordality. This is commonly done using heuristics that seek to minimize the *weight*, or domain size, of the maximal cliques that result from the triangulation (see [Kjæ90] for details).

This is an important phase, as the quality of the triangulated graph obtained in this phase ultimately determines the computational cost of inference. Even though Kjærulff's heuristic typically gives excellent results, the problem is known to be NP-hard.

Clustering

The third operation is the identification of all maximal cliques in the triangulated moral graph. The maximal cliques define the clusters in the junction tree. An identification procedure for arbitrary chordal graphs proposed by Golumbic [Gol80] is described in [GVP90], based on a particular ordering of the nodes which can be obtained as Tarjan and Yannakakis suggest [TY84]. Huang and Darwiche [HD94] present an alternative method which identifies the maximal cliques as the moral graph is being triangulated.

Junction

The last step of the construction is to arrange the set of clusters obtained in the previous step into a junction tree that satisfies the running intersection property. An efficient greedy construction is offered by Jensen and Jensen [JJ94].

3.4.3 Probability propagation

Once the structure of the junction tree has been determined, the various cluster and sepset potentials must be determined before inference can be conducted. The necessary steps are summarized in the following sections.

Initialization

The first required operation consists of assigning initial potentials to the junction tree, so that it encodes the same probability distribution as the given Bayesian network $\mathcal{B}$. In other words, potentials are assigned to all clusters and sepsets to satisfy the global semantic constraint:

$$\mathbb{P}[\mathbf{U}] = \frac{\prod_i \boldsymbol{\phi}_{\mathbf{X}_i}}{\prod_j \boldsymbol{\phi}_{\mathbf{S}_j}} .$$

This is obtained as follows. Each cluster and sepset is first initialized with the constant potential which assigns the value 1 to any instantiation of the cluster or sepset. Then, for each variable U in $\mathbf{U}$, an arbitrary cluster $\mathbf{Y}$ that includes $\{U\} \cup Pa[U]$ is selected, and the potential $\boldsymbol{\phi}_{\mathbf{Y}}$ of $\mathbf{Y}$ is multiplied by the conditional probability distribution $\mathbb{P}[U|Pa[U]]$ associated with U in the Bayesian network. It can be shown that the graphical construction of the junction tree guarantees the existence of at least one suitable cluster $\mathbf{Y}$ for each variable U.

Calibration

Once the junction tree globally encodes the Bayesian network distribution, it remains to make it locally consistent, ensuring that each sepset potential agrees with the potentials of its adjacent clusters when marginalized over the sepset variables. In other words, for each cluster $\mathbf{X}$ and adjacent sepset $\mathbf{S}$, it is required that:

$$\sum_{\mathbf{X} \setminus \mathbf{S}} \boldsymbol{\phi}_{\mathbf{X}} = \boldsymbol{\phi}_{\mathbf{S}} .$$

The local consistency requirement is achieved by a procedure known as *calibration*, *probability propagation*, or *message passing*. The calibration procedure consists of a series of local transformations, called *message passes*, between a pair of adjacent clusters $\mathbf{X}$ and $\mathbf{Y}$ and their shared sepset $\mathbf{S}$. A message from $\mathbf{X}$ to $\mathbf{Y}$ causes the potential of $\mathbf{S}$ to become locally consistent with $\boldsymbol{\phi}_{\mathbf{X}}$, while keeping the global semantic constraint invariant. The messages are ordered in such a way that the consistency achieved by previous messages is preserved. The details of the calibration process may be found in [HD94].

Marginalization

As mentioned earlier, once local consistency and global semantics are achieved, the marginal probability distribution over a set of variables may be readily extracted from any cluster that includes it, provided such a cluster exists. In particular, queries involving single variables can always be answered by the junction tree.

Joint queries

In the general case of queries $\mathbb{P}[\mathbf{Z}]$ involving a set of variables $\mathbf{Z} \subseteq \mathbf{U}$, the easiest approach is to require that all the queried variables be jointly contained in at least one cluster. This condition can be achieved during the graphical construction of the junction tree, by adding an undirected edge between each pair of variables in $\mathbf{Z}$ to the moralized graph $\mathcal{G}_M$, before the clusterization step as previously described. This will force $\mathbf{Z}$ to form a clique, which will either cause $\mathbf{Z}$ to be in a cluster of its own, or to be included in an even larger cluster.

It is noted that in the case where $\mathbf{Z} = \mathbf{U}$, the junction tree reduces to a single huge cluster containing the entire set of variables $\mathbf{U}$.

3.4.4 Evidential conditioning

The above procedure answers unconditional queries of the form $\boldsymbol{P}[\mathbf{Z}]$, with $\mathbf{Z} \subseteq \mathbf{U}$, although a slight modification will allow conditional queries of the form $\boldsymbol{P}[\mathbf{Z}|\mathbf{W} = \mathbf{w}]$, for any $\mathbf{W} \subseteq \mathbf{U}$ and any $\mathbf{w} \in dom[\mathbf{W}]$.

To compute conditional probabilities of the form $\boldsymbol{P}[\mathbf{Z}|\mathbf{W} = \mathbf{w}]$ using the junction tree, one computes $\boldsymbol{P}[\mathbf{Z}, \mathbf{W} = \mathbf{w}]$ and $\boldsymbol{P}[\mathbf{W} = \mathbf{w}]$, and uses the definition of conditional probability to obtain:

$$\boldsymbol{P}[\mathbf{Z}|\mathbf{W} = \mathbf{w}] \;=\; \frac{\boldsymbol{P}[\mathbf{Z}, \mathbf{W} = \mathbf{w}]}{\boldsymbol{P}[\mathbf{W} = \mathbf{w}]} \;,$$

where the denominator $\boldsymbol{P}[\mathbf{W} = \mathbf{w}]$ is advantageously computed as the sum:

$$\boldsymbol{P}[\mathbf{W} = \mathbf{w}] \;=\; \sum_{\mathbf{Z}} \boldsymbol{P}[\mathbf{Z}, \mathbf{W} = \mathbf{w}] \;.$$

Incorporation

The required modification consists of incorporating the available evidence into the potentials of the junction tree during or immediately after the initialization phase. This is done by generating one *evidence factor* Λ_{W_i} for each variable $W_i \in \mathbf{W}$, defined as a function over $dom[W_i]$ that is 0 everywhere, except for $W_i = w_i$ where its value is 1. Each evidence factor Λ_{W_i} is incorporated multiplicatively into the potential of a suitable cluster, exactly like the conditional probability distributions from the Bayesian network. Since Λ_{W_i} depends only on the single variable W_i, it may be incorporated into the potential of any cluster that contains W_i.

Normalization

Following evidence incorporation, all probabilities subsequently computed using the junction tree will be probabilities of events conjoined with the incorporated evidence. It follows that a query over $\mathbf{Z}$ will produce the distribution $\boldsymbol{P}[\mathbf{Z}, \mathbf{W} = \mathbf{w}]$, as required. As for the normalization scalar $\boldsymbol{P}[\mathbf{W} = \mathbf{w}]$, it is obtained by marginalizing the distribution $\boldsymbol{P}[\mathbf{Z}, \mathbf{W} = \mathbf{w}]$ over the empty set, which amounts to taking the sum over all instantiations of $\mathbf{Z}$. The conditional distribution of interest $\boldsymbol{P}[\mathbf{Z}|\mathbf{W} = \mathbf{w}]$ is then given by:

$$\boldsymbol{P}[\mathbf{Z}|\mathbf{W} = \mathbf{w}] \quad = \quad \frac{\boldsymbol{P}[\mathbf{Z}, \mathbf{W} = \mathbf{w}]}{\sum_{\mathbf{Z}} \boldsymbol{P}[\mathbf{Z}, \mathbf{W} = \mathbf{w}]} \ .$$

3.5 Dynamic Bayesian networks

Dynamic Bayesian networks (DBNs) [DK89] extend the Bayesian network framework by providing an explicit discrete temporal dimension. As a Bayesian network represents a probability distribution over the possible states of a finite system, a dynamic Bayesian network represents a probability distribution over the possible state histories of a time-invariant process.

DBNs have been exploited in a number of interesting applications, ranging from forecasting [DGH92] to speech recognition [ZR98], to the control of a water purification station [JKOP89], to the study of automotive traffic [HKM$^+$94] and vehicle behavior [FHKR95].

A DBN is essentially a discrete stochastic transition model factored over a number of random variables, and which describes a single step of the transition dynamics of the process. The time invariance assumption ensures that this generic model correctly represents the process dynamics at any point in time. The transition model is encoded as a k-TBN, or k-time slice temporal Bayesian network. A k-TBN is a fragment of Bayesian network over a set of time-dependent variables, that represents *semi-Markovian stochastic transition of order* $k-1$, *i.e.*, a conditional probability distribution over the variables given their $k-1$ previous values. Unless otherwise specified, the term dynamic Bayesian network is usually used to denote a 2-TBN, which is necessarily Markovian.

Definition 3.11 A *k-time slice temporal Bayesian network*, or k-TBN, over a set of variables $\mathbf{U} = \{U_1, ..., U_n\}$ is a tuple $\mathcal{B} = \langle \mathcal{G}, \Theta \rangle$ consisting of the following two components:

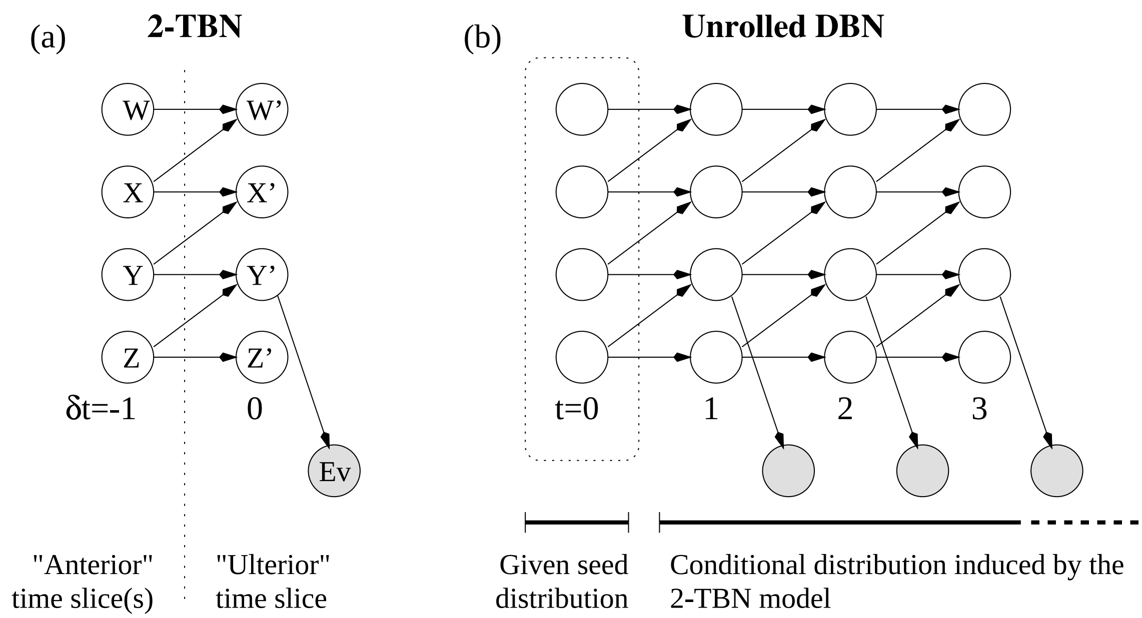


Figure 3.2: BN interpretation of the DBN semantics: (a) simple DBN represented as a 2-TBN; (b) resulting unrolled BN over the first three time slices. Shaded nodes denote observable variables.

1. A *directed acyclic graph* $\mathcal{G} = \langle \mathbf{V}, \mathbf{E} \rangle$, whose vertices are organized in k layers. The layers are denoted by relative time indices in $\{-k+1, ..., -1, 0\}$, or, equivalently, by absolute indices in $\{t-k+1, ..., t-1, t\}$, where t is a specified time reference. $\mathbf{V}$ is such that, to each variable $U \in \mathbf{U}$ corresponds exactly one vertex $U^{(t)}$ in the layer having index t, and at most one vertex $U^{(t-d)}$ in each layer of index $t-d$ with $0 < d < k$. $\mathbf{E}$ defines a binary acyclic relation, in which all edges are further restricted to belong to $\mathbf{V} \times \mathbf{V}^{(t)}$, where $\mathbf{V}^{(t)}$ denotes the last layer of $\mathbf{V}$ of index t.

2. A *parameterization* or *quantification* Θ, which associates to each vertex $U^{(t)}$ in the last layer of $\mathbf{V}$, a conditional probability distribution $\boldsymbol{P}[U^{(t)}|Pa[U^{(t)}]]$.

The semantics of DBNs are easily understood if we consider the operation of "unrolling" a k-TBN into an ordinary BN. As illustrated in Figure 3.2, a k-TBN can be unrolled into a BN simply by duplicating the nodes in the right-most column of the k-TBN over and over again, along with their associated conditional probability distributions, in the obvious way. (We call *ulterior* the right-most layer of the k-TBN, and *anterior* the $k-1$ layers on which the ulterior layer depends.) This produces an ordinary BN, albeit arbitrarily large, depending on the number of time slices over which the DBN is unrolled. We call the variables of the DBN *instantaneous* when indexed by time; *i.e.*, the nodes in the unrolled BN represent as many instantaneous variables.

To complete the conversion, it remains to specify an initial distribution for the very first $k-1$ slices of the unrolled DBN (which correspond to the anterior layers of the first instantiation of the k-TBN in the unrolled DBN). This is because the unrolled DBN, being a BN, must specify a well-defined distribution, and all that we have at this point is a conditional distribution conditioned on the node in those first $k-1$ layers. For example, Figure 3.2(a) shows a 2-TBN with four state variables and one observable response variable. Figure 3.2(b) shows the corresponding unrolled DBN, which is a well-specified BN when adjoined with an initial "seed" distribution over the variables in layer $t=0$. We now provide a more rigorous characterization of the DBN semantics.

Formally, we define the probabilistic semantics of a dynamic Bayesian network, assuming a primordial seed distribution over $k-1$ time slices, where $k-1$ is the Markovian order of the transition model represented by the k-TBN. For simplicity, the primordial distribution is assumed to span the $k-1$ non-positive time indices $-k+2, ..., 0$. Then, given the joint probability distribution $\boldsymbol{P}[\mathbf{U}^{(-k+2)..(0)}]$ over all variables in the first $k-1$ time slices, the joint probability distribution $\boldsymbol{P}[\mathbf{U}^{(-k+2)..(\tau)}]$ over all variables from the origin up to time $\tau > 0$ is given by:

$$\boldsymbol{P}[\mathbf{U}^{(-k+2)..(\tau)}] \quad = \quad \boldsymbol{P}[\mathbf{U}^{(-k+2)..(0)}] \otimes \prod_{t=1}^{\tau} \prod_{i=1}^{n} \boldsymbol{P}[U_i^{(t)} | Pa[U_i^{(t)}]] \ .$$

More generally, for $0 < \tau_1 \leq \tau_2$, the conditional probability distribution over all variables in $\mathbf{U}$ from time τ_1 through τ_2, given the $k-1$ preceding values of those variables (*i.e.*, at times $\tau_1 - k + 1, ..., \tau_1 - 1$), is:

$$\boldsymbol{P}[\mathbf{U}^{(\tau_1)..(\tau_2)}|\mathbf{U}^{(\tau_1-k+1)..(\tau_1-1)}] \quad = \quad \prod_{t=\tau_1}^{\tau_2}\prod_{i=1}^{n}\boldsymbol{P}[U_i^{(t)}|Pa[U_i^{(t)}]] \ .$$

Similarly, the independence assumptions encoded by a dynamic Bayesian network are defined in terms of the unrolled Bayesian network. Specifically, two sets of instantaneous variables $\mathbf{X} = \{U_{i_1}^{(t_{i_1})}, ..., U_{i_x}^{(t_{i_x})}\}$ and $\mathbf{Y} = \{U_{i_1}^{(t_{i_1})}, ..., U_{i_y}^{(t_{i_y})}\}$ are conditionally independent given a third set $\mathbf{Z} = \{U_{i_1}^{(t_{i_1})}, ..., U_{i_z}^{(t_{i_z})}\}$, if the vertices of the unrolled Bayesian network that correspond to $\mathbf{X}$ and those that correspond to $\mathbf{Y}$ are d-separated given those corresponding to $\mathbf{Z}$. In particular, any instantaneous variable is independent of its non-descendents given its parents, in the unrolled Bayesian network. It is stressed that the above independence assumptions only make sense in terms of *instantaneous variables*, which are time-dependent random variables considered at specific points in time, as opposed to the variables themselves. Indeed, whereas a DBN variable translates to a potentially infinite set of time-indexed vertices in the unrolled BN, an instantaneous variable corresponds to a single vertex in the unrolled BN.

In spite of the compact representation of evolving processes provided by the DBN model, this representation is not directly amenable to probabilistic inference over multiple time slices. Probabilistic reasoning usually requires that the DBN be first unrolled into a BN, on which the previously described inference algorithm can be used.

One major problem with this procedure is that, not only is DBN inference unable to exploit the compactness of the representation along the temporal axis, but, as will become evident in the next chapter, it generally fails to make use of the very independence relations between variables that are expressed by the graphical structure of the model. The latter limitation renders algorithms that try to track dynamic systems exactly [Kjæ92] impractical for complex problems. This phenomenon is pathological of temporal inference, and would seem to present a fundamental impediment to using the Bayesian formalism for reasoning with dynamic systems.

3.6 Other structured representations

We have seen that DBNs provide a compact and flexible representation of multivariate Markovian processes. Other representations exist, such as the somewhat less expressive Factorial HMMs (FHMMs) of [GJ96].

FHMMs may be thought of as a special kind of DBN with a number of state and observable variables per time slice, arranged in such a way that the state variables are hidden and evolve without influencing each other, and the observable variables are influenced by any or all the state variables in the same time slice.

Chapter 4

Tractable inference

Given the obvious similarity of the DBN representation for structured stochastic processes to the BN represenation for atemporal distributions, a reasonable question to ask is whether conditional independence assumptions can be exploited to make inference practical in large DBNs, just as in the case of atemporal BNs. Unfortunately, this question has a resounding negative answer, for reasons discussed in Section 4.1.

The next question to ask, then, is whether a scalable approximate algorithm could be devised at the expense of the requirement for an exact solution. It turns out that such an algorithm exists, which can be shown to produce reasonably accurate answers, under the right conditions. In this chapter, we present such an algorithm and experimentally study its performance on actual models. The various aspects of its analysis are deferred until the next two chapters.

4.1 Debunking the myth of structure

In view of the success of Bayesian networks at providing efficient representation and inference of static stochastic systems, it would be expected that DBNs would enjoy analogous properties for dynamic stochastic processes. As described in Section 3.5, the structure of DBNs supports a decomposed representation of the trajectory distribution. Unfortunately, the structure is, in most cases, unexploitable for inference, for very fundamental reasons. The main cause of this difficulty resides in the fact

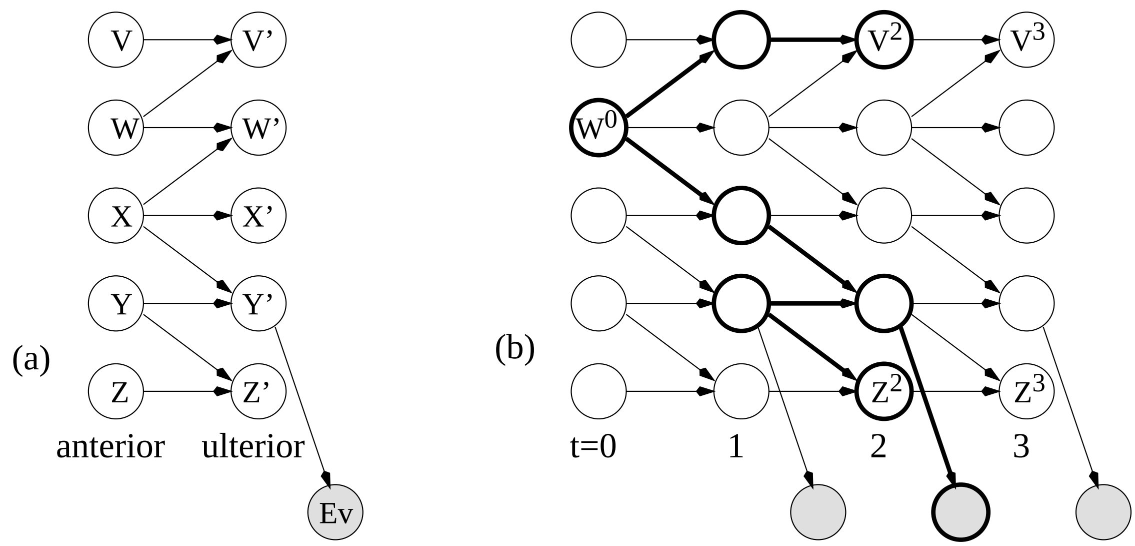


Figure 4.1: Illustration of propagating correlations in stochastic processes: (a) structured DBN represented as a 2-TBN; (b) unrolled DBN over the first three time slices. The figure shows one of the active paths between $V^{(2)}$ and $Z^{(2)}$, using thicker lines to show the edges and nodes involved.

that, even if the interactions between variables are very structured and local, as they would be in a DBN with a sparse graph, correlations quickly propagate throughout the entire system, and eventually all variables become correlated with each other. At this point and beyond, the belief state no longer has a decomposed representation, and inference is rendered intractable.

To illustrate, consider the DBN of Figure 4.1(a), which has a fairly sparse graph, and in which all direct dependences of one variable on another are fairly localized. Furthermore, all variables are assumed to start out being independent at time 0, *e.g.*, as would be the case if the start state were known exactly (a point distribution) or not at all (the uniform distribution). Nevertheless, if we consider the unrolled BN of Figure 4.1(b), for now disregarding the shaded evidence nodes, we can easily see that all the variables are structurally correlated at time 3, according to the rules of d-separation. In particular, any pair of variables at time 3 share a common ancestor at time 0, whose influence propagates to both via variables at times 1 and 2, causing their correlation (for instance, $V^{(3)}$ and $Z^{(3)}$ have a common ancestor $W^{(0)}$). The situation is even worse if we bring in the evidence nodes; in this case, it takes only 2 time steps for all the state variables to become fully correlated. For example, there are two active paths between $V^{(2)}$ and $Z^{(2)}$, one of which is shown in Figure 4.1(b).

In general, unless a process decomposes into completely independent subprocesses, the belief state will become fully correlated very early in time. The independence condition to avoid such a fateful outcome is unrealistically stringent, as it requires not only that the respective states of the processes do not influence each other, but also that the two processes do not influence the same evidence variable, even indirectly. As any factored decomposition of a distribution rests on some form of conditional independence, it follows that, for most stochastic processes of interest, no decomposition of the belief state is possible. This phenomenon, which has been tacitly observed by several researchers [Pro92, GJ96], is perhaps the most serious impediment to applying probabilistic reasoning methods to dynamic systems.

Notation In the sequel, we conventionally denote the ulterior time slice of a transition model using the mark $^{\triangleright}$ in exponent of the symbol it applies to, as in $X^{\triangleright}$. Similarly, we denote the anterior time slice by the modified $^{\triangleleft}$ as in $Y^{\triangleleft}$.

Proposition 4.1 *Let $\mathcal{G}$ be the directed graph of a 2-TBN model, assumed devoid of intra-time-slice edges. Let $\bar{\mathcal{G}}$ be the directed graph obtained by extending $\mathcal{G}$ with backward edges going from all (non-evidence) ulterior nodes $Z^{\triangleright}$ to the corresponding anterior nodes $Z^{\triangleleft}$, i.e., $\bar{\mathcal{G}} = \mathcal{G} \cup \{\langle Z^{\triangleright}, Z^{\triangleleft}\rangle\}$. If two ulterior nodes $X^{\triangleright}$ and $Y^{\triangleright}$ are connected by a directed path that involves n backward edges in $\bar{\mathcal{G}}$, then the corresponding variables X and Y are structurally correlated after at most n time slices in the unrolled DBN.*

Proof It suffices to observe that any directed path in $\bar{\mathcal{G}}$ involving n backward edges can be transformed into an active path over n time slices in the unrolled DBN. Specifically, $\exists W$ such that $W^{(t-n)}$ is a common ancestor of both $X^{(t)}$ and $Y^{(t)}$, provided that $t \geq n$. ◻

Corollary 4.2 *If every pair of (non-evidence) ulterior nodes is connected by a directed path in $\bar{\mathcal{G}}$ involving at most n backward edges, then all variables are structurally correlated after at most n time slices.*

This general phenomenon is unfortunate, whereby even loosely coupled variables are bound to become irretrievably correlated, defeating all hope of exploiting any model structure for exact inference. However, it is also the case that, had two subsets of variables not interacted at all with each other, no correlation would have ever occured between them. By a continuity principle, this suggests that, in a system where most dependence relations between variables are weak or altogether missing, many of the induced correlations should be weak, and propagate slowly. Even though, in all likelihood, all variables are quickly going to become correlated with each other, the hope is that many of those correlations will be weak enough that they may safely be ignored without incurring excessive error. As these decouplings result in independence relations between variables, this approximation provides us with a factored belief state, much more compactly representable than the exact one, and on which inference should be much easier to conduct.

In particular, a filtering algorithm as described in Section 3.2, which would, in addition, approximate away properly chosen weak correlations at every step to ensure a compactly representable belief state, should stand to benefit considerably from a computational standpoint. For example, even though many or all variables in the system are likely to be structurally correlated, if those variables can be grouped in clusters which have provably weak correlations with one another, the belief state could be advantageously approximated as a product of smaller belief states over the individual clusters. However, the risks associated with this idea are clear: the errors introduced at every step by the approximation may accumulate over time to make the results of the chain of inferences completely erroneous. The hope is that the very stochasticity of the process will prevent such an unlimited accumulation of error, and instead strike a balance where accumulated errors are progressively forgotten, as fast as new approximation errors are introduced.

4.2 Structured filtering algorithm

We now present a general algorithm that implements the above ideas for DBN inference, and study its empirical performance on data. Its analysis is deferred till Chapters 5 and 6.

As we saw earlier, the key problem with DBN inference lies with the representation of the instantaneous belief state at time t: its size grows exponentially with the number of state variables. The simple idea behind our approximation algorithm is to approximate the joint belief state as a collection of marginals over chosen clusters of variables. This amounts to representing the joint distribution as its various orthogonal projections on the chosen clusters. Since the size of each marginal is exponential in the number of the variables it covers, the total size of the representation will remain small as long as the size of the marginals is bounded sufficiently.

Example 4.3 To illustrate, consider a system composed of a fairly large number of cars on a freeway. A correct representation of the belief state of the system at some time t is given by the joint distribution over the state of all the cars at time t. Assuming that the state of each car is described by a similar set of discrete variables, such as its position and speed, it is easy to see that the size of the joint distribution grows exponentially with the number of cars involved. Alternatively, we can choose to track the state of the system by maintaining a marginal belief state for each car in the system. This leads to a much more compact representation, linear in the the number of cars, at the expense of losing all correlations between cars in our estimation of the global state. If desired, the joint belief state can be (approximately) reconstructed from this representation by taking the product of all the marginals.

Clearly, the approximation is incorrect, and could lead to mistakes, such as attributing greater probability to the event that two cars occupy the same position, than the exact belief state would. However, the error will be small as long as the individual cars are almost independent.

▟

Example 4.4 In the previous aproximation scheme, a separate marginal is used for each car, which is appropriate if the pairwise correlation between cars is uniformly weak. In more realistic situations, there may be a group of cars traveling together, in which case it would make sense to gather them all in a single cluster, and track them using a single joint marginal.

In an even more difficult case, such as in a traffic jam, each car may interact significantly with its neighbors, causing a non-negligible correlation with a small number of other cars. However, and contrarily to the previous case, the only way to account for those correlations without resorting to the exact joint belief state, is to use clusters that overlap. In this case, the marginal belief states are no longer (trivially) independent in the approximate representation, but (pairwise) conditionally independent given the variables they have in common.

∎

Recall the notation of Section 3.1 for belief states, and denote by $\tilde{\boldsymbol{\sigma}}^{(t)}$ our compactly represented approximate belief state at time t. It is updated from the previous time slice using the same propagation-conditioning process as used for the true belief state $\boldsymbol{\sigma}^{(t\bullet)}$: first, $\tilde{\boldsymbol{\sigma}}^{(t-1)}$ is propagated through the transition model $\mathcal{S}$, to obtain $\hat{\boldsymbol{\sigma}}^{(\bullet t)}$; then, $\hat{\boldsymbol{\sigma}}^{(\bullet t)}$ is conditioned on the observed response at time t, to give $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$. However, as new correlations are introduced by the update process, $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ does not usually admit a compact representation. In order to maintain the feasibility of the update process, it is necessary to further approximate $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$, typically by finding a similar distribution that admits a compact representation; the result is our new approximate belief state $\tilde{\boldsymbol{\sigma}}^{(t)}$. In the case of our freeway example, we would compute our new beliefs about the state of each vehicle by projecting $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ on the state space of the vehicle, and use the cross product of these individual belief states as our approximation $\tilde{\boldsymbol{\sigma}}^{(t)}$.

Naturally, for this scheme to be workable, it will have to bypass the explicit computation of $\hat{\boldsymbol{\sigma}}^{(\bullet t)}$ and $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$, which may not be compactly representable, and infer $\tilde{\boldsymbol{\sigma}}^{(t)}$ from $\tilde{\boldsymbol{\sigma}}^{(t-1)}$ in one operation. We now delve into the details of how this is achieved, first in the case of non-overlapping clusters, then in the more general case of overlapping clusters.

4.2.1 Independent clusters

The objective is now to apply the general idea presented above to the problem of monitoring a process described as a DBN. In all generality, we consider a process structured in terms of an ordered set of time-dependent state variables $\mathbf{X}_1, ..., \mathbf{X}_n$. The probability model of a DBN is typically described using a time-invariant 2-TBN, as defined in Section 3.5 and illustrated in Figure 3.2. The 2-TBN associates each variable $\mathbf{X}_k$ with a conditional probability distribution $\boldsymbol{P}[X_k^{(t)}|Pa[X_k^{(t)}]]$, where $Pa[X_k^{(t)}]$ can contain any variable at time $t-1$ and such variables at time t that preceed $\mathbf{X}_k$ in the total ordering—the latter condition being equivalent to the acyclicity requirement of Section 3.5. The model represents the conditional distribution over the state at time t given the values of the variables at time $t-1$, for any t. Recall that the two layers of the 2-TBN at $t-1$ and t are referred to as *anterior* and *ulterior*, respectively. It is useful to define the *canonical set of state variables* as $\{\mathbf{Y} : Y^{(t-1)} \in \cup_k Pa[X_k^{(t)}]\}$, *i.e.*, the set of variables that have a direct edge from the anterior layer to another variable in the ulterior layer. A 2-TBN is in *canonical form* if only the canonical variables are represented at time $t-1$; in other words, in a canonical form 2-TBN, the anterior layer contains only variables that have a direct influence on the same or another variable in the ulterior layer.

To capture the idea of a subprocess, we partition the set of canonical state variables into L disjoint subsets or clusters $\mathbf{S}_1, ..., \mathbf{S}_L$. The partition must satisfy the requirement that no variable in $\mathbf{S}_{l_1}$ may share an edge with a variable in $\mathbf{S}_{l_2}$ within the same time slice; *i.e.*, if $\mathbf{X} \in \mathbf{S}_l$, then $X^{(t)}$ cannot have $Y^{(t)}$ as a parent, where the variable $\mathbf{Y} \notin \mathbf{S}_l$.

The approximate filtering procedure for DBNs follows the same lines as the general procedure described in Section 3.2. At each point t in time, we wish to compute the current approximate belief state $\tilde{\boldsymbol{\sigma}}^{(t)}$, having at our disposal the preceding belief $\tilde{\boldsymbol{\sigma}}^{(t-1)}$ and the most recent observed response $R^{(t)}$. By the inductive hypothesis, in $\tilde{\boldsymbol{\sigma}}^{(t-1)}$ the clusters or subprocesses $\mathbf{S}_l$ are all independent. We propagate $\tilde{\boldsymbol{\sigma}}^{(t-1)}$ through the transition model, and condition the result $\hat{\boldsymbol{\sigma}}^{(\bullet t)}$ on the observed response $R^{(t)}$ to produce $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$; we then project it onto the partition of clusters $\{\mathbf{S}_l\}$, to give our updated approximate belief state $\tilde{\boldsymbol{\sigma}}^{(t)}$, whose marginals on the various clusters are given by $\tilde{\boldsymbol{\sigma}}^{(t)}[\mathbf{S}_l] = \hat{\boldsymbol{\sigma}}^{(t\bullet)}[\mathbf{S}_l]$, and the entire distribution by the product of these marginals. As mentioned earlier, however, to be feasible, the entire procedure ought to be carried out without explicitly computing the intermediate distributions.

In the case of DBNs, we can actually accomplish this update procedure quite efficiently. The method is based on the junction tree inference algorithm [LS88], which was described in detail in Section 3.4. Given $\tilde{\boldsymbol{\sigma}}^{(t-1)}$ represented as marginals over a set of disjoint clusters, and $R^{(t)}$, we wish to compute $\tilde{\boldsymbol{\sigma}}^{(t)}$ over the same set of clusters. From the 2-TBN, we first construct the structure of a junction tree with the requirement that every anterior cluster $S_l^{(t-1)}$ and every ulterior cluster $S_l^{(t)}$ be fully contained in at least one tree clique each. This is easily done during the construction of the junction tree by adding additional edges to the moral graph before it is triangulated (see Section 3.4); specifically, for each cluster of interest, we add an edge between each pair of variables it contains. We then incorporate the various conditional probability distributions from the 2-TBN into the appropriate tree potentials, as well as the marginals from $\tilde{\boldsymbol{\sigma}}^{(t-1)}$, and the evidence $R^{(t)}$. This is possible since the tree has been constructed so that the variables of each marginal from the approximate belief state are contained in full in at least one tree cluster. A standard junction tree calibration algorithm such as that of Section 3.4 can then be used to compute the posterior distribution over every clique. Once this is done, the various marginals $\tilde{\boldsymbol{\sigma}}^{(t)}[S_l^{(t)}]$ defining the new belief $\tilde{\boldsymbol{\sigma}}^{(t)}$ are easily extracted from the appropriate cliques.

Further savings can be obtained under the time-invariance assumption if the approximation scheme is held static. In this case, it is possible to compute the topology of the clique tree ahead of time, and initialize it once and for all with the numerical information contained in the 2-TBN. This results in a *proto junction tree* that can serve as a starting point for each propagation. The following algorithm summarizes the above computations, including this optimization.

Algorithm 4.5 Approximate DBN monitoring assuming independent marginals.

INPUTS:

◇ a 2-TBN in canonical form;

◇ a disjoint partition $\{\mathbf{S}_l\}$ of the canonical variables;

◇ an initial belief state $\tilde{\boldsymbol{\sigma}}^{(0)}$;

◇ a stream of observations $R^{(1)}, R^{(2)}, R^{(3)}, \ldots$.

OUTPUT:

▷ a stream of approximate belief states $\tilde{\boldsymbol{\sigma}}^{(1)}, \tilde{\boldsymbol{\sigma}}^{(2)}, \tilde{\boldsymbol{\sigma}}^{(3)}, \ldots$.

METHOD:

1. Construct a clique tree from the 2-TBN, requiring that every $\mathbf{S}_l^{(t-1)}$ and $\mathbf{S}_l^{(t)}$ be contained in full in at least one clique.
2. Initialize each clique potential to the constant function 1.
3. Incorporate the conditional probability distributions associated with the (ulterior) variables of the 2-TBN into the appropriate clique potentials. Let Υ_0 be the resulting (uncalibrated) proto junction tree.
4. For $t = 1, 2, 3, \ldots$:
   (a) Let Υ_t be a clone of Υ_0, assigning indices $t-1$ and t to the anterior and ulterior variables of Υ_t, respectively.
   (b) For each cluster $\mathbf{S}_l$, incorporate the marginal $\tilde{\boldsymbol{\sigma}}^{(t-1)}[S_l^{(t-1)}]$ in a suitable clique potential of Υ_t.
   (c) Incorporate the evidence $R^{(t)}$ in a suitable clique potential of Υ_t.
   (d) Calibrate the potentials in Υ_t using a standard junction tree propagation algorithm.
   (e) For each cluster $\mathbf{S}_l$, compute the posterior distribution $\Upsilon_t[S_l^{(t)}]$ by marginalizing a suitable clique from the calibrated Υ_t.
   (f) Represent $\tilde{\boldsymbol{\sigma}}^{(t)}$ as the (implicit) product of the extracted marginals $\Upsilon_t[\mathbf{S}_l^{(t)}]$.
   (g) Output $\tilde{\boldsymbol{\sigma}}^{(t)}$ and discard Υ_t.

4.2.2 Conditionally independent clusters

Dealing with the conditional independence assumption requires a few modifications to Algorithm 4.5, since the set of clusters $\{\mathbf{S}_l\}$ no longer forms a partition of the canonical variables, but a covering with overlaps. First, a refined definition of cluster is in order, based on the notion of cluster forest [JLO90] .

Definition 4.6 A *cluster forest* $\mathcal{F}$ over a set of BN or DBN nodes $\{A_1, ..., A_n\}$, is an undirected forest $\langle \mathbf{V}, \mathbf{E} \rangle$, whose vertices or *clusters* $\mathbf{V} = \{F_i\}$ are arbitrarily chosen non-empty subsets of the A_i's, and whose edges $\mathbf{E} = \{E_{i,j} : i < j\}$ are such that there is a path between distinct clusters F_i and F_j if the intersection $F_i \cap F_j$ is not empty. In addition, the cluster forest satisfies the *running intersection property*, whereby any vertex $A_k \in F_i \cap F_j$ is also contained in every cluster on the path between F_i and F_j.

This definition generalizes the previous notion of cluster as used in Algorithm 4.5, by allowing clusters to overlap. Cluster forests are also virtually identical to the notion of junction tree on which the inference procedure of Section 3.4 relies. In close connection with our previous use of junction trees, forests of overlapping clusters allow us to define a broader class of compactly representable distributions than in the disjoint case, as follows.

Definition 4.7 We say that a distribution $\boldsymbol{\varphi}$ is *representable* over $\mathcal{F}$ if it is represented as a set of *calibrated* marginals $\boldsymbol{\varphi}_i$ over the clusters F_i, *i.e.*, such that $\boldsymbol{\varphi}_i[F_i \cap F_j] = \boldsymbol{\varphi}_j[F_i \cap F_j]$ for any i, j. The distribution $\boldsymbol{\varphi}$ is defined as:

$$\boldsymbol{\varphi}[A_1, ..., A_n] \quad = \quad \frac{\prod_{F_i \in \mathbf{V}} \boldsymbol{\varphi}_i[F_i]}{\prod_{E_{i,j} \in \mathbf{E}} \boldsymbol{\varphi}_i[F_i \cap F_j]} \; .$$

It is now easy to extend Algorithm 4.5 to deal with conditionally independent clusters, under the assumption that the graph of these clusters forms a cluster forest. The full belief state at time t becomes, in the notation of the algorithm:

$$\boldsymbol{\sigma}^{(t)}[X_1^{(t)}, ..., X_n^{(t)}] \quad = \quad \frac{\prod_l \boldsymbol{\sigma}^{(t)}[S_l^{(t)}]}{\prod_{l_1 < l_2} \boldsymbol{\sigma}^{(t)}[S_{l_1}^{(t)} \cap S_{l_2}^{(t)}]} \; ,$$

which differs from the corresponding expression in the independent case by the introduction of the denominator. Hence, in order to maintain a useful belief state, we also need to obtain the marginals over all non-empty cluster intersections in the forest, and incorporate the inverse factors $(\boldsymbol{\sigma}^{(t)}[S_{l_1}^{(t)} \cap S_{l_2}^{(t)}])^{-1}$ in addition to the regular $(\boldsymbol{\sigma}^{(t)}[S_l^{(t)}])$ in the junction tree Υ before calibrating it. We note the deep similarity between dividing by the cluster intersection factors and the treatment of sepset potentials in the algorithm of Section 3.4.

The complete DBN filtering algorithm for overlapping clusters is as follows.

Algorithm 4.8 Approximate DBN monitoring assuming conditionally independent marginals.

INPUTS:

- ◇ a 2-TBN in canonical form;
- ◇ a forest of clusters $\{\mathbf{S}_l\}$ over the canonical variables;
- ◇ an initial belief state $\tilde{\boldsymbol{\sigma}}^{(0)}$;
- ◇ a stream of observations $R^{(1)}, R^{(2)}, R^{(3)}, \ldots$.

OUTPUT:

- ▷ a stream of approximate belief states $\tilde{\boldsymbol{\sigma}}^{(1)}, \tilde{\boldsymbol{\sigma}}^{(2)}, \tilde{\boldsymbol{\sigma}}^{(3)}, \ldots$.

METHOD:

1. Construct a clique tree from the 2-TBN, requiring that every $\mathbf{S}_l^{(t-1)}$ and $\mathbf{S}_l^{(t)}$ be contained in full in at least one clique.
2. Initialize each clique potential to the constant function 1.
3. Incorporate the conditional probability distributions associated with the (ulterior) variables of the 2-TBN into the appropriate clique potentials. Let Υ_0 be the resulting (uncalibrated) proto junction tree.
4. For $t = 1, 2, 3, \ldots$:
   (a) Let Υ_t be a clone of Υ_0, assigning indices $t-1$ and t to the anterior and ulterior variables of Υ_t, respectively.
   (b) For each cluster $\mathbf{S}_l$, incorporate the marginal $\tilde{\boldsymbol{\sigma}}^{(t-1)}[S_l^{(t-1)}]$ in a suitable clique potential of Υ_t.

(c) For each unordered pair of clusters $\{\mathbf{S}_l, \mathbf{S}_{l'}\}$ corresponding to an edge $\langle l, l' \rangle$ in the forest, marginalize $\tilde{\boldsymbol{\sigma}}^{(t-1)}[S_l^{(t-1)}]$ over the variables in $\mathbf{S}_l \cap \mathbf{S}_{l'}$, take the inverse of each value in the result, and incorporate the final result in a suitable clique potential of Υ_t.

(d) Incorporate the evidence $R^{(t)}$ in a suitable clique potential of Υ_t.

(e) Calibrate the potentials in Υ_t using a standard junction tree propagation algorithm.

(f) For each cluster $\mathbf{S}_l$, compute the posterior distribution $\Upsilon_t[S_l^{(t)}]$ by marginalizing a suitable clique from the calibrated Υ_t.

(g) Represent $\tilde{\boldsymbol{\sigma}}^{(t)}$ as the set of the extracted marginals $\Upsilon_t[\mathbf{S}_l^{(t)}]$ (using the semantics of Definition 4.7).

(h) Output $\tilde{\boldsymbol{\sigma}}^{(t)}$ and discard Υ_t.

The following theorem characterizes the approximate belief states produced by the above algorithms.

Theorem 4.9 *At each time step t, among the distributions in the representation class of Algorithm 4.5 (resp. Algorithm 4.8), the approximate belief state $\tilde{\boldsymbol{\sigma}}^{(t)}$ computed by that algorithm from the previous belief $\tilde{\boldsymbol{\sigma}}^{(t-1)}$, is the one with the smallest KL divergence to the distribution that one would obtain by performing one step of exact inference from the same starting belief $\tilde{\boldsymbol{\sigma}}^{(t-1)}$ and evidence $R^{(t)}$.*

Proof The case of Algorithm 4.5 follows immediately by observing that the KL divergence decomposes additively with respect to the non-overlapping clusters, and that projecting a distribution onto a cluster by marginalization yields the closest approximation representable over that cluster in terms of KL divergence.

The case of Algorithm 4.8 is similar to result due to Chow and Liu [CL68], based on the notion of *tree distribution*. We say that a distribution $\boldsymbol{\tau}$ over a set of variables $\mathbf{V} = \{V_1, ..., V_n\}$ is conformal with a forest $\langle \mathbf{V}, \mathbf{E} \rangle$ if it can be factored as:

$$\boldsymbol{\tau}[\mathbf{V}] \quad = \quad \frac{\prod_{\langle V_i, V_j \rangle \in \mathbf{E}} \boldsymbol{\tau}[V_i, V_j]}{\prod_{V_k \in \mathbf{V}} \boldsymbol{\tau}[V_k]^{\deg V_k - 1}} \;, \tag{4.1}$$

where $\boldsymbol{\tau}[V_i, V_j]$ denotes the distribution $\boldsymbol{\tau}$ marginalized over V_i and V_j.

Using techniques similar to [CL68], it is easy to show that, if a distribution $\boldsymbol{\tau}$ is conformal with a forest $\langle \mathbf{V}, \mathbf{E} \rangle$, and has the same marginals over all elements of $\mathbf{V}$ as some other distribution $\boldsymbol{\pi}$, then, among all the distributions that conform with $\langle \mathbf{V}, \mathbf{E} \rangle$, $\boldsymbol{\tau}$ is the best approximation of $\boldsymbol{\pi}$ in the sense of KL divergence. In other words, $\boldsymbol{\tau}$ satisfies:

$$\boldsymbol{\tau} \quad = \quad argmin_{\boldsymbol{\tau}' \in \Xi[\langle \mathbf{V}, \mathbf{E} \rangle]} \boldsymbol{D}_{KL}[\pi \,||\, \boldsymbol{\tau}'] \ ,$$

where $\Xi[\langle \mathbf{V}, \mathbf{E} \rangle]$ is the class of distributions representable over $\langle \mathbf{V}, \mathbf{E} \rangle$.

We now describe how a belief state $\tilde{\boldsymbol{\sigma}}^{(t)}$ as produced by Algorithm 4.8 is mapped to a tree-conforming distribution. The forest $\langle \mathbf{V}, \mathbf{E} \rangle$ is constructed as follows. For each cluster in $\tilde{\boldsymbol{\sigma}}^{(t)}$, we create one node $V_i \in \mathbf{V}$. For each sepset in $\tilde{\boldsymbol{\sigma}}^{(t)}$ that links two overlapping clusters represented as V_i and V_j, we create one node $V_k \in \mathbf{V}$ and two edges $\langle V_i, V_k \rangle, \langle V_j, V_k \rangle \in \mathbf{E}$. The domains of the V_i's and their marginal probabilities $\boldsymbol{\tau}[V_i]$ are defined the obvious way from $\tilde{\boldsymbol{\sigma}}^{(t)}$; the entire distribution $\boldsymbol{\tau}$ is then defined as in Equation 4.1.

It is easy to see that $\boldsymbol{\tau}$ represents the same distribution as $\tilde{\boldsymbol{\sigma}}^{(t)}$. To show this, we observe that the numerator Equation 4.1 contains one factor $\boldsymbol{\tau}[V_i, V_j]$ for each adjacent cluster-sepset pair in $\tilde{\boldsymbol{\sigma}}^{(t)}$. Since, by definition each cluster encompasses its adjacent sepsets, the factors over these pairs all reduce to factors over the corresponding clusters. Hence, the numerator of Equation 4.1 reduces to a product of cluster potentials from $\tilde{\boldsymbol{\sigma}}^{(t)}$, where each potential is counted with a multiplicity equal to the number of sepsets adjacent to that cluster. As for the denominator, by construction of τ, it decomposes in a product of sepset factors and cluster factors. It is easy to see that the sepset factors all have multiplicity one (since the degree of a sepset is 2), whereas the cluster factors appear with the same multiplicity as in the numerator, minus one. In summary, the distribution $\boldsymbol{\tau}$ can be rewritten as:

$$\boldsymbol{\tau} \quad = \quad \frac{\prod_{\text{clusters}} \tilde{\boldsymbol{\sigma}}^{(t)}[\text{cluster}]}{\prod_{\text{sepsets}} \tilde{\boldsymbol{\sigma}}^{(t)}[\text{sepset}]} \ ,$$

which is our representation of $\tilde{\boldsymbol{\sigma}}^{(t)}$.

Thus, since $\boldsymbol{\tau}$ represents the same distribution as $\tilde{\boldsymbol{\sigma}}^{(t)}$, it also shares with $\tilde{\boldsymbol{\sigma}}^{(t)}$ the same marginals as the calibrated clique tree Υ_t in Algorithm 4.8. We deduce from Equation 4.2 that $\boldsymbol{\tau}$ is the best approximation of Υ_t with that respective structure. Since any distribution $\tilde{\boldsymbol{\sigma}}_*^{(t)}$ with the same structure as $\tilde{\boldsymbol{\sigma}}^{(t)}$ can be similarly mapped to a distribution $\boldsymbol{\tau}' \in \Xi[\langle \mathbf{V}, \mathbf{E} \rangle]$, we conclude that $\tilde{\boldsymbol{\sigma}}^{(t)}$ is also the best approximation of Υ_t for that structure. Since Υ_t is an exact representation of the distribution of interest, the claim follows. ◻

4.3 Experimental results

We have validated the above algorithm in the context of two structured dynamic Bayesian networks constructed for real life applications. The first model—the BAT network—is used for monitoring freeway traffic [FHKR95]; the second model—the WATER network—is used for monitoring the biological processes of a water purification plant [JKOP89].

Both models are sketched in Figures 4.2 and 4.3 as they are used in our experiments. Hidden and observed variables are respectively drawn as clear and shaded nodes.

As the original WATER network did not have any distinguished response variables, we added four of them to obtain the network represented in Figure 4.3. Each of the added variables has the same domain as and replicates the immediate value of one of the original state variables, with added noise. Specifically, for a variable X taking m possible values, the conditional probability table for $X' \to noisyX'$ is arbitrarily set (given that $2 \leq m \ll 10$ for the variables of interest in this particular model) to:

$$\mathbb{P}[noisyX'|X'] = \begin{cases} 1 - \frac{m-1}{10} & \text{when } X' = noisyX' \\ \frac{1}{10} & \text{otherwise} \end{cases} .$$

This modification allows us to treat the WATER network as a partially observable model with clearly identified response variables, which is more conducive to interesting

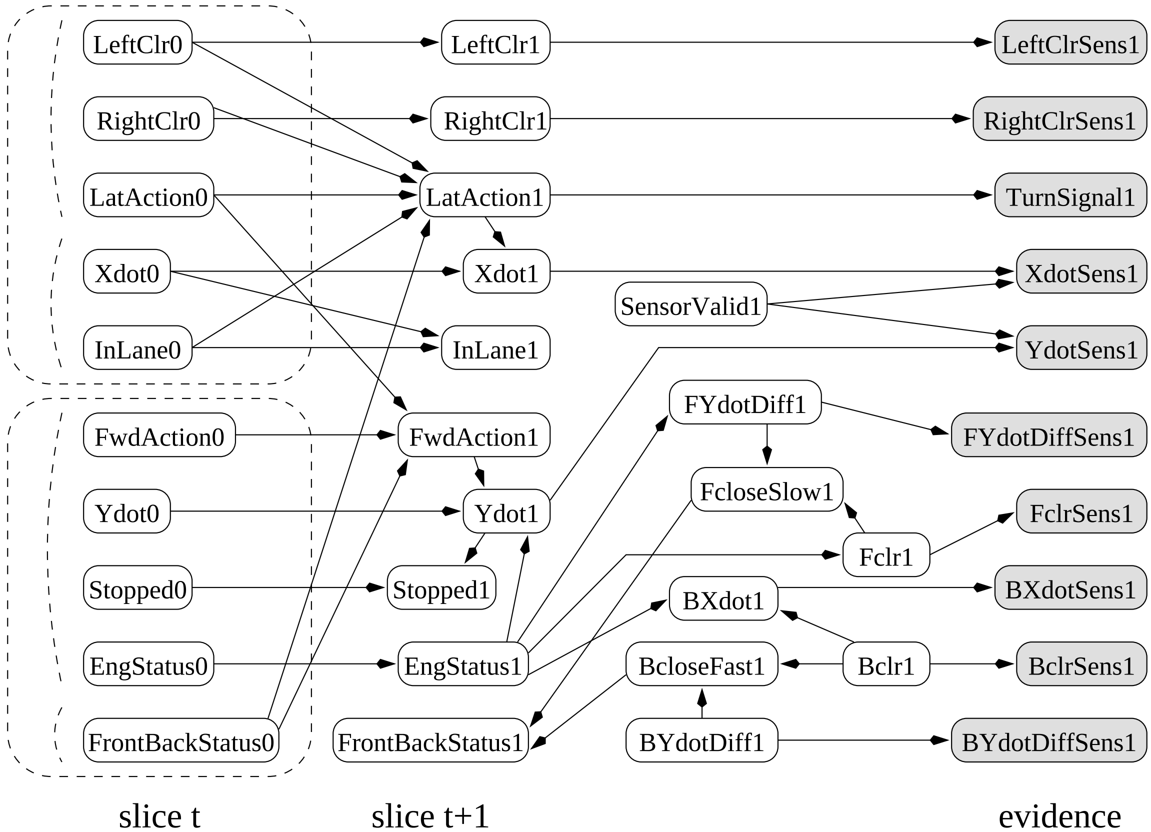


Figure 4.2: Structure of the BAT network, represented in canonical 2-TBN form. The anterior and ulterior hidden state variables are gathered in the first and second column, respectively. The fourth column contains the observed variables in the ulterior time slice. The third column contains the remaining ulterior variables, which are hidden but not part of the canonical state. The dotted boxes and arcs respectively delineate the two-cluster and four-cluster approximations as used in the experiments.

experiments. The BAT model is used without any alteration.

The experimental methodology for each model is as follows. In the preparation phase, a number of trajectories of simulated responses are generated by random sampling of the given model; this phase was made necessary due to the lack of published actual data for either network—and also allowed us to test our algorithm in isolation, by removing the possibility of a mismatch between model and data. The initial belief

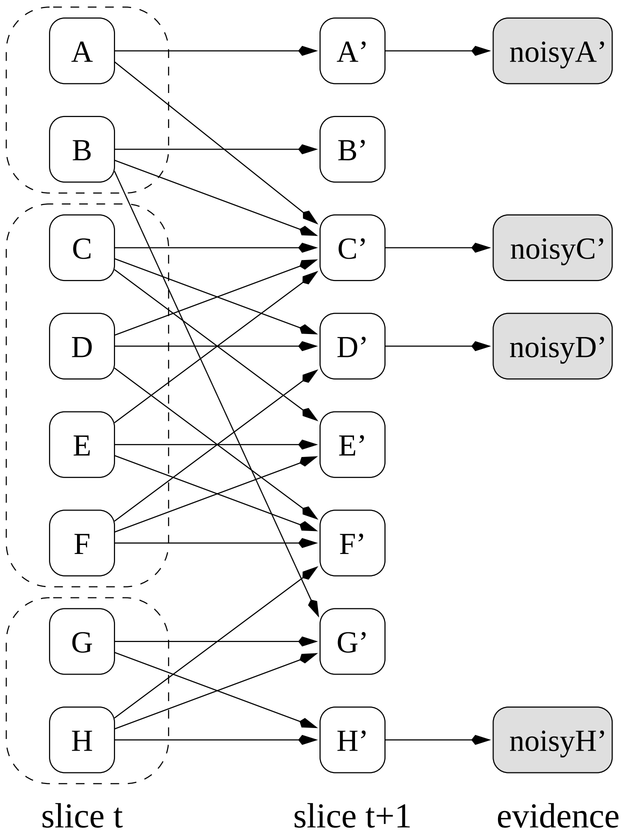


Figure 4.3: Structure of the WATER network with added observation variables, in canonical 2-TBN form. The dotted boxes represent one of the approximation clusterings used in the experiments.

state in the sampled data is chosen to be the uniform distribution, for it is fully decomposable and assumes no knowledge of the initial process state. In the test phase, for each trajectory, the evolution of the process is monitored time slice after time slice, using both exact inference and our approximation algorithm, feeding on the stream of sampled responses to supply the needed observations. At every time slice, the approximate belief state is compared to the true one, using several discrepancy measures, such as relative entropy (or KL divergence) and Manhattan distance (or distance in $\mathcal{L}_1$). For simplicity, and to facilitate speed comparisons, exact inference is emulated by using the approximation algorithm with the trivial partition which places all canonical variables in the same cluster (it is easy to see that this results in an exact computation; moreover, the computational overhead for using the approximate algorithm to perform exact inference is truly negligible). Unless specified otherwise, the approximated runs use the non-overlapping version of the algorithm, *i.e.*, Algorithm 4.5.

Figure 4.4 shows the evolution of the relative entropy error, for the BAT network, on a typical 1000-point trajectory independently sampled from the same network. In this network, the canonical belief state is determined by 10 variables partitioned in two weakly interacting groups of 5, as can be observed from Figure 4.2: this is the approximation scheme that was used to generate the plot of Figure 4.4. On a 360 MHz workstation, approximate monitoring with the above partition took about 0.11 second per time slice, as compared to 1.72 for exact inference, yielding a 15-fold speedup. In terms of accuracy, the relative entropy averages at 0.0007 in base e, or one thousandth of one bit as expressed in base 2, and remains very low most of the time, with, as would be expected, infrequent spikes to somewhat larger values, peaking at 0.065. We also note that the error does not appear to grow over time. Since, for traditional reasons, the $\mathcal{L}_1$ error over individual variables is also regarded as a relevant benchmark, we also computed this error for the marginalized beliefs over two selected variables—'LateralAction' and 'ForwardAction', whose meaning can be found in [FHKR95]. Their $\mathcal{L}_1$ error respectively averaged 0.00013 and 0.0019, and remained bounded by 0.02 and 0.07 over the 1000-step run of our experiment, featuring a qualitative pattern similar to that of relative entropy.

Similar experiments were conducted on the WATER network with the aforementioned amendment, as drawn in Figure 4.3, using the three-cluster partition (A-B) (C-D-E-F) (G-H) as shown on the figure. Over a typical run of length 3000, the two largest relative entropy values reached 0.14 and 0.06, while the average hovered at 0.006 for that run. Running times were 0.19 second per time slice under the above approximation scheme, versus 6.02 seconds per slice for the exact algorithm, or a 31-fold speedup.

To investigate the influence of the chosen partition for the approximate belief state representation, we compared three different approximation clusterings on the

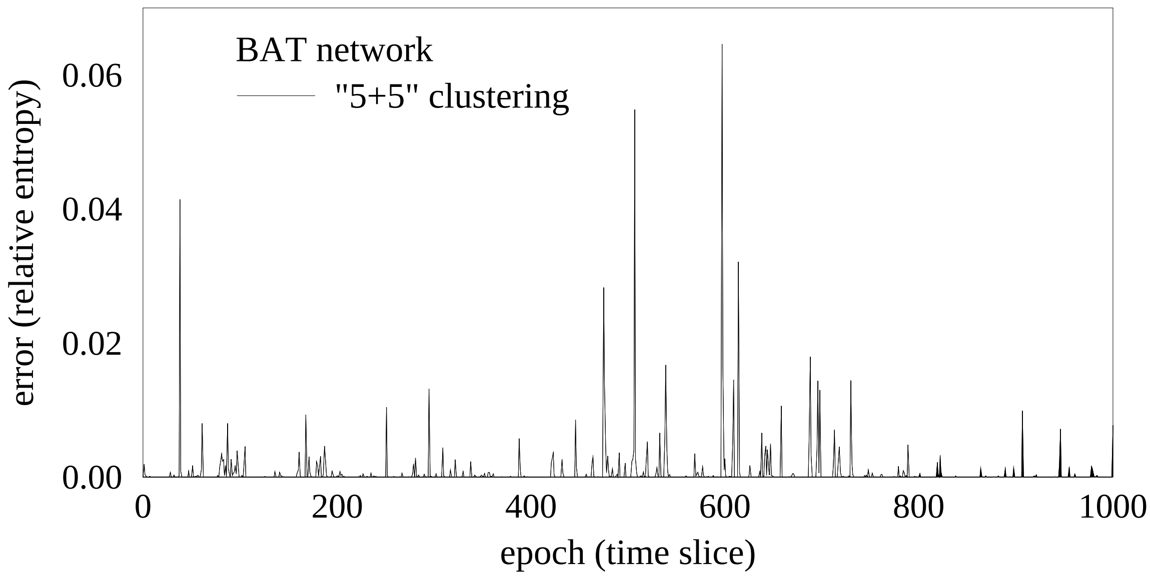


Figure 4.4: Experimentally observed relative entropy error for a typical run using the BAT network.

BAT network, in addition to the exact case. Figure 4.5 shows, on a logarithmic scale, the evolution of the relative entropy error for all three approximations. In order to improve their legibility, these curves were produced by averaging eight independent runs, using the same eight distinct trajectories for all three approximations. The lower curve corresponds to the '5+5' clustering used in the previous experiment, which now achieves an average error of 0.0006, for a 15-fold computational speedup with respect to exact inference. The medium curve corresponds to the four-cluster '3+2+4+1' partition shown in Figure 4.2, which is obtained by further splitting the clusters of the previous partition; this approximation achieves an average error of 0.015 and a 20-fold computational speedup, which demonstrates the increased efficiency and decreased accuracy of the more aggressive partition. The upper curve corresponds to a three-cluster '3+3+4' partition, whose clusters bear no relationship to the connectivity of the variables in the network; in this case, the average error jumps to 0.13, for a comparable 20-fold speedup. This case clearly illustrates the importance of devising approximations that respect the structure of the process.

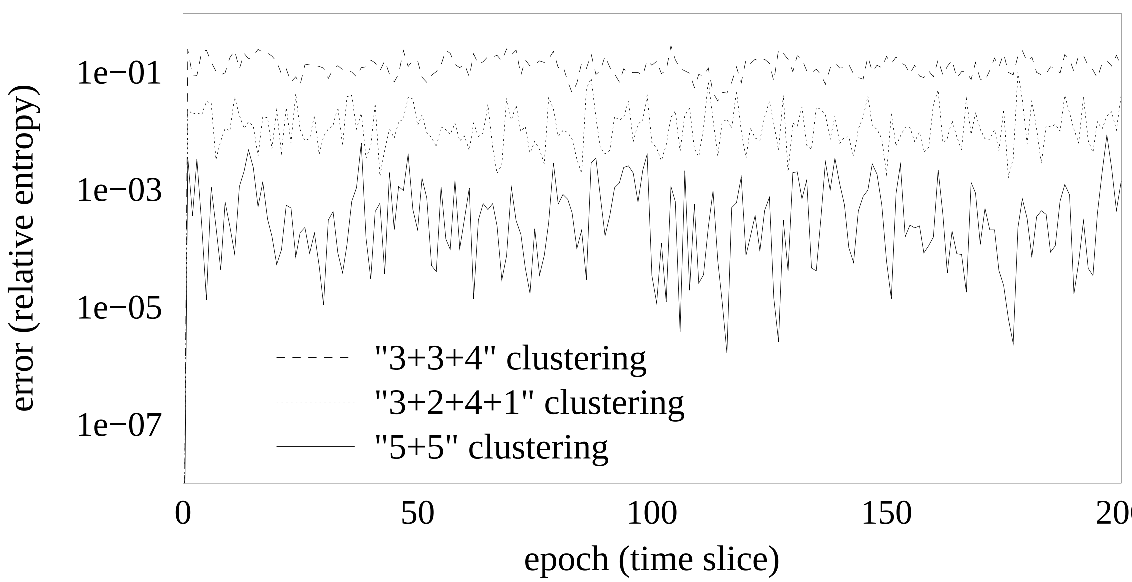


Figure 4.5: Experimental comparison of relative entropy error for three different approximate belief state representations (BAT network, results averaged over 8 runs).

The accuracy is further improved by using Algorithm 4.8 with a conditionally independent approximation scheme, *i.e.*, with overlapping clusters arranged as a cluster forest. Tests were conducted on the WATER network using a belief state decomposed into the three overlapping clusters (A-B-C-D-E) (C-D-E-F-G) (G-H). Using this approximation, we obtain, for the same sequence of observations as above, an average error of just 0.0015. Also, the error now remains bounded by 0.018 throughout, reducing the maximum error by a factor of 8. Approximate inference here took 0.47 second per slice, corresponding to a 13-fold speedup over exact inference.

We also briefly validated our approach in the case where there is no evidence. Clearly, in this case, the instantaneous belief state converges quickly to the stationary distribution of the process. The approximate distribution converges as well, to a stationary distribution that can be represented under the approximation scheme. This distribution will typically not be the best representable approximation of the exact stationary distribution, although the foregoing discussion suggests that the error will be small. To verify this, we tested the effect of removing all evidence in our filtering experiments. Without evidence, the error curve is completely smooth, as it is not constantly pushed up and down at every step by the incoming data. It also converges quickly to a constant error, corresponding to the error between the correct stationary distribution of the process and our limiting approximation to it. Interestingly, the error curve with evidence features a lower average than without, indicating that the evidence is not only harmless to the average tracking error, but is in fact beneficial as it further contributes to its reduction. This agrees with the intuition that conditioning two distributions on the same evidence "should" bring them closer to each other.

Chapter 5

Contraction analysis

As mentioned in the introduction, the ability to reason efficiently and accurately with complex, structured models of dynamic systems is fundamental to many applications. Of particular importance is the task of filtering, or monitoring the state of a system given observations. This is an important task both in its own right, and as it forms the basis of many other tasks of practical importance. Although long solved and routinely applied in many applications that rely on traditional models such as Kalman filters and hidden Markov models, the filtering problem is intractable when applied to more expressive models, such as DBNs, where the total state space may grow exponentially with the size of the representation.

The purpose of this chapter is to study the fundamental intractability of inference in complex dynamic systems, and to investigate ways to go around it. In particular, the focus is on the study of some theoretical properties of a stochastic system that make it amenable to a certain class of approximation whereunder inexact but accurate and efficient inference becomes feasible.

5.1 Contraction and distances

Let $\boldsymbol{\sigma}^{(t\bullet)}$ be the belief state of the system at the current time t, given all observations obtained at time t and before. Under the Markov assumption, this belief state captures all available knowledge about the state of the system that is pertinent to its

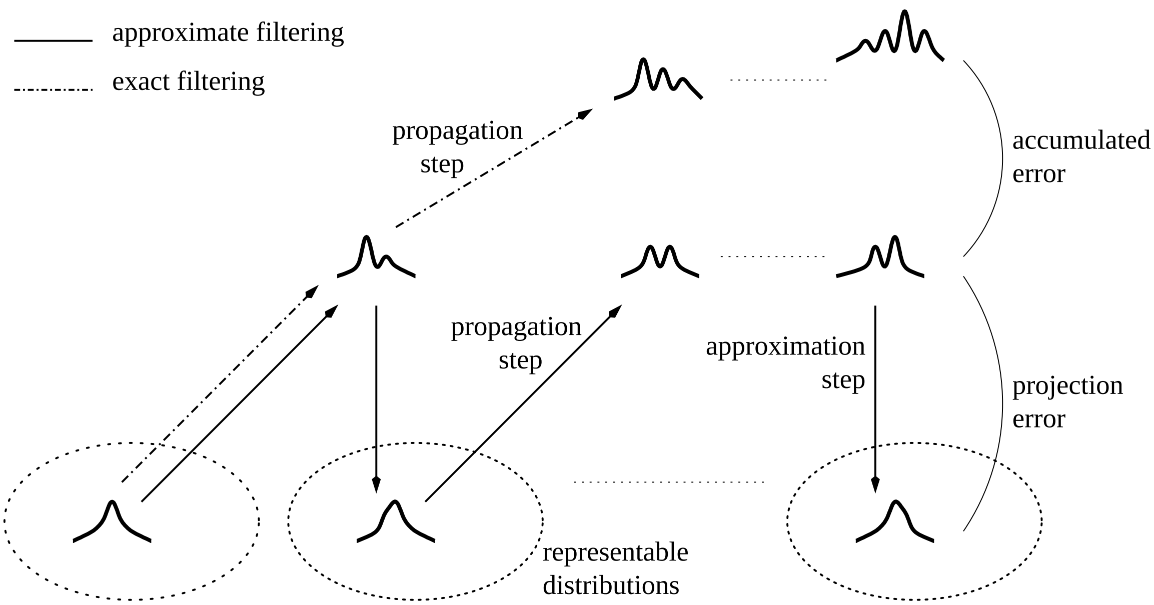


Figure 5.1: Evolution of the error between approximate and exact inference. The total error is composed of the accumulated error from the previous time steps, and the momentary projection error resulting from the latest approximation step.

future evolution. This is also the belief state that one would obtain using the general filtering procedure described in Section 3.2. For the reasons detailed in Section 4.1, $\boldsymbol{\sigma}^{(t\bullet)}$ is not feasibly computable for reasonably complex DBN models. Therefore, let $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ be the *estimated* belief state as obtained by some approximate inference procedure, such as one of the algorithms of Section 4.2, under the same observations.

Consider applying the general filtering procedure of Section 4.2 on each of $\boldsymbol{\sigma}^{(t\bullet)}$ and $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$, for one additional step. Intuitively, as each of them is propagated through the transition model $\mathcal{S}$, some of the information about the state at time t is forgotten; and, as the two belief states forget about their differences, they become closer to each other.

Naturally, further approximations on $\hat{\boldsymbol{\sigma}}$ will again pull the exact and estimated belief states away from each other, as illustrated in Figure 5.1. As will be shown in Section 5.4, in order for the total approximation error to remain bounded, the effects of each individual approximation error need to dampen exponentially quickly. That is, propagation through the transition model $\mathcal{S}$ should reduce the distance between two belief states $\boldsymbol{\sigma}^{(t\bullet)}$ and $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ by a constant factor. More generally, such a *contraction property* should hold true for the entire filtering step, where the propagation through the transition model $\mathcal{S}$ is followed by the conditioning on evidence according to the observation model $\mathcal{R}$.

The first half of this result is easily shown for various norms in $\mathcal{L}_p$, such as the Manhattan distance in $\mathcal{L}_1$, and, under certain restrictive conditions, the Euclidean distance in $\mathcal{L}_2$. Namely, if such an $\mathcal{L}_p$ norm is used to evaluate the error between the exact and estimated state distributions, there exists a constant β_p depending on the transition model $\mathcal{S}$, such that:

$$\| \mathcal{S}[\boldsymbol{\sigma}^{(t\bullet)}] - \mathcal{S}[\hat{\boldsymbol{\sigma}}^{(t\bullet)}] \|_p \quad \leq \quad \beta_p \; \| \boldsymbol{\sigma}^{(t\bullet)} - \hat{\boldsymbol{\sigma}}^{(t\bullet)} \|_p \quad .$$

Unfortunately, all $\mathcal{L}_p$ norms for $p \geq 1$ are inappropriate with respect to the conditioning step. The desirable contraction featured with respect to the transition model is ruined by their behavior during the conditioning step. In general, if $R^{(t)} = r_k$ is the observation at time t, the distance $\| \mathcal{R}_{r_k}[\boldsymbol{\sigma}^{(\bullet t)}] - \mathcal{R}_{r_k}[\hat{\boldsymbol{\sigma}}^{(\bullet t)}] \|_p$ can be larger than $\| \boldsymbol{\sigma}^{(\bullet t)} - \hat{\boldsymbol{\sigma}}_p^{(\bullet t)} \|$ by an arbitrary factor. This gap would be reasonable if the distance would only increase under unlikely evidence, and still decrease on expectation. Unfortunately, not only are there situations where the distance increases on expectation, but there are cases where the distance increases under all possible observations, as illustrated below.

Example 5.1 Consider a simple process with state space $\mathbf{S} = \{s_0, s_1, s_2, s_3\}$, and possible responses $\mathbf{R} = \{r_1, r_2, r_3\}$. The observation model $\mathcal{R}$ specifies that, when the system is in state s_0, either of r_1, r_2, or r_3 is randomly observed with equal probability $\frac{1}{3}$. When the system is in s_i with $i \neq 0$, then r_i is deterministically observed. Let the estimated state distribution at time t prior to observation be the uniform distribution, *i.e.*, $\hat{\boldsymbol{\sigma}}^{(\bullet t)} = [\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}]$, and assume that the true distribution places probability one on s_0, *i.e.*, $\boldsymbol{\sigma}^{(\bullet t)} = [1, 0, 0, 0]$. It follows from the true distribution that the response at time t will be r_1, r_2, or r_3 with equal probability. Without loss of generality, suppose that r_1 is the actual observation. Conditioning on r_1, the true and estimated state distributions are straightforwardly computed per the rules of Section 3.2. One obtains, for the true distribution, $\boldsymbol{\sigma}^{(t\bullet)} = [1, 0, 0, 0]$, and for the the estimated distribution, $\hat{\boldsymbol{\sigma}}^{(\bullet t)} = [\frac{1}{4}, \frac{3}{4}, 0, 0]$.

It is easy to observe that, for $p > 1$ and $p = \infty$, the $\mathcal{L}_p$ distance between the true and estimated posterior distributions is strictly greater than between the corresponding priors. By symmetry of argument, this conclusion holds uniformly for the three possible observations r_1, r_2, r_3.

∎

A slightly more involved scenario can be constructed to show that conditioning may also increase the $\mathcal{L}_1$ distance on expectation, if not uniformly over all observations. The following example illustrates this situation.

Example 5.2 Let the state space $\mathbf{S} = \{s_1, s_2, s_3\}$, and the response set $\mathbf{R} = \{r_1, r_2\}$. Let the observation model:

$$\mathcal{R} \;=\; \boldsymbol{P}[R|S] \;=\; \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ 1 & 0 \\ 0 & 1 \end{pmatrix} .$$

Let the true prior distribution $\boldsymbol{\sigma}^{(\bullet t)} = [\frac{1}{2}, 0, \frac{1}{2}]$, and the estimated prior $\hat{\boldsymbol{\sigma}}^{(\bullet t)} = [\frac{4}{5}, \frac{1}{5}, 0]$, so that their initial $\mathcal{L}_1$ distance is 1. It follows from $\boldsymbol{\sigma}^{(\bullet t)}$ and $\mathcal{R}$ that $\boldsymbol{P}[R^{(t)}] = [\frac{1}{4}, \frac{3}{4}]$. In the event that $R^{(t)} = r_1$, the true and estimated posteriors are, respectively, $\boldsymbol{\sigma}^{(t\bullet)} = [1, 0, 0]$ and $\hat{\boldsymbol{\sigma}}^{(t\bullet)} = [\frac{2}{3}, \frac{1}{3}, 0]$, for a distance of $\frac{2}{3}$. Similarly, in the event that $R^{(t)} = r_2$, the true and estimated posteriors are, respectively, $\boldsymbol{\sigma}^{(t\bullet)} = [\frac{1}{3}, 0, \frac{2}{3}]$, and, $\hat{\boldsymbol{\sigma}}^{(t\bullet)} = [1, 0, 0]$, for a distance of $\frac{4}{3}$.

Since the true probabilities of r_1 and r_2 are respectively $\frac{1}{4}$ and $\frac{3}{4}$, it follows that the expected $\mathcal{L}_1$ distance between the posteriors is $\frac{1}{4}\frac{2}{3} + \frac{3}{4}\frac{4}{3} = \frac{7}{6}$, which is greater than the initial distance of 1.

∎

The above scenarios eloquently illustrate that the $\mathcal{L}_p$ distance between a pair of distributions may increase under expectation, or even with probability one, when conditioned on the same observation. This phenomenon is very counter-intuitive: indeed, even if an unlikely observation may actually pull the distributions apart, one would expect that, on average, the distance should be reduced. It turns out that the above phenomenon is due to the nature of the $\mathcal{L}_p$ norms, rather than being a fundamental property of the conditioning itself.

A more useful measure of the error between a target distribution $\boldsymbol{\psi}$ and a reference distribution $\boldsymbol{\varphi}$, is the discrete *relative entropy*, as defined in Section 2.2. Contrarily to $\mathcal{L}_p$ norms, relative entropy is not a distance metric; rather, it quantifies the cost, in an information theoretic sense, of using the distribution $\boldsymbol{\psi}$ when the true distribution is $\boldsymbol{\varphi}$. Relative entropy is, for a variety of reasons detailed in [CT91, chap. 2], an excellent measure of discrepancy between a reference distribution and an approximation to it. In particular, it is closely related to the theoretical penalty incurred by a rational agent that acts optimally using an estimated distribution $\boldsymbol{\psi}$ when the true distribution is $\boldsymbol{\varphi}$. Furthermore, and in contrast to $\mathcal{L}_p$ norms, relative entropy behaves very reasonably with respect to conditioning:

Fact 5.3 *For any observation model $\mathcal{R}$ as previously defined, and for any $\boldsymbol{\sigma}^{(\bullet t)}$, $\hat{\boldsymbol{\sigma}}^{(\bullet t)}$,*

$$\boldsymbol{E}_{\boldsymbol{\rho}^{(t)}}[\mathbb{D}_{KL}[\mathcal{R}_{R^{(t)}}[\boldsymbol{\sigma}^{(\bullet t)}] \,\|\, \mathcal{R}_{R^{(t)}}[\hat{\boldsymbol{\sigma}}^{(\bullet t)}]]] \quad \leq \quad \mathbb{D}_{KL}[\boldsymbol{\sigma}^{(\bullet t)} \,\|\, \hat{\boldsymbol{\sigma}}^{(\bullet t)}] \ ,$$

where $\boldsymbol{\rho}^{(t)} = (\boldsymbol{\sigma}^{(\bullet t)}\,\mathcal{R})$ is the prior distribution for the response at time t just before it occurs.

For all of the above reasons, relative entropy makes a very natural choice for the purpose of quantifying approximation errors in probabilistic inference.

Unfortunately, the problem seems to have simply shifted from one place to another. While relative entropy is better behaved with respect to the conditioning phase, the situation is not as clear with respect to the transition step. The literature contains a large number of asymptotic contraction results applicable to stochastic transitions, as outlined in [CT91, chap. 3]. Relative entropy never increases through a stochastic process, *i.e.*, $\mathbb{D}_{KL}[\mathcal{S}[\boldsymbol{\sigma}^{(t\bullet)}] \,\|\, \mathcal{S}[\hat{\boldsymbol{\sigma}}^{(t\bullet)}]] \leq \mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \hat{\boldsymbol{\sigma}}^{(t\bullet)}]$, and it ultimately tends to zero for a very broad class of processes, *i.e.*, $\mathbb{D}_{KL}[\mathcal{S}^k[\boldsymbol{\sigma}^{(t\bullet)}] \,\|\, \mathcal{S}^k[\hat{\boldsymbol{\sigma}}^{(t\bullet)}]] \to 0$ as $k \to \infty$

when $\mathcal{S}$ is *ergodic* [CT91, page 34]. However, stronger results are needed to bound the accumulation of approximation errors over time.

The next section presents a proof that stochastic transitions contract relative entropy at a geometric rate, in the case of unstructured processes, such as hidden Markov models. The analysis is later strengthened in various ways, to exploit the structure present in more expressive representations, such as dynamic Bayesian networks.

5.2 Elementary contraction

We now show that a stochastic process $\mathcal{S}$ does, indeed, lead to a contraction in relative entropy.

Before proceeding, we note that a similar result to the central theorem of this section had been proven a few years earlier, unbeknownst to us, by a group of researchers using a rather different approach [AL95]. However, an essential feature of our own proof technique is that it generalizes readily to the more interesting case of structured processes, as we will soon see, starting in Section 5.3.

It will be useful later on to consider a somewhat more general setting, where the sets of states before and after the stochastic transition are not necessarily the same. Thus, let $\Omega^{\triangleleft} = \{\omega_1^{\triangleleft}, ..., \omega_{n^{\triangleleft}}^{\triangleleft}\}$ be the *anterior* state space, and $\Omega^{\triangleright} = \{\omega_1^{\triangleright}, ..., \omega_{n^{\triangleright}}^{\triangleright}\}$ be the *ulterior* state space. Let $\mathcal{Q}$ be an $n^{\triangleleft} \times n^{\triangleright}$ stochastic matrix, representing an arbitrary random transition from $\Omega^{\triangleleft}$ to $\Omega^{\triangleright}$. Let $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ be two arbitrary distributions over the anterior state space $\Omega^{\triangleleft}$, and let $\boldsymbol{\varphi}^{\triangleright}$ and $\boldsymbol{\psi}^{\triangleright}$ be the corresponding ulterior distributions induced over $\Omega^{\triangleright}$ by one application of the stochastic transformation $\mathcal{Q}$.

The objective of this analysis is to determine the minimal extent to which the stochastic transition $\mathcal{Q}$ coerces the two anterior distributions $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ toward the same image. Clearly, if $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ start out the same, their images will also be the same. In the worst case, $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ are as different as they can be, *i.e.*, they have no probability mass in common. For example, one distribution $\boldsymbol{\varphi}^{\triangleleft}$ might assign probability one to some state $\omega_{i_1}^{\triangleleft}$, while all of the mass in the other distribution $\boldsymbol{\psi}^{\triangleleft}$ is on some other state $\omega_{i_2}^{\triangleleft}$. However, even in those circumstances, the stochastic nature of the transition $\mathcal{Q}$ will typically cause each of $\boldsymbol{\varphi}^{\triangleright}$ and $\boldsymbol{\psi}^{\triangleright}$ to place some weight on any posterior state $\omega^{\triangleright}_j$: specifically, the probability $\varphi^{\triangleright}[\omega^{\triangleright}_j]$ is equal to $\mathcal{Q}[\omega^{\triangleright}_j|\omega^{\triangleleft}_{i_1}] = \mathcal{Q}_{i_1,j}$, while $\psi^{\triangleright}[\omega^{\triangleright}_j]$ is given by $\mathcal{Q}[\omega^{\triangleright}_j|\omega^{\triangleleft}_{i_2}] = \mathcal{Q}_{i_2,j}$. Thus, even though none of the probability mass of $\varphi^{\triangleleft}$ and $\psi^{\triangleleft}$ was in agreement, $\varphi^{\triangleright}$ and $\psi^{\triangleright}$ agree on $\omega^{\triangleright}_j$ for a mass of $min\{\mathcal{Q}[\omega^{\triangleright}_j|\omega^{\triangleleft}_{i_1}], \mathcal{Q}[\omega^{\triangleright}_j|\omega^{\triangleleft}_{i_2}]\}$, independently of the starting distributions $\varphi^{\triangleleft}$ and $\psi^{\triangleleft}$. By repeating this reasoning and summing over all the elements of $\Omega^{\triangleright}$, one would obtain the minimum fraction of the mass by which any two distributions would be forced to agree, following one application of the stochastic transformation $\mathcal{Q}$.

Based on this insight, we define the following natural characterization of the mixing properties of a discrete stochastic transition:

Definition 5.4 For a discrete Markov stochastic transition $\mathcal{Q}$, the *pairwise mixing coefficient* of $\mathcal{Q}$ is:

$$\gamma_{\mathcal{Q}} \quad = \quad min_{i_1,i_2} \sum_{j=1}^{n^{\triangleright}} min\{\mathcal{Q}[\omega^{\triangleright}_j|\omega^{\triangleleft}_{i_1}], \mathcal{Q}[\omega^{\triangleright}_j|\omega^{\triangleleft}_{i_2}]\} \ .$$

The pairwise mixing coefficient of any stochastic transition lies in the interval between 0 and 1. A mixing coefficient of 0 indicates that the transition is not mixing for at least one pair of states; in other words, that there exists a pair of anterior states that can be reliably distinguished from each other, given the induced ulterior distribution. By contrast, a mixing coefficient of 1 characterizes completely mixing transformations, which produce the same ulterior distribution regardless of the anterior distribution.

Shortly, our first theorem will show the fundamental relationship between the mixing coefficient $\gamma_{\mathcal{Q}}$ and the reduction of relative entropy through stochastic propagation by $\mathcal{Q}$. We start by proving the following lemma, which provides a *contraction decomposition* of $\mathcal{Q}$, which is useful for isolating the probability mass in the two distributions that is guaranteed to mix.

Intuitively, the lemma says that, for any pair of anterior distributions $\varphi^{\triangleleft}$ and $\psi^{\triangleleft}$, the transition matrix $\mathcal{Q}$ can be additively decomposed into two matrices $\mathcal{Q}^{\Gamma}$ and $\mathcal{Q}^{\Delta}$, such that the outcomes of transitioning $\varphi^{\triangleleft}$ and $\psi^{\triangleleft}$ by $\mathcal{Q}^{\Gamma}$ are indistinguishable. The additive decomposition is later used to show that, as far as $\varphi^{\triangleleft}$ and $\psi^{\triangleleft}$ are concerned, the process defined by $\mathcal{Q}$ is equivalent to a coin flip (biased by the respective total masses of $\mathcal{Q}^{\Gamma}$ and $\mathcal{Q}^{\Delta}$) followed by a transition through either $\mathcal{Q}^{\Gamma}$ or $\mathcal{Q}^{\Delta}$, depending on the outcome of the coin flip (and where $\mathcal{Q}^{\Gamma}$ or $\mathcal{Q}^{\Delta}$ are suitably renormalized). Since $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ are indistinguishable starting points when the $\mathcal{Q}^{\Gamma}$ transition is chosen, the entire process must contract with at least as much probability.

Lemma 5.5 *Let $\mathcal{Q}$ be any $n^{\triangleleft} \times n^{\triangleright}$ stochastic transition from $\Omega^{\triangleleft}$ to $\Omega^{\triangleright}$ as previously defined. For any non-negative $\gamma \leq \gamma_{\mathcal{Q}}$, and any pair of anterior probability distributions $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ over $\Omega^{\triangleleft}$, the matrix $\mathcal{Q}$ admits an additive decomposition $\mathcal{Q} = \mathcal{Q}^{\Gamma} + \mathcal{Q}^{\Delta}$ satisfying these three properties:*

1. $\forall i \in \{1, ..., n^{\triangleleft}\},\ \forall j \in \{1, ..., n^{\triangleright}\},\ 0 \leq \mathcal{Q}^{\Gamma}_{i,j} \leq \mathcal{Q}_{i,j}$;
2. $\forall i \in \{1, ..., n^{\triangleleft}\},\ \sum_{j=1}^{n^{\triangleright}} \mathcal{Q}^{\Gamma}_{i,j} = \gamma$;
3. $\forall j \in \{1, ..., n^{\triangleright}\},\ \sum_{i=1}^{n^{\triangleleft}} \boldsymbol{\varphi}^{\triangleleft}[\omega^{\triangleleft}_i]\, \mathcal{Q}^{\Gamma}_{i,j} = \sum_{i=1}^{n^{\triangleleft}} \boldsymbol{\psi}^{\triangleleft}[\omega^{\triangleleft}_i]\, \mathcal{Q}^{\Gamma}_{i,j}$.

The proof is based on the fact that, for at least a portion γ of their probability mass, the ulterior distributions $\boldsymbol{\varphi}^{\triangleright}$ and $\boldsymbol{\psi}^{\triangleright}$ must agree. The purpose of $\mathcal{Q}^{\Gamma}$ is to capture the fraction of $\mathcal{Q}$ that forces this agreement.

Proof The first task is to establish a correspondence from the masses of one distribution to the other. If some amount of mass in $\boldsymbol{\varphi}^{\triangleleft}$ sent to $\omega^{\triangleright}_j$ by $\mathcal{Q}$ could be mapped to a comparable amount of mass in $\boldsymbol{\psi}^{\triangleleft}$ also sent to $\omega^{\triangleright}_j$, this would ensure that this fraction of the mass will end up distributed identically.

Let $m_i = min\{\boldsymbol{\varphi}^{\triangleleft}[\omega^{\triangleleft}_i], \boldsymbol{\psi}^{\triangleleft}[\omega^{\triangleleft}_i]\}$, $p_i = \boldsymbol{\varphi}^{\triangleleft}[\omega^{\triangleleft}_i] - m_i$ and $q_i = \boldsymbol{\psi}^{\triangleleft}[\omega^{\triangleleft}_i] - m_i$. Note that for every i, either $p_i = 0$ or $q_i = 0$, or both. Intuitively, m_i is the fraction of mass that is common to both anterior distributions for $\omega^{\triangleleft}_i$, and is trivially mapped, while p_i is the portion of $\boldsymbol{\varphi}^{\triangleleft}[\omega^{\triangleleft}_i]$ that remains to be mapped, and similarly for q_i. Let $I_p = \{i : p_i > 0\}$ and $I_q = \{i : q_i > 0\}$. We now choose a set of numbers $\pi_{i_1,i_2} \geq 0$ such that $\sum_{i_2} \pi_{i_1,i_2} = p_{i_1}$ and $\sum_{i_1} \pi_{i_1,i_2} = p_{i_2}$. Since $\sum_{i_1} p_{i_1} = \sum_{i_2} p_{i_2}$, the construction of such numbers is straightforward. Let also another set of numbers $\kappa_{i_1,i_2,j} \geq 0$, such that $\sum_j \kappa_{i_1,i_2,j} = \gamma$ and $\kappa_{i_1,i_2,j} \leq min\{\mathcal{Q}[\omega^{\triangleright}_j|\omega^{\triangleleft}_{i_1}], \mathcal{Q}[\omega^{\triangleright}_j|\omega^{\triangleleft}_{i_2}]\}$. Such a choice is always possible, by the definition of $\gamma_{\mathcal{Q}}$ and the fact that $0 \leq \gamma \leq \gamma_{\mathcal{Q}}$.

We now construct the matrix $\mathcal{Q}^\Gamma$ as follows. For $i_0 \notin I_p \cup I_q$, define $\mathcal{Q}^\Gamma_{i_0,j} = \gamma\, \mathcal{Q}_{i_0,j}$. For $i_1 \in I_p$, we take $\mathcal{Q}^\Gamma_{i_1,j} = \sum_{i'_2} (\pi_{i_1,i'_2}/p_{i_1})\, \kappa_{i_1,i'_2,j}$, while for $i_2 \in I_q$, we let $\mathcal{Q}^\Gamma_{i_2,j} = \sum_{i'_1} (\pi_{i'_1,i_2}/q_{i_2})\, \kappa_{i'_1,i_2,j}$. Since I_p and I_q are disjoint, all cases are covered without conflict.

We now show that the construction satisfies the three required properties. It follows from the definition of π_{i_1,i_2} that, for $i_1 \in I_p$, $\mathcal{Q}^\Gamma_{i_1,j}$ is a weighted average of $\kappa_{i_1,i'_2,j}$ for varying i'_2. Properties 1 and 2 now follow from the definition of $\kappa_{i_1,i_2,j}$. An analogous reasoning holds for $\mathcal{Q}^\Gamma_{i_2,j}$, with $i_2 \in I_q$; and the result holds trivially for $\mathcal{Q}^\Gamma_{i_0,j}$, where $i_0 \notin I_p \cup I_q$. Property 3 ensues from the following derivations:

$$
\begin{aligned}
&\sum_{i=1}^{n^\triangleleft} \varphi^\triangleleft[\omega_i^\triangleleft]\, \mathcal{Q}^\Gamma_{i,j} - \sum_{i=1}^{n^\triangleright} \boldsymbol{\psi}^\triangleright[\omega_i^\triangleright]\, \mathcal{Q}^\Gamma_{i,j} \\
&\quad = \sum_{i=1}^{n^\triangleleft} (\boldsymbol{\varphi}^\triangleleft[\omega_i^\triangleleft] - \boldsymbol{\psi}^\triangleleft[\omega_i^\triangleleft])\, \mathcal{Q}^\Gamma_{i,j} \\
&\quad = \sum_{i=1}^{n^\triangleleft} (p_i - q_i)\, \mathcal{Q}^\Gamma_{i,j} \\
&\quad = \sum_{i_1 \in I_p} p_{i_1}\, \mathcal{Q}^\Gamma_{i_1,j} - \sum_{i_2 \in I_q} q_{i_2}\, \mathcal{Q}^\Gamma_{i_2,j} \\
&\quad = \sum_{i_1 \in I_p} p_{i_1} \sum_{i'_2} \frac{\pi_{i_1,i'_2}}{p_{i_1}}\, \kappa_{i_1,i'_2,j} - \sum_{i_2 \in I_q} q_{i_2} \sum_{i'_1} \frac{\pi_{i'_1,i_2}}{q_{i_2}}\, \kappa_{i'_1,i_2,j} \\
&\quad = \sum_{i_1 \in I_p} \sum_{i'_2} \pi_{i_1,i'_2}\, \kappa_{i_1,i'_2,j} - \sum_{i_2 \in I_q} \sum_{i'_1} \pi_{i'_1,i_2}\, \kappa_{i'_1,i_2,j} \\
&\quad = \sum_{i_1} \sum_{i_2} (\pi_{i_1,i_2} - \pi_{i_1,i_2})\, \kappa_{i_1,i_2,j}\ ,
\end{aligned}
$$

where the last step is based on the fact, implied by the definition of π_{i_1,i_2}, that $\pi_{i_1,i_2} = 0$ when $i_2 \in I_q$ or when $i_1 \in I_p$. Since that last expression equals 0, the desired result is established. ◻

Based on this lemma, the contraction result now follows easily. Essentially, the argument is based on a construction that makes explicit the different behavior of the

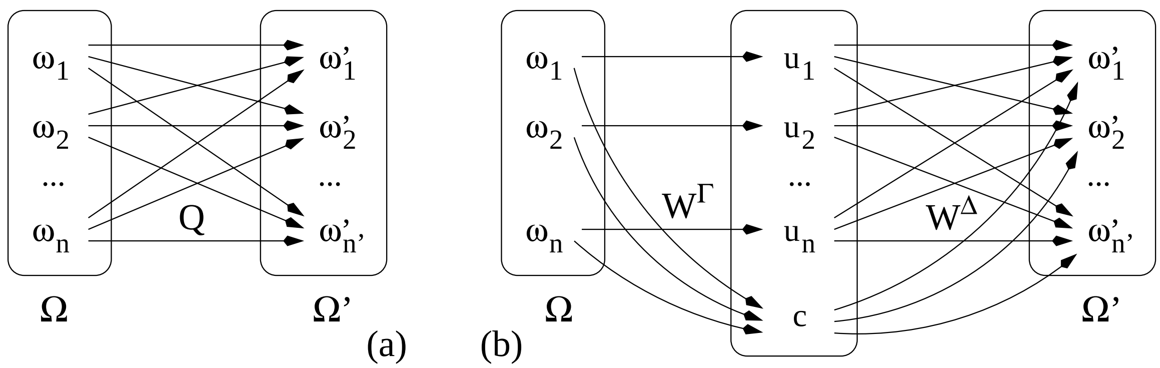


Figure 5.2: Decomposition of a stochastic transition, as used in Theorem 5.6: (a) discrete stochastic Markov transformation $\mathcal{Q}$; (b) equivalent two-phase transformation for the distributions $\varphi^{\triangleleft}$ and ψ. In this diagram, the arrows denote stochastic state transitions with non-zero probability.

stochastic transformation $\mathcal{Q}$, as induced by each part of the contraction decomposition, $\mathcal{Q}^{\Gamma}$ and $\mathcal{Q}^{\Delta}$. The idea is to decompose the transformation into two separate phases, as illustrated in Figure 5.2. In the first phase, the process randomly decides whether to contract and forget its starting state, or to preserve its state unchanged. In the second phase, the appropriate transition occurs, emulating either $\mathcal{Q}^{\Gamma}$ or $\mathcal{Q}^{\Delta}$, depending on the choice made in the first phase. The decomposition makes use of a new intermediate state space $\Omega^{\dagger}$, which duplicates the states of $\Omega^{\triangleleft}$, with one additional distinguished *contraction state* denoted **c**. The decision to contract corresponds to taking the transition from the initial state to **c**, in the first phase of the decomposed transformation; in **c**, the initial state is forgotten, and the subsequent behavior is constructed so that the whole process emulates $\mathcal{Q}^{\Gamma}$. The remaining states of $\Omega^{\dagger}$ correspond to the the initial states and are reached when there is no contraction; from there, the behavior is dictated by $\mathcal{Q}^{\Delta}$, so that, as a whole, the entire decomposed process behaves according to $\mathcal{Q}$ (at least for the specified $\varphi^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$, on which the decomposition depends).

Theorem 5.6 *For any discrete $\Omega^{\triangleleft}$, $\varphi^{\triangleleft}$, $\boldsymbol{\psi}^{\triangleleft}$, $\Omega^{\triangleright}$, $\varphi^{\triangleright}$, $\boldsymbol{\psi}^{\triangleright}$, $\mathcal{Q}$, $\gamma_{\mathcal{Q}}$, as above:*

$$\boldsymbol{D}_{KL}[\varphi^{\triangleright} \,\|\, \boldsymbol{\psi}^{\triangleright}] \;\; \leq \;\; (1-\gamma_{\mathcal{Q}})\, \boldsymbol{D}_{KL}[\varphi^{\triangleleft} \,\|\, \boldsymbol{\psi}^{\triangleleft}] \; .$$

Proof Fix $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$, and let $\Omega^{\dagger} = \{u_1, ..., u_{n^{\triangleleft}}, \mathbf{c}\}$ be a set of $n^{\triangleleft} + 1$ elements, where $n^{\triangleleft}$ is the cardinality of $\Omega^{\triangleleft}$. Define a two-phase stochastic transformation $\mathcal{W} = \mathcal{W}^{\Delta} \circ \mathcal{W}^{\Gamma}$, comprising a first Markovian transition $\mathcal{W}^{\Gamma}$ from $\Omega^{\triangleleft}$ to $\Omega^{\dagger}$, followed by a second Markovian transition $\mathcal{W}^{\Delta}$ from $\Omega^{\dagger}$ to $\Omega^{\triangleright}$, as in Figure 5.2. Intuitively, the distinguished intermediary state $\mathbf{c}$ is the state entered whenever contraction occurs, while the remaining intermediary states u_i simply duplicate the starting state in the converse situation. The first stage $\mathcal{W}^{\Gamma}$ is constructed to effect a random transition to the contraction state $\mathbf{c}$ with probability γ, and preserve its starting state with probability $1-\gamma$; *i.e.*, for all i, $\mathcal{W}^{\Gamma}[\mathbf{c}|\omega_i^{\triangleleft}] = \gamma$ while $\mathcal{W}^{\Gamma}[u_i|\omega_i^{\triangleleft}] = 1-\gamma$ and $\mathcal{W}^{\Gamma}[u_{i'}|\omega_i^{\triangleleft}] = 0$ for all $i' \neq i$. The second stage $\mathcal{W}^{\Delta}$ behaves on each u_i as $\mathcal{Q}^{\Delta}$ would on $\omega_i^{\triangleleft}$ (suitably normalized), while, from $\mathbf{c}$, it duplicates the aggregate behavior of $\mathcal{Q}^{\Gamma}$ on $\boldsymbol{\varphi}^{\triangleleft}$ (suitably normalized); *i.e.*, $\mathcal{W}^{\Delta}[\omega_j^{\triangleright}|u_i] = \mathcal{Q}^{\Delta}_{i,j}/(1-\gamma)$, while $\mathcal{W}^{\Delta}[\omega_j^{\triangleright}|\mathbf{c}] = \sum_i \boldsymbol{\varphi}^{\triangleleft}[\omega_i^{\triangleleft}]\, \mathcal{Q}^{\Gamma}_{i,j}/\gamma$. (It is noted that the normalized matrices $\mathcal{Q}^{\Gamma}/\gamma$ and $\mathcal{Q}^{\Delta}/(1-\gamma)$ define legitimate stochastic transformations, according to Lemma 5.5.)

We first need to show that the decomposed transformation is equivalent to the original $\mathcal{Q}$ when applied to either one of the distributions $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$. Let $\varphi^{\dagger}$ and $\boldsymbol{\psi}^{\dagger}$ be the result of applying $\mathcal{W}^{\Gamma}$ to $\varphi^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$, respectively. Consider the distribution obtained by applying the two-phase transformation $\mathcal{W} = \mathcal{W}^{\Delta} \circ \mathcal{W}^{\Gamma}$ to $\varphi^{\triangleleft}$. The probability assigned to $\omega_j^{\triangleright}$ by the resulting distribution is:

$$\begin{aligned}
&\frac{1}{1-\gamma} \sum_i \varphi^{\dagger}[u_i]\, \mathcal{Q}^{\Delta}_{i,j} + \frac{\varphi^{\dagger}[\mathbf{c}]}{\gamma} \sum_{i'} \varphi^{\triangleleft}[\omega_{i'}^{\triangleleft}]\, \mathcal{Q}^{\Gamma}_{i',j} \\
&\quad = \frac{1}{1-\gamma} \sum_i (1-\gamma)\, \varphi^{\triangleleft}[\omega_i^{\triangleleft}]\, \mathcal{Q}^{\Delta}_{i,j} + \frac{\gamma}{\gamma} \sum_{i'} \boldsymbol{\varphi}^{\triangleleft}[\omega_i^{\triangleleft}]\, \mathcal{Q}^{\Gamma}_{i',j} \\
&\quad = \sum_i \boldsymbol{\varphi}^{\triangleleft}[\omega_i^{\triangleleft}]\, \mathcal{Q}^{\Delta}_{i,j} + \sum_{i'} \boldsymbol{\varphi}^{\triangleleft}[\omega_{i'}^{\triangleleft}]\, \mathcal{Q}^{\Gamma}_{i',j} \\
&\quad = \sum_i \boldsymbol{\varphi}^{\triangleleft}[\omega_i^{\triangleleft}]\, \mathcal{Q}_{i,j} \ .
\end{aligned}$$

Similarly, applying $\mathcal{W} = \mathcal{W}^{\Delta} \circ \mathcal{W}^{\Gamma}$ to $\boldsymbol{\psi}^{\triangleleft}$ and evaluating at $\omega_j^{\triangleright}$ yields the probability $\sum_i \boldsymbol{\psi}^{\triangleleft}[\omega_i^{\triangleleft}]\, \mathcal{Q}^{\Delta}_{i,j} + \sum_i \varphi^{\triangleleft}[\omega_i^{\triangleleft}]\, \mathcal{Q}^{\Gamma}_{i,j}$, which, by property 3 of Lemma 5.5, reduces to $\sum_i \boldsymbol{\psi}^{\triangleleft}[\omega_i^{\triangleleft}]\, \mathcal{Q}_{i,j}$. This shows that $\mathcal{W}$ correctly emulates $\mathcal{Q}$ on both $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$, as required.

To show the contraction property $\boldsymbol{D}_{KL}[\boldsymbol{\varphi}^{\triangleright} \,\|\, \boldsymbol{\psi}^{\triangleright}] \leq (1-\gamma)\,\boldsymbol{D}_{KL}[\boldsymbol{\varphi}^{\triangleleft} \,\|\, \boldsymbol{\psi}^{\triangleleft}]$, we first note that since $\mathcal{W}^{\Delta}$ is Markovian, we have $\boldsymbol{D}_{KL}[\boldsymbol{\varphi}^{\triangleright} \,\|\, \boldsymbol{\psi}^{\triangleright}] \leq \boldsymbol{D}_{KL}[\boldsymbol{\varphi}^{\dagger} \,\|\, \boldsymbol{\psi}^{\dagger}]$, by the well-known property that relative entropy does not increase through a stochastic transformation [CT91, page 34]. On the other hand, we have:

$$
\begin{aligned}
\boldsymbol{D}_{KL}[\boldsymbol{\varphi}^{\dagger} \,\|\, \boldsymbol{\psi}^{\dagger}] &= \sum_i \boldsymbol{\varphi}^{\dagger}[u_i]\, log_2 \frac{\boldsymbol{\varphi}^{\dagger}[u_i]}{\boldsymbol{\psi}^{\dagger}[u_i]} + \boldsymbol{\varphi}^{\dagger}[\mathbf{c}]\, log_2 \frac{\boldsymbol{\varphi}^{\dagger}[\mathbf{c}]}{\boldsymbol{\psi}^{\dagger}[\mathbf{c}]} \\
&= \sum_i (1-\gamma)\, \boldsymbol{\varphi}^{\triangleleft}[\omega_i^{\triangleleft}]\, log_2 \frac{\boldsymbol{\varphi}^{\triangleleft}[\omega_i^{\triangleleft}]}{\boldsymbol{\psi}^{\triangleleft}[\omega_i^{\triangleleft}]} + \gamma\, log_2 \frac{\gamma}{\gamma} \\
&= (1-\gamma)\, \boldsymbol{D}_{KL}[\boldsymbol{\varphi}^{\triangleleft} \,\|\, \boldsymbol{\psi}^{\triangleleft}]\ .
\end{aligned}
$$

Letting $\gamma = \gamma_{\mathcal{Q}}$ in the above expression concludes the proof. ◻

It is emphasized that the above transformation is specific to the given $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$.

To conclude this section, we remark that a mixing coefficient equal to 0 provides no help in terms of contraction, since a null mixing coefficient is the reflection of some kind of "deterministic core" within the process, at least for one stochastic transition. This can happen even for ergodic processes. In this case, there is little that our single-step contraction analysis can provide. However, it is sometimes the case that a null mixing coefficient is due not to an unavoidable deterministic component of (part of) the process, but to some simplifying assumption in its modelization, or to a bounded "propagation speed" between non-neighboring states. In this case, it is useful to consider a aggregate process obtained by composing a finite number of elementary transitions of the original process. The analysis could be generalized this way to arbitrary finite-state discrete-time ergodic systems by considering the minimum time after which pairwise mixing must occur. In slight anticipation of the next section, we note that, even though it is fairly harmless to lump together several time slices for the contraction analysis in the flat case, this will not necessarily be the case when we deal with structured contraction.

5.3 Structured contraction

The previous results show that errors decrease geometrically in stochastic Markov processes. The rate of decrease, however, depends on the transition model of the process, and more specifically on its mixing coefficient γ_Q. Clearly, one cannot guarantee that this coefficient will always be large enough to be useful: for example, if the process has a component which is almost deterministic, γ_Q will be very close to zero.

That deterministic processes should not contract, and would therefore not be amenable to belief state approximation, is not cause for concern; after all, deterministic systems often tend to preserve and propagate their state in precise ways for extended periods of time, and do not respond well to external noise and other random disturbances. This restriction also applies, to a lesser extent, to locally distinguishable stochastic processes, in which contraction can only be guaranteed after a plurality of transitions have been taken. In this case, the contraction analysis applies to the macro-transition obtained by combining the appropriate number of elementary transitions. Unfortunately, a lack of useful contraction also arises for truly stochastic processes with a large number of variables, even if each individual variable is governed by nicely stochastic transition dynamics.

Example 5.7 As an extreme example, imagine a process composed of N binary variables evolving independently in the domain $\{0, 1\}$, flipping their value from one time slice to the next with probability δ. Each variable, viewed as a separate Markov process, has a mixing coefficient of $min\{2\,\delta, 2\,(1-\delta)\}$, and is thus rapidly mixing provided $\delta \in [\varepsilon \cdot\cdot 1-\varepsilon]$, for a large enough $\varepsilon > 0$. Thus, one may expect the N-variable process as a whole to be similarly rapidly mixing; indeed, since all of the N subprocesses are independent, one could hardly expect otherwise. However, Theorem 5.6 tells a different story: computing the mixing coefficient γ for the transition matrix of the compound process as a whole, according to Definition 5.4, one obtains a discouragingly small value: $\gamma \leq 2\,(2\,e\,\delta)^{N/2}$.

To see this, observe that γ is bounded by twice the probability that at least $N/2$ of the N variables flip[1]. Let X be the number of flips in a given step; X has a binomial distribution $\mathcal{B}(N,\delta)$, which, for small δ and bounded $\mu = \delta\, N$, can be approximated as a Poisson distribution $\mathcal{P}(\mu)$ of mean μ. The probability of interest is now easily bounded using the Chernoff bound, $\mathbb{P}_{X\sim\mathcal{P}(\mu)}[X \geq \xi] \leq e^{-\mu}\,(e\,\mu)^{\xi}/\xi^{\xi}$ for $\xi > \mu$, giving in our case [Ros96, page 40]:

$$\mathbb{P}_{X\sim\mathcal{P}(\mu)}[X \geq N/2] \;\leq\; e^{-\mu}\,(\frac{e\,\mu}{N/2})^{N/2} \;=\; e^{-N\,\delta}\,(2\,e\,\delta)^{N/2} \;\leq\; (2\,e\,\delta)^{N/2}\ .$$

∎

Is our definition of the mixing coefficient simply too pessimistic? Unfortunately not. The fallacy lies in the assumption that local mixing properties would automatically carry over to the compound process. In the example, each subprocess is rapidly mixing for belief states defined over its own variable only. If the state of the compound process somehow involves dependencies between variables belonging to different subprocesses, then the compound belief state will contain correlations that are invisible at the level of individual variables. And, indeed, such correlations can render the contraction ratio as bad as the theorem predicts.

Example 5.8 In the context of the previous example, assume that the true starting state distribution $\boldsymbol{\varphi}^{\triangleleft}$ of the compound process gives probability 1 to the state $\langle 0,\ldots,0\rangle$ (expressed as an assignment of values to the N variables); meanwhile, the estimated starting distribution $\boldsymbol{\psi}^{\triangleleft}$ gives some probability p to that state, and probability $q = 1-p$ to its complement $\langle 1,\ldots,1\rangle$. The state space can be viewed as a hypercube, and each of these distributions as an assignment of probability mass to vertices of the hypercube. A single step through the stochastic transformation diffuses the respective mass of the two distributions around their starting points. However, the probability that the diffusion process around two opposite corners of the hypercube brings them to the same place is exponentially low, since all of the bits have to flip in one or the other of the two distributions. In the example, the relative entropy between the starting

[1]Consider two diametrically opposed corners a and b on the hypercube, and a dichotomy of the cube in two regions based on closest proximity to the respective corners. For two imaginary markers starting on a and b to meet after one stochastic transition, it is necessary that at least (in fact, exactly) one of them travel to the region occupied by the other.

distributions $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ is fairly low already, commensurate to $log_2\,[1/p] \simeq 1 - p = q$, provided that $p \simeq 1$. However, after the stochastic transition, even though $\boldsymbol{\varphi}^{\triangleleft}$ will have moved in unison with the fraction of $\boldsymbol{\psi}^{\triangleleft}$ at $\langle 0, ..., 0\rangle$, hardly any of it will have traveled enough along the hypercube to meet the remaining fraction of $\boldsymbol{\psi}$ that started at $\langle 1, ..., 1\rangle$. In terms of relative entropy, this translates to an almost imperceptible reduction of the error.

∎

Given the above scenario, one must face the question of whether the previous contraction theorem is even useful for large processes. As the example shows, even the assumption that the process is highly decomposable is not necessarily helpful. One idea is to make some additional assumptions about the structure of the state distributions, for instance, that they decompose into a set of factors. In general, an analysis based on decomposing a relative entropy expression requires an assumption on the decomposability of the true or reference distribution (the first argument of $\mathbb{D}_{KL}[\cdot \,\|\, \cdot]$). In the present context, such an assumption would be patently false; indeed, the whole point of Section 4.1 was that the state distributions in dynamic systems tend to lose any kind of structure, even when the transition model itself features a lot of structure. Furthermore, the above scenario shows that decomposability—and, indeed, complete independence—of the reference distribution $\boldsymbol{\varphi}^{\triangleleft}$ did not help in this context.

Surprisingly, significant advantage may be gained by making a decomposability assumption on the approximated belief state. Specifically, we can show that if the process decomposes well, and the estimated distribution decomposes in a way that matches the structure of the process, then significantly better bounds on the error contraction coefficient can be obtained, regardless of the true belief state. Thus, as far as error contraction goes, the properties of the true belief state are not crucial: only those of the approximate belief state and the process itself matter. This is very fortunate, as it is feasible to enforce decomposability properties on the approximate belief state, whereas the true belief state usually remains beyond control.

Formally, it is most convenient to describe our results in the framework of interacting HMMs; in the next section, we discuss how they can be applied to dynamic

Bayesian networks. In the framework of interacting HMMs, a stochastic process is composed of a number L of subprocesses. Each subprocess has a state, and is described by a disctete-time Markovian evolution model. The state of subprocess l at time t is denoted $S_l^{(t)}$. The transition model, denoted $\mathcal{S}_l$, is a stochastic mapping from the states of some set of subprocesses at some time $t-1$ to the state of process l at the time slice t; the mapping is time-invariant, and thus applies for all t. We say that subprocess l *depends* on subprocess l' if the map $\mathcal{S}_l$ takes the state of subprocess l' as argument. In addition, the stochastic process may include a set of response variables, which can depend arbitrarily on the state of the various subprocesses in the same time slice; however, as we are primarily interested in the contraction properties of the transition model, the response variables as well as the observation model and the conditioning phase are irrelevant to the analysis. For this reason, and for the sake of simplicity, notations such as $\boldsymbol{\sigma}^{(t-1\bullet)}$ and $\boldsymbol{\sigma}^{(\bullet t)}$ are temporarily dropped in favor of $\boldsymbol{\sigma}^{(t-1)}$ and $\boldsymbol{\sigma}^{(t)}$ respectively.

5.3.1 Independent subprocesses

We begin by considering the simple case where multiple subprocesses exist, but are completely independent, *i.e.*, where each subprocess depends only on itself. In addition, we require that, at any time t, the estimated belief state $\tilde{\boldsymbol{\sigma}}^{(t)}$ decomposes along the same lines, *i.e.*, as the product of independent estimated belief states over the individual subprocesses: $\tilde{\boldsymbol{\sigma}}^{(t)} = \prod_l \tilde{\boldsymbol{\sigma}}_l^{(t)}$. We show that, under those fairly restrictive assumptions, the contraction ratio of the entire process is commensurate to that of the individual subprocesses, regardless of their numbers. Later on, this theorem will form the basis for a more general result. We now prove the theorem, starting with the following lemma.

Lemma 5.9 *Let $\boldsymbol{\varphi}^\star$ and $\boldsymbol{\psi}^\star$ be two distributions over the same space $\Omega^\star$. Let W and Z be two random variables (or sets of variables) over this space, and E be some event in the space. If W and Z are conditionally independent given E in $\boldsymbol{\psi}^\star$, then:*

$$\mathbb{D}_{KL}[\boldsymbol{\varphi}^\star[Z|E] \,\|\, \boldsymbol{\psi}^\star[Z|E]] \;\; \leq \;\; \boldsymbol{E}_{\boldsymbol{\varphi}^\star[W|E]}[\mathbb{D}_{KL}[\boldsymbol{\varphi}^\star[Z|W,E] \,\|\, \boldsymbol{\psi}^\star[Z|W,E]]] \; .$$

Proof Letting $\boldsymbol{H}[\cdot]$ represent the standard entropy function as defined in Section 2.2, we have:

$$\begin{aligned}
&\boldsymbol{E}_{\boldsymbol{\varphi}^\star[W|E]}[\boldsymbol{D}_{KL}[\boldsymbol{\varphi}^\star[Z|W,E] \,\|\, \boldsymbol{\psi}^\star[Z|W,E]]] \\
&\quad= \boldsymbol{E}_{\boldsymbol{\varphi}^\star[W|E]}[\boldsymbol{E}_{\boldsymbol{\varphi}^\star[Z|W,E]}[\log_2 \boldsymbol{\varphi}^\star[Z|W,E]] - \boldsymbol{E}_{\boldsymbol{\varphi}^\star[Z|W,E]}[\log_2 \boldsymbol{\psi}^\star[Z|W,E]]] \\
&\quad= \boldsymbol{E}_{\boldsymbol{\varphi}^\star[W|E]}[-\boldsymbol{H}[\boldsymbol{\varphi}^\star[Z|W,E]]] - \boldsymbol{E}_{\boldsymbol{\varphi}^\star[W|E]}[\boldsymbol{E}_{\boldsymbol{\varphi}^\star[Z|W,E]}[\log_2 \boldsymbol{\psi}^\star[Z|E]]] \\
&\quad\geq -\boldsymbol{H}[\boldsymbol{E}_{\boldsymbol{\varphi}^\star[W|E]}[\boldsymbol{\varphi}^\star[Z|W,E]]] - \boldsymbol{E}_{\boldsymbol{\varphi}^\star[W|E]}[\boldsymbol{E}_{\boldsymbol{\varphi}^\star[Z|W,E]}[\log_2 \boldsymbol{\psi}^\star[Z|E]]] \\
&\quad= -\boldsymbol{H}[\boldsymbol{\varphi}^\star[Z|E]] - \boldsymbol{E}_{\boldsymbol{\varphi}^\star[Z|E]}[\log_2 \boldsymbol{\psi}^\star[Z|E]] \\
&\quad= \boldsymbol{E}_{\boldsymbol{\varphi}^\star[Z|E]}[\log_2 \boldsymbol{\varphi}^\star[Z|E]] - \boldsymbol{E}_{\boldsymbol{\varphi}^\star[Z|E]}[\log_2 \boldsymbol{\psi}^\star[Z|E]] \\
&\quad= \boldsymbol{D}_{KL}[\boldsymbol{\varphi}^\star[Z|E] \,\|\, \boldsymbol{\psi}^\star[Z|E]] \ .
\end{aligned}$$

The second equality is by linearity of expectation and the conditional independence assumption for $\boldsymbol{\psi}^\star$. The inequality follows from the convexity of the negative entropy $-\boldsymbol{H}[\cdot]$, as a consequence of Jensen's inequality [CT91, sec. 2.6]. The subsequent equality follows from marginalization. The remaining steps are definitional. ◻

Theorem 5.10 *Consider a process $\mathcal{Q}$ composed of L independent stochastic subprocesses $\mathcal{Q}_1, ..., \mathcal{Q}_L$, where each subprocess $\mathcal{Q}_l$ maps distributions over $\Omega_l^\triangleleft$ to distributions over $\Omega_l^\triangleright$, and depends only on itself. Let γ_l be the mixing coefficient of $\mathcal{Q}_l$, and let $\gamma = \min\{\gamma_1, ..., \gamma_l\}$. Let $\boldsymbol{\varphi}^\triangleleft$ and $\boldsymbol{\psi}^\triangleleft$ be two arbitrary distributions over the joint anterior state space $\Omega_1^\triangleleft \times ... \times \Omega_L^\triangleleft$. Let $\boldsymbol{\varphi}^\triangleright$ and $\boldsymbol{\psi}^\triangleright$ be the corresponding ulterior distributions induced by $\mathcal{Q}$ over $\Omega_1^\triangleright \times ... \times \Omega_L^\triangleright$. If the subspaces $\Omega_l^\triangleleft$ are marginally independent in the anterior distribution $\boldsymbol{\psi}^\triangleleft$ (but not necessarily in $\boldsymbol{\varphi}^\triangleleft$), then the entire process obeys the following contraction property:*

$$\boldsymbol{D}_{KL}[\boldsymbol{\varphi}^\triangleright \,\|\, \boldsymbol{\psi}^\triangleright] \quad\leq\quad (1-\gamma)\, \boldsymbol{D}_{KL}[\boldsymbol{\varphi}^\triangleleft \,\|\, \boldsymbol{\psi}^\triangleleft] \ .$$

Proof It suffices to show the result for two independent subprocesses $\mathcal{Q}_X$ and $\mathcal{Q}_Y$; the general case follows by induction on the number of subprocesses.

Let $X^\triangleleft$ and $X^\triangleright$ be the random variables over $\Omega_X^\triangleleft$ and $\Omega_X^\triangleright$ respectively, describing the anterior and ulterior state of $\mathcal{Q}_X$, *i.e.*, before and after the transition $\mathcal{Q}_X$.

Similarly, let $Y^{\triangleleft}$ and $Y^{\triangleright}$ describe the anterior and ulterior state of $\mathcal{Q}_Y$, with analogous notations. Let $\boldsymbol{\varphi}^{\triangleleft}[X^{\triangleleft}, Y^{\triangleleft}]$ and $\boldsymbol{\psi}^{\triangleleft}[X^{\triangleleft}, Y^{\triangleleft}]$ be the given distributions over the joint anterior state space $\Omega_X^{\triangleleft} \times \Omega_Y^{\triangleleft}$. Let $\boldsymbol{\varphi}^{\star}[X^{\triangleleft}, Y^{\triangleleft}, X^{\triangleright}, Y^{\triangleright}]$ and $\boldsymbol{\psi}^{\star}[X^{\triangleleft}, Y^{\triangleleft}, X^{\triangleright}, Y^{\triangleright}]$ be the joint distributions over the anterior and ulterior spaces, as respectively induced from $\boldsymbol{\varphi}^{\triangleleft}[X^{\triangleleft}, Y^{\triangleleft}]$ and $\boldsymbol{\psi}^{\triangleleft}[X^{\triangleleft}, Y^{\triangleleft}]$ by the process $\mathcal{Q}$, *i.e.*, $\boldsymbol{\varphi}^{\star}[X^{\triangleleft}, Y^{\triangleleft}, X^{\triangleright}, Y^{\triangleright}] = \boldsymbol{\varphi}^{\triangleleft}[X^{\triangleleft}, Y^{\triangleleft}] \otimes \mathcal{Q}[X^{\triangleright}, Y^{\triangleright} | X^{\triangleleft}, Y^{\triangleleft}]$.

Using the standard decomposition properties of relative entropy, we obtain:

$$\begin{aligned} & \mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[X^{\triangleright}, Y^{\triangleright}] \,\|\, \boldsymbol{\psi}^{\star}[X^{\triangleright}, Y^{\triangleright}]] \\ = \;& \mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[X^{\triangleright}] \,\|\, \boldsymbol{\psi}^{\star}[X^{\triangleright}]] + \mathbb{E}_{\boldsymbol{\varphi}^{\star}[X^{\triangleright}]}[\mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[Y^{\triangleright}|X^{\triangleright}] \,\|\, \boldsymbol{\psi}^{\star}[Y^{\triangleright}|X^{\triangleright}]]] \;. \end{aligned} \tag{5.1}$$

By the contraction property for $\mathcal{Q}_X$ per Theorem 5.6, the first term of Expression 5.1 is bounded as:

$$\mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[X^{\triangleright}] \,\|\, \boldsymbol{\psi}^{\star}[X^{\triangleright}]] \;\leq\; (1-\gamma)\, \mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[X^{\triangleleft}] \,\|\, \boldsymbol{\psi}^{\star}[X^{\triangleleft}]] \;. \tag{5.2}$$

To simplify the second term, we apply Lemma 5.9 to the internal component of the expectation, substituting $Y^{\triangleright}$ for Z, the value of $X^{\triangleright}$ for E, and $X^{\triangleleft}$ for W. The conditions of the lemma hold due to our assumptions: $X^{\triangleleft}$ and $Y^{\triangleleft}$ are independent in $\boldsymbol{\psi}^{\star}$, and the subprocesses evolve independently; therefore, the pairs $\langle X^{\triangleleft}, X^{\triangleright}\rangle$ and $\langle Y^{\triangleleft}, Y^{\triangleright}\rangle$ are independent in $\boldsymbol{\psi}^{\star}$; and certainly $X^{\triangleleft}$ and $Y^{\triangleright}$ are conditionally independent given $X^{\triangleright}$. It follows that:

$$\begin{aligned} & \mathbb{E}_{\boldsymbol{\varphi}^{\star}[X^{\triangleright}]}[\mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[Y^{\triangleright}|X^{\triangleright}] \,\|\, \boldsymbol{\psi}^{\star}[Y^{\triangleright}|X^{\triangleright}]]] \\ \leq \;& \mathbb{E}_{\boldsymbol{\varphi}^{\star}[X^{\triangleright}]}[\mathbb{E}_{\boldsymbol{\varphi}^{\star}[X^{\triangleleft}|X^{\triangleright}]}[\mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[Y^{\triangleright}|X^{\triangleright}, X^{\triangleleft}] \,\|\, \boldsymbol{\psi}^{\star}[Y^{\triangleright}|X^{\triangleright}, X^{\triangleleft}]]]] \\ = \;& \mathbb{E}_{\boldsymbol{\varphi}^{\star}[X^{\triangleleft}, X^{\triangleright}]}[\mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[Y^{\triangleright}|X^{\triangleleft}] \,\|\, \boldsymbol{\psi}^{\star}[Y^{\triangleright}|X^{\triangleleft}]]] \\ = \;& \mathbb{E}_{\boldsymbol{\varphi}^{\star}[X^{\triangleleft}]}[\mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[Y^{\triangleright}|X^{\triangleleft}] \,\|\, \boldsymbol{\psi}^{\star}[Y^{\triangleright}|X^{\triangleleft}]]] \\ \leq \;& \mathbb{E}_{\boldsymbol{\varphi}^{\star}[X^{\triangleleft}]}[(1-\gamma)\, \mathbb{D}_{KL}[\boldsymbol{\varphi}^{\star}[Y^{\triangleleft}|X^{\triangleleft}] \,\|\, \boldsymbol{\psi}^{\star}[Y^{\triangleleft}|X^{\triangleleft}]]] \;, \end{aligned} \tag{5.3}$$

where the second equality follows from the conditional independence assumptions, and the last inequality follows from the contraction property of Theorem 5.6 for $\mathcal{Q}_Y$, applied to each of the pairs of distributions $\boldsymbol{\varphi}^\star[Y^\triangleleft | X^\triangleleft = x^\triangleleft]$ and $\boldsymbol{\psi}^\star[Y^\triangleleft | X^\triangleleft = x^\triangleleft]$, for all possible values of $x^\triangleleft \in \Omega_X^\triangleleft$.

Putting together Expressions 5.1, 5.2, and 5.3, it follows that:

$$\begin{aligned}
&\boldsymbol{D}_{KL}[\boldsymbol{\varphi}^\star[X^\triangleright, Y^\triangleright] \,\|\, \boldsymbol{\psi}^\star[X^\triangleright, Y^\triangleright]] \\
&\quad\le\ (1-\gamma)\, \boldsymbol{D}_{KL}[\boldsymbol{\varphi}^\star[X^\triangleleft] \,\|\, \boldsymbol{\psi}^\star[X^\triangleleft]] + (1-\gamma)\, \boldsymbol{E}_{\boldsymbol{\varphi}^\star[X^\triangleleft]}[\boldsymbol{D}_{KL}[\boldsymbol{\varphi}^\star[Y^\triangleleft | X^\triangleleft] \,\|\, \boldsymbol{\psi}^\star[Y^\triangleleft | X^\triangleleft]]] \\
&\quad=\ (1-\gamma)\, \boldsymbol{D}_{KL}[\boldsymbol{\varphi}^\star[X^\triangleleft, Y^\triangleleft] \,\|\, \boldsymbol{\psi}^\star[X^\triangleleft, Y^\triangleleft]] \ ,
\end{aligned}$$

as required. ◻

Thus, for a process composed of a set of independent subprocesses, if, in addition, the chosen representation of the approximate belief state decomposes along the same lines as the process, then the contraction of the process as a whole is no worse than the contraction of the individual subprocesses. Since each subprocess involves an exponentially smaller number of states than the whole process, its transition probabilities are likely to be correspondingly much larger (assuming the process is reasonably stochastic). As much better mixing coefficients are to be expected for the individual subprocesses than for the process as a whole, the analysis of Theorem 5.10 typically provides in a much more useful contraction ratio than Lemma 5.6, when it is applicable.

Nevertheless, the above result is not really useful in itself, because, if the subprocesses were really independent, the belief state would never become correlated in the first place, and there would be no need to approximate it. The main purpose of this result is to lay a foundation for the general case to be studied next.

5.3.2 Conditionally independent subprocesses

The objective now is to generalize the above result to the more realistic situation where the different subprocesses are allowed to depend on each other.

As before, consider a process composed of L subprocesses, and assume that subprocess l depends on subprocesses $l_1, ..., l_k$. Then, the stochastic transition $\mathcal{Q}_l$ defines a conditional probability measure $I\!P[S_l^{\triangleright} | S_{l_1}^{\triangleleft}, ..., S_{l_k}^{\triangleleft}]$, where $\mathbf{S}_l$ denotes the time-dependent random variable describing the state of subprocess l, and $S_l^{\triangleleft}$ and $S_l^{\triangleright}$ denote the instantiation of $\mathbf{S}_l$ at two consecutive times. This transition probability can be defined as a transition matrix, albeit one whose anterior and ulterior state spaces differ. Since the treatment introduced in Section 5.2 accounted for this possibility, the mixing coefficient of $\mathcal{Q}_l$ is well-defined. Thus, let γ_l be the mixing coefficient of $\mathcal{Q}_l$, and let $\gamma = min\{\gamma_1, ..., \gamma_L\}$. We show that, as before, if the estimated belief state decomposes along the lines of the process structure, then we can place a bound on the contraction ratio of the entire process. This bound depends on both γ and the process structure. We first describe the basic construction for a structure composed of two subprocesses; generalization to arbitrary structures is straightforward.

Consider the process depicted in Figure 5.3(a). This process comprises two subprocesses $\mathcal{Q}_X$ and $\mathcal{Q}_Y$, in which $\mathcal{Q}_X$ depends on $\mathcal{Q}_X$ alone, and $\mathcal{Q}_Y$ depends on both $\mathcal{Q}_X$ and $\mathcal{Q}_Y$. Our construction follows the lines of the proof of Theorem 5.6: the new process is obtained by splitting the transition of each subprocess into two successive phases, where the first one chooses whether or not to contract, and the second one concludes the transition in a way that depends on whether the subprocess has contracted. For the subprocess $\mathcal{Q}_Y$, this involves the construction of the intermediate variable $Y_Y^{\dagger}$ to represent the state of $\mathbf{Y}$ after the first phase of the two-phase transition, in addition to the anterior and ulterior instantiations $Y^{\triangleleft}$ and $Y^{\triangleright}$ of $\mathbf{Y}$, which have their usual meaning. A subtlety is that, since $\mathcal{Q}_Y$ depends on $\mathcal{Q}_X$, the variable $\mathbf{X}$ plays a role in both $\mathcal{Q}_X$ and $\mathcal{Q}_Y$; furthermore, the transitions of these two subprocesses are conditionally independent given $X^{\triangleleft}$. Thus, $\mathbf{X}$ cannot make a single decision to contract, and apply it in the context of both processes; rather, $\mathbf{X}$ has to make two independent decisions as to whether to contract in the context of $\mathcal{Q}_X$ and $\mathcal{Q}_Y$ in isolation. We therefore introduce two separate intermediate variables $X_X^{\dagger}$ and $X_Y^{\dagger}$ to capture the state of $\mathbf{X}$ after the first phase, where $X_X^{\dagger}$ is relevant to $\mathcal{Q}_X$, and $X_Y^{\dagger}$ to $\mathcal{Q}_Y$. The result of the construction is the segmented process shown in Figure 5.3(b).

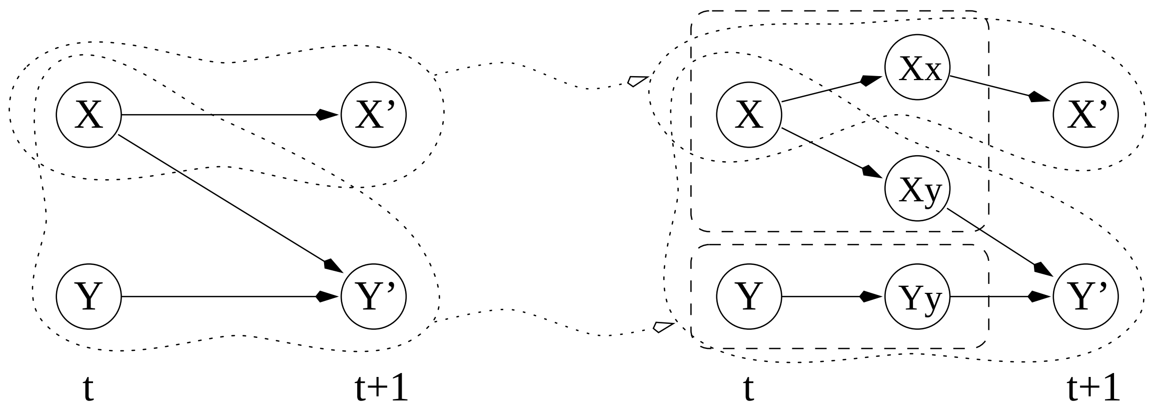


Figure 5.3: Principle of the construction used in Theorem 5.11: (left) Markov process composed of two subprocesses; (right) decomposition as a two-phase segmented process. The thin dotted contours delineate corresponding stochastic transformations that induce the same ulterior distribution in (left) and (right), when the anterior distribution is either $\boldsymbol{\varphi}[X^{\triangleleft}, Y^{\triangleleft}]$ or $\boldsymbol{\psi}[X^{\triangleleft}, Y^{\triangleleft}]$. The dashed boxes emphasize that, throughout the first stage of (right), the two subprocesses are fully independent.

The stochastic transitions of the segmented process are defined so that the two-phase transition,

$$X^{\triangleleft} \to X_X^{\dagger} \to X^{\triangleright} ,$$

induces the same ulterior distributions as $\mathcal{Q}_X$ from the anterior distributions $\boldsymbol{\varphi}^{\triangleleft}[X^{\triangleleft}]$ and $\boldsymbol{\psi}^{\triangleleft}[X^{\triangleleft}]$. This construction is essentially identical to that of Theorem 5.6; in particular, the domain of the intermediate variable $X_X^{\triangleleft}$ duplicates all the states in the domain of $X^{\triangleleft}$, plus one additional contraction state denoted $\mathbf{c}_{X_X}$. As before, the first stage of the segmented process either preserves the state of $X^{\triangleleft}$, or jumps to the contraction state $\mathbf{c}_{X_X}$ with some probability λ_{X_X} to be specified below. Similarly, the two-phase transition,

$$\begin{array}{l} X^{\triangleleft} \to X_Y^{\dagger} \searrow \\ Y^{\triangleleft} \to Y_Y^{\dagger} \to Y^{\triangleright} , \end{array}$$

is constructed to induce the same behavior as $\mathcal{Q}_Y$ for the distributions $\boldsymbol{\varphi}^{\triangleleft}[X^{\triangleleft}, Y^{\triangleleft}]$ and $\boldsymbol{\psi}^{\triangleleft}[X^{\triangleleft}, Y^{\triangleleft}]$, with a slight subtlety. It was explained in the proof of Theorem 5.6 that the second phase should transition according to $\mathcal{Q}^{\Delta}$, except when in the contraction state, in which case a special behavior should be invoked. In the present situation, the transition $\langle X_Y^\dagger, Y_Y^\dagger \rangle \to Y'$ has two contraction substates to consider: one for $X_Y^\dagger$, and one for $Y_Y^\dagger$. The prescription is that either one will do: for the purpose of this transition, contraction is deemed in play as soon as at least one of $X_Y^\dagger$ and $Y_Y^\dagger$ is in a contraction state. There is no alternative, because if even one is in the contraction state, the process no longer has enough information to transition according to $\mathcal{Q}_Y^\Delta$. Thus, if $X^\triangleleft \to X_Y^\dagger$ contracts with probability λ_{X_Y} and $Y^\triangleleft \to Y_Y^\dagger$ with probability λ_{Y_Y}, the $\mathbf{Y}$ process as a whole contracts with probability $1-(1-\lambda_{X_Y})(1-\lambda_{Y_Y})$.

Therefore, in order to be able to construct a contraction decomposition $(\mathcal{Q}_Y^\Gamma, \mathcal{Q}_Y^\Delta)$ as in Theorem 5.6, one must select λ_{X_X}, λ_{X_Y}, λ_{Y_Y} so as to satisfy $\lambda_{X_X} \le \gamma_{\mathcal{Q}_X}$ and $1-(1-\lambda_{X_Y})(1-\lambda_{Y_Y}) \le \gamma_{\mathcal{Q}_Y}$. Assuming γ such that $\gamma = \gamma_{\mathcal{Q}_X} = \gamma_{\mathcal{Q}_Y}$, it follows that $\lambda_{X_Y} \le 1-\sqrt{1-\gamma}$ and $\lambda_{Y_Y} \le 1-\sqrt{1-\gamma}$ are one legitimate set of constraints; observe that $1-\sqrt{1-\gamma} \ge \gamma/2$.

As for the contraction ratio of the process as a whole, it is no smaller than that of the first stage alone, since the first and second stages form a Markov chain. Thus, we focus on the contraction from the anterior variables $X^\triangleleft, Y^\triangleleft$ to the intermediate variables $X_X^\dagger, X_Y^\dagger, Y_Y^\dagger$. This analysis uses a somewhat different process structure than the one just used to show the correctness of the partition. Let $\mathcal{W}_X$ denote the joint transition from $X^\triangleleft$ to the pair $\langle X_X^\dagger, X_Y^\dagger \rangle$, and let $\mathcal{W}_Y$ denote the transition from $Y^\triangleleft$ to $Y_Y^\dagger$. These two processes are independent by design; further, it has been assumed that $X^\triangleleft$ and $Y^\triangleleft$ are independent in the approximated anterior distribution $\boldsymbol{\psi}^\triangleleft$. Thus, the conditions of Theorem 5.10 apply and the contraction of the process from $\langle X^\triangleleft, Y^\triangleleft \rangle$ to $\langle X_X^\dagger, X_Y^\dagger, Y_Y^\dagger \rangle$ is the minimum of the contractions of $\mathcal{W}_X$ and $\mathcal{W}_Y$. Straightforwardly, the contraction of $\mathcal{W}_Y$ is λ_{Y_Y}. However, $\mathcal{W}_X$ fully contracts, *i.e.*, loses all information about its original state, only when both $X_X^\dagger$ and $X_Y^\dagger$ are in their contraction state. These events are independent, hence the probability that they both occur is the product $\lambda_{X_X}\,\lambda_{X_Y}$.

It is easy to generalize how interconnectivity between the processes degrades the contraction ratio. Consider a subprocess $\mathcal{Q}_l$ whose mixing coefficient is γ_l, and assume that $\mathcal{Q}_l$ depends on r subprocesses $l_1, ..., l_r$. In the above construction, one would have to place a contraction factor λ_{l_i} on each of the r first-stage transitions that lead to

$\mathcal{Q}_l$, in such a way that $1 - \prod_{i=1}^{r}(1 - \lambda_{l_i}) \leq \gamma_l$, which, for equal λ_{l_i}, is achieved optimally by taking $\lambda_{l_i} = 1 - \sqrt[r]{1 - \gamma_l}$, or almost optimally by $\lambda_{l_i} = \gamma_l / r$. Thus, the cost of inward connectivity is a linear reduction of the contraction ratio with the number of incoming influences. The cost of outward connectivity is much higher. Each influence of a subprocess $\mathcal{Q}_l$ on another subprocess involves the construction of another intermediate variable in $\mathcal{W}_{X_l}$, which contracts independently. Since the total contraction of $\mathcal{W}_{X_l}$ is the product of the individual contractions, the cost of outward connectivity is an exponential reduction in the contraction ratio with the number of outgoing influences. This phenomenon has an intuitive explanation: if a process influences many others, it is much less likely that its value will be lost. This analysis is the basis for the following theorem.

Theorem 5.11 *Consider a process $\mathcal{Q}$ consisting of L subprocesses $\mathcal{Q}_1, ..., \mathcal{Q}_L$, and assume that: each subprocess depends on at most r others; each subprocess influences at most q others; and each transition $\mathcal{Q}_l$ has a mixing coefficient $\gamma_l \geq \gamma$, for some $\gamma \geq 0$. Let $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ be distributions over the joint anterior state space of $\mathcal{Q}$, and assume that the states of the individual subprocesses are all independent in $\boldsymbol{\psi}^{\triangleleft}$. Then:*

$$\mathbb{D}_{KL}[\boldsymbol{\varphi}^{\triangleright} \,\|\, \boldsymbol{\psi}^{\triangleright}] \quad \leq \quad (1 - \gamma^*)\, \mathbb{D}_{KL}[\boldsymbol{\varphi}^{\triangleleft} \,\|\, \boldsymbol{\psi}^{\triangleleft}] \ ,$$

where

$$\gamma^* \quad = \quad (\frac{\gamma}{r})^q \ .$$

Proof The proof is based on the following construction. For the purpose of this construction, we treat each process $\mathcal{Q}_i$, $i = 1, ..., L$, as a (time-dependent) random variable, denoted Q_i. Let $Q_i^{\triangleleft}$ and $Q_i^{\triangleright}$ be the corresponding instantaneous anterior and ulterior variables, respectively. Without loss of generality, assume that each of the $\mathcal{Q}_i$ depends on exactly r, and influences exactly q, subprocesses. We show how to construct a two-phase process $\bar{\mathcal{Q}}$ such that $\bar{\mathcal{Q}}[\boldsymbol{\varphi}^{\triangleleft}] = \mathcal{Q}[\boldsymbol{\varphi}^{\triangleleft}]$ and $\bar{\mathcal{Q}}[\boldsymbol{\psi}^{\triangleleft}] = \mathcal{Q}[\boldsymbol{\psi}^{\triangleleft}]$, with the required contraction properties.

The modified process $\bar{\mathcal{Q}}$ is composed of two successive transitions between three layers of variables. The first transition, denoted $\mathcal{Q}^\Gamma$, is defined from a set of L anterior variables $Q_i^{\triangleleft}$ (corresponding exactly to the similarly denoted variables in the original process $\mathcal{Q}$) to a maximum of L^2 intermediate variables denoted $Q_{i,j}^{\dagger}$, for $i, j \in \{1, ..., L\}$. The second transition, denoted $\mathcal{Q}^\Delta$, is defined from the intermediate variables $Q_{i,j}^{\dagger}$ to L ulterior variables $Q_i^{\triangleright}$ (again, mimicking the ulterior variables in the original process $\mathcal{Q}$). The intermediate layer is arranged in such a way that a node $Q_{i,j}^{\dagger}$ has at most a single incoming edge from $Q_i^{\triangleleft}$, and has at most a single outgoing edge to $Q_j^{\triangleright}$, for all $i, j \in \{1, ..., L\}$.

In order to satisfy the stated upper bounds on the in-degree and out-degree of the variables, the out-degree of each $Q_i^{\triangleleft}$ is at most q, and the in-degree of each $Q_j^{\triangleright}$ is at most r. Therefore, at most $q\,r$ of the L^2 intermediate variables actually transmit information from the anterior to the ulterior layer; the remainder of the intermediate variables can be safely deleted. If we denote by $\#\{Q_{i,*}^{\dagger}\}$ and $\#\{Q_{*,j}^{\dagger}\}$ the number of remaining intermediate variables that depend on $Q_i^{\triangleleft}$, and influence $Q_j^{\triangleright}$, respectively, we have the following properties:

$$\begin{aligned} \#\{Q_{i,*}^{\dagger}\} &\leq q \\ \#\{Q_{*,j}^{\dagger}\} &\leq r \ . \end{aligned}$$

We now have to show how to parameterize $\mathcal{Q}^\Gamma$ and $\mathcal{Q}^\Delta$ so that $\bar{\mathcal{Q}}$ as a whole behaves like $\mathcal{Q}$ on $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$. In other words, we need to select the conditional probabilities for the (non deleted) $Q_{i,j}^{\dagger}$ and the $Q_j^{\triangleright}$. Following the discussion preceding this theorem, each intermediate variable $Q_{i,j}^{\dagger}$ is set to transition to an explicit contraction state denoted $\mathbf{c}_{i,j}$, with probability $\lambda_{i,j}$, or to replicate the value of the anterior variable $Q_i^{\triangleleft}$, with probability $1 - \lambda_{i,j}$. The parameters for the ulterior variables $Q_j^{\triangleright}$ are then chosen in a way similar to that in the proof of Theorem 5.6, which is always possible provided that the elementary contraction probabilities $\lambda_{i,j}$ do not force a contraction probability greater than γ, for any process $\mathcal{Q}_j$, *i.e.*, if, $\forall j = 1, ..., L$:

$$1 - \prod_{i:Q_{i,j}^{\dagger} \in \{Q_{*,j}^{\dagger}\}} (1 - \lambda_{i,j}) \ \leq \ \gamma \ .$$

This condition can be satisfied by setting $\lambda_{i,j} = \gamma/r$ across the board.

To conclude, we note that, on the one hand, the anterior variables $Q_i^{\triangleleft}$ are independent in $\boldsymbol{\psi}^{\triangleleft}$, and, on the other hand, the first-phase processes $Q_i^{\triangleleft} \to \langle Q_{i,1}^{\dagger}, ..., Q_{i,L}^{\dagger} \rangle$ are independent of each other. Theorem 5.10 thus applies to the first phase of $\bar{\mathcal{Q}}$, from which the claim follows. $\square$

Thus, if we have a system composed of several sparsely interacting subprocesses, each of which is fairly stochastic, and we use an approximate belief state in which the states of the subprocesses are independent, then our process as a whole contracts at a reasonable rate. We will soon exploit this property to devise an approximate monitoring strategy that achieves both accuracy and efficiency.

Before moving on, one should note that the independence assumption on $\boldsymbol{\varphi}^{\triangleleft}$ is a necessary condition: in particular, relaxing it to conditional independence will not work without additional assumptions. To see this, recall the example of N independent subprocesses from Section 5.3, and observe that even though both distributions $\boldsymbol{\varphi}^{\triangleleft}$ and $\boldsymbol{\psi}^{\triangleleft}$ already satisfied all possible non-trivial conditional independence relations (*i.e.*, for any variables or sets of variables A, B, C, we had that A and B were independent in $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ given C whenever $C \neq \emptyset$), the lack of unconditional independence in $\boldsymbol{\psi}^{\triangleleft}$ was enough to give the whole process an extremely small contraction ratio.

5.3.3 Dealing with partially deterministic processes

As already mentioned in Section 5.2, contraction analyses such as the above will fail to give any useful result if the process contains a "deterministic core", *i.e.*, a pair of states that are mapped to two separate regions of the state space. This may happen even if the process is ergodic, although in this case there is a finite number τ of steps after which any pair of initial states will start to blur. In this case, even if the one-step mixing coefficient of the process is null, watching that same process at a coarser time granularity, by agglomerating τ elementary transitions into a macro-transition, will guarantee that we have non-zero mixing for all pairs of starting states.

Example 5.12 Consider a HMM with four states, given by the following transition matrix:

$$\mathcal{S} \;=\; \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{pmatrix} .$$

It is easy to see that this process has a null mixing coefficient, since two instances starting from the the 1^{st} and 3^{rd} states will end up, respectively, in the 1^{st} or 2^{nd} state, and in the 3^{rd} or 4^{th} state; *i.e.*, there can be no overlap.

By contrast, if we compose the above process with itself over two time steps, we obtain the following transition matrix:

$$\mathcal{S}^2 \;=\; \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{4} & 0 & \frac{1}{4} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{4} & 0 & \frac{1}{4} \end{pmatrix} .$$

The mixing coefficient is now equal to $\frac{1}{2}$.

▟

Although such agglomeration is useful to uncover the mixing properties of an ergodic process with a deterministic core, this comes at a cost. Some dependences between variables that only manifested themselves after a delay in the original transition will now be treated as immediate in the macro-transition. These extra dependences force us to select bigger clusters than we normally would, and are also likely to increase the connectivity between the clusters. In the structured case, these manifestations have a negative impact on the overall mixing coefficient.

5.4 Iterated approximation

It remains to be seen how the above contraction results apply to our inference algorithm. Of prime importance is the relationship between the contraction phenomenon and the accumulation of approximation errors over time. We thus analyze the total error resulting from our approximation strategy, as measured by the divergence from the current true posterior belief state $\boldsymbol{\sigma}^{(t\bullet)}$ to our approximation thereof $\tilde{\boldsymbol{\sigma}}^{(t)}$. Intuitively, this error originates from two sources: the accumulated error inherited from the previous approximation $\tilde{\boldsymbol{\sigma}}^{(t-1)}$, and the fresh error caused by the approximation of $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ as $\tilde{\boldsymbol{\sigma}}^{(t)}$.

Suppose that each such approximation introduces an error, increasing the divergence from the exact belief state to our approximation thereof by ϵ. However, the contraction resulting from propagating the belief states through the stochastic transitions serves to drive them closer to each other, reducing the accumulated error by a fraction γ, as per Theorem 5.11. Conditioning on the various observations moves the two distributions even closer to each other, at least on expectation over the possible responses, in accordance with Fact 5.3. Therefore, the total expected error, as accumulated through t inference steps, would behave as,

$$\epsilon + (1-\gamma)\,\epsilon + (1-\gamma)^2\,\epsilon + \ldots + (1-\gamma)^{t-1}\,\epsilon \;\;\leq\;\; \sum_{i=0}^{\infty} \epsilon\,(1-\gamma)^i \;=\; \frac{\epsilon}{\gamma}\;,$$

and would therefore be bounded independently of t.

To formalize this result, we first need to quantify the incremental error resulting from one approximation step. As the approximate belief state $\tilde{\boldsymbol{\sigma}}^{(t)}$ is an approximation of $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$, most obviously, the incremental approximation error could be defined as $\mathbb{D}_{KL}[\hat{\boldsymbol{\sigma}}^{(t\bullet)}\,\|\,\tilde{\boldsymbol{\sigma}}^{(t)}]$, the relative entropy from $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ to $\tilde{\boldsymbol{\sigma}}^{(t)}$. This relative entropy has the advantage of capturing the intrinsic error caused by the approximation step, in isolation from the context of the true belief state of the process. However, the actual error that matters in an application is measured relative to the true distribution $\boldsymbol{\sigma}^{(t\bullet)}$, and not to $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$. Therefore, we use the following definition.

Definition 5.13 We say that an approximation $\tilde{\boldsymbol{\sigma}}^{(t)}$ to a distribution $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ *incurs error ϵ relative* to the true distribution $\boldsymbol{\sigma}^{(t\bullet)}$, if

$$\epsilon \;\;=\;\; \mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)}\,\|\,\tilde{\boldsymbol{\sigma}}^{(t)}] - \mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)}\,\|\,\hat{\boldsymbol{\sigma}}^{(t\bullet)}]\;.$$

We note that this definition allows the error ϵ to be negative; this would be the case when the approximate distribution $\tilde{\boldsymbol{\sigma}}^{(t)}$ is closer to the true distribution $\boldsymbol{\sigma}^{(t\bullet)}$ than the distribution $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ it approximates.

By induction on t, it now follows easily that, if the assumption holds, the total error remains bounded forever on expectation.

Theorem 5.14 *Let $\mathcal{S}$ be a stochastic process which, together with any fixed approximation scheme, achieves a contraction coefficient of γ. Assume that, at each time slice t, the approximation scheme incurs error at most ϵ relative to the true belief state $\boldsymbol{\sigma}^{(t\bullet)}$. Then, for any t, it holds that:*

$$\mathbb{E}_{\boldsymbol{\rho}^{(1)..(t)}}[\mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \tilde{\boldsymbol{\sigma}}^{(t)}]] \quad \leq \quad \frac{\epsilon}{\gamma} \ ,$$

where the expectation is taken with respect to the distribution $\boldsymbol{\rho}^{(1)..(t)}$ of response sequences $R^{(1)}, ..., R^{(t)}$, with the probability ascribed to them by the process $\mathcal{S}$.

Proof We start by peeling off the expectations over the observed responses from the inside out, going backward in time starting with $\boldsymbol{\rho}^{(t)}$:

$$\mathbb{E}_{\boldsymbol{\rho}^{(1)..(t)}}[\mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \tilde{\boldsymbol{\sigma}}^{(t)}]] \quad = \quad \mathbb{E}_{\boldsymbol{\rho}^{(1)..(t-1)}}[\mathbb{E}_{\boldsymbol{\rho}^{(t)}}[\mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \tilde{\boldsymbol{\sigma}}^{(t)}]]] \ .$$

Focusing on the inner expectation $\mathbb{E}_{\boldsymbol{\rho}^{(t)}}[...]$, successively using (1) the assumed bound on the incurred error, (2) the non-increase of expected KL divergence under conditioning (observing that the distribution of $\boldsymbol{\rho}^{(t)}$ stems from $\boldsymbol{\sigma}^{(\bullet t)}$), and (3) the third is the contraction property, we obtain:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\rho}^{(t)}}[\mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \tilde{\boldsymbol{\sigma}}^{(t)}]] \quad &\leq \quad \epsilon + \mathbb{E}_{\boldsymbol{\rho}^{(t)}}[\mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \hat{\boldsymbol{\sigma}}^{(t\bullet)}]] \\
&\leq \quad \epsilon + \mathbb{D}_{KL}[\boldsymbol{\sigma}^{(\bullet t)} \,\|\, \hat{\boldsymbol{\sigma}}^{(\bullet t)}] \\
&\leq \quad \epsilon + (1-\gamma)\, \mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t-1\bullet)} \,\|\, \tilde{\boldsymbol{\sigma}}^{(t-1)}] \ .
\end{aligned}$$

Proceeding likewise for the remaining time steps, from $t-1$ down to 0 (or, even, to the limit at $-\infty$, observing convergence in the limit), we finally get:

$$\begin{aligned}\boldsymbol{E}_{\boldsymbol{\rho}^{(1)..(t)}}[\mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \tilde{\boldsymbol{\sigma}}^{(t)}]] &\leq \epsilon + (1-\gamma)\,\epsilon + (1-\gamma)^2\,\epsilon + ... + (1-\gamma)^{t-1}\,\epsilon + ... \\ &= \frac{\epsilon}{\gamma}\,,\end{aligned}$$

where we have used the well-known formula for converging geometric series. ◻

Of course, it is not trivial to show that a particular approximation scheme will satisfy the accuracy requirement of ϵ. The main difficulty stems from the fact that the notion of error from Definition 5.13 depends on the true belief state $\boldsymbol{\sigma}^{(t\bullet)}$, which is usually not known, and often unknowable. Nevertheless, it is easy to show that if the maximum over all states of the relative error caused by approximating $\hat{\boldsymbol{\sigma}}^{(t\bullet)}$ by $\tilde{\boldsymbol{\sigma}}^{(t)}$ is bounded at time t, then so is the incurred approximation error ϵ at time t.

Proposition 5.15 *If $\eta^{(t)}$ is such that:*

$$\eta^{(t)} = max_i \frac{\hat{\boldsymbol{\sigma}}^{(t\bullet)}[s_i]}{\tilde{\boldsymbol{\sigma}}^{(t)}[s_i]}\,,$$

then, necessarily:

$$\mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \tilde{\boldsymbol{\sigma}}^{(t)}] - \mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \hat{\boldsymbol{\sigma}}^{(t\bullet)}] \leq log_2\,[\eta^{(t)}]\,.$$

Proof We have:

$$\begin{aligned}\mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \tilde{\boldsymbol{\sigma}}^{(t)}] - \mathbb{D}_{KL}[\boldsymbol{\sigma}^{(t\bullet)} \,\|\, \hat{\boldsymbol{\sigma}}^{(t\bullet)}] &= \boldsymbol{E}_{\boldsymbol{\sigma}^{(t\bullet)}}[log_2 \frac{\boldsymbol{\sigma}^{(t\bullet)}}{\tilde{\boldsymbol{\sigma}}^{(t)}} - log_2 \frac{\boldsymbol{\sigma}^{(t\bullet)}}{\hat{\boldsymbol{\sigma}}^{(t\bullet)}}] \\ &= \boldsymbol{E}_{\boldsymbol{\sigma}^{(t\bullet)}}[log_2 \frac{\hat{\boldsymbol{\sigma}}^{(t\bullet)}}{\tilde{\boldsymbol{\sigma}}^{(t)}}] \\ &\leq max_i log_2 \frac{\hat{\boldsymbol{\sigma}}^{(t\bullet)}[s_i]}{\tilde{\boldsymbol{\sigma}}^{(t)}[s_i]} = log_2\,[\eta^{(t)}]\,.\end{aligned}$$

◻

The expression of $\eta^{(t)}$ above is the maximum relative error caused by the approximation scheme at time t, a value for which it is often easy to assess an upper bound when executing a given approximation step.

It is important to note that Theorem 5.14 only provides a bound on the expected error. The bounds for specific sequences of evidence are much weaker; in particular, the error after a long burst of very unlikely responses might be subtantially larger than the average. Nevertheless, the contraction results hold for arbitrary distributions, no matter how far apart the approximation might be from the reference. Even if $\boldsymbol{\sigma}^{(t\bullet)}$ and $\tilde{\boldsymbol{\sigma}}^{(t)}$ are momentarily quite different, the error will be resorbed by the contraction property at a geometric rate.

5.5 Choosing an approximation

Going back to the experiments of Section 4.3, we now compare how the measured error correlates with the theoretical error bounds of this chapter. In particular, we wish to assess how our theoretical tools predict how well each of the various approximation schemes of Section 4.3 would perform, compared with each other. The two determining factors here are the stepwise approximation error ϵ and the overall mixing coefficient bounded by γ^*. The former is directly linked to the expressiveness of the approximation; so according to this criterion we expect the quality to decrease with the aggressiveness of the belief state clustering. Thus, referring to the approximations used in the BAT network experiments of Section 4.3, we should have, in decreasing order of quality: '5+5' $\gg$ '3+3+4' $\succeq$ '3+2+4+1', although, as mentioned earlier, the actual error incurred at each step depends on the state of the system at that time. The mixing coefficients can be asessed quantitatively, using the results of this chapter. For each clustering as above, we computed the vector $\vec{\gamma}$ of mixing coefficients for all clusters, and used it together with the connectivity characteristics q and r to calculate γ^* from Theorem 5.11.

- For the '5+5' clustering, $\vec{\gamma} = \langle 0.00040, 0.0081 \rangle$, $r = 2$, $q = 2$, hence $\gamma^* = (\gamma_{min}/r)^q = 4 \times 10^{-8}$.

- For the '3+2+4+1' clustering, $\vec{\gamma} = \langle 0.00077, 0.080, 0.0081, 0.96 \rangle$, $r = 3$, $q = 2$, hence $\gamma^* = 7 \times 10^{-8}$.

- For the '3+3+4' clustering, $\vec{\gamma} = \langle 0.0022, 0.020, 0.0034 \rangle$, $r = 3$, $q = 3$, hence $\gamma^* = 4 \times 10^{-10}$.

Thus, for this criterion, we get, by decreasing order of quality, '3+2+4+1' $\simeq$ '5+5' $\gg$ '3+3+4': the latter clustering is heavily penalized by the higher inter-connectivity of its clusters, whereas the former two benefit from having boundaries that align nicely with the natural "islands" of the process, as predicted.

In summary, in order to apply the approximate inference algorithm of Section 4.2 to a particular problem, it is necessary to define a partition of the canonical variables into clusters, *i.e.*, choose a partition of the process into subprocesses. The contraction analysis presented in this chapter can be used to evaluate the alternatives. The tradeoffs, however, are subtle. Subprocesses with a small number of state variables will generally allow more efficient inference; they also have a smaller transition model, and therefore their mixing coefficient is likely to be better. On the other hand, subprocesses need to be large enough to ensure that there are no edges crossing a subprocess boundary within a single time slice; furthermore, making the subprocesses larger will likely decrease the error incurred by this approximation: specifically, if two variables are highly correlated, keeping them together will result in a much improved incremental approximation error. The analysis of the coming chapter gives further insight into the factors that affect the quality of an approximation.

Chapter 6

Projection analysis

Tracking the state of a complex stochastic system, as we saw in the previous chapters, is a delicate task, mainly due to unpredictable dynamics and partial observability. Factored stochastic models such as DBNs provide a coherent framework for modeling such systems, where the state of the system is often represented using a set of state variables. Transition dynamics are represented compactly by exploiting the fact that each variable typically interacts with only a few others. Furthermore, even though all variables are bound to be structurally correlated with each other, only a few of these correlations will be of any perceivable importance, while the others may be neglected to facilitate the reasoning process.

Part of the difficulty of implementing this intuition is to weed out induced weak correlations in the belief state in a way that does not impact the transition model itself. For example, when considering cars on a freeway, it may be perfectly reasonable to represent the instantaneous state of each car separately from the others, using marginals instead of a joint distribution. However, the interactions between cars should still be taken into account when updating the individual beliefs. This is the distinction between tracking the state of the freeway using a decomposed representation of the belief state, and treating the cars as completely independent of one another, which would amount to neglecting all interactions. In other words, while the induced effects of the cross interactions between variables may often be safely neglected *a posteriori*, it is usually dangerous to neglect these interactions *a priori*.

As Herbert Simon argues [Sim62],

> [These] nearly decomposable systems, in which the interactions among the subsystems are weak but not negligible [...] are far from rare.
>
> On the contrary, systems in which each variable is linked with almost equal strength with almost all other parts of the system are far rarer and less typical.

In the previous chapters, we propose an algorithm that exploits this idea by momentarily ignoring the weak correlations between the states of different system components. More precisely, the algorithm represents the belief state over the entire system as a set of localized beliefs about its parts, even though the entire set of interactions present in the model is used to update those beliefs. The analysis shows that the stochasticity of the process prevents the repeated errors resulting from the projection onto localized beliefs at every step from accumulating without bound. However, the overall bound depends on the value of the error incurred by the individual approximation steps, which is difficult to quantify a priori. Rather, the justification is based on the intuition that, if the subprocesses interact only weakly, the error incurred by approximating away the induced correlations has to be small. Or, as stated by Simon [Sim62]: "In a nearly decomposable system, the short-run behavior of each of the component subsystems is approximately independent of the short run behavior of the other components."

In order to make this intuition precise, we must formally define what it means for processes to interact weakly, and show that weak interactions allows us to bound the error introduced by this approximation. We also analyze a new notion of *sparse interaction*, where the usual mode of interaction between subprocesses is weak, albeit tainted by an occasional strong interaction. In this case, the weak interaction assumption is warranted only part of the time, and so is the approximation scheme chosen to monitor the system under that assumption; however, it would be wasteful to uniformly relax a good approximation for the rare times when it is not applicable. To cope with this type of processes, we propose an adaptive approximation, which keeps the states of the interacting subprocesses momentarily coupled, following a strong interaction. As our analysis shows, the coupling needs to persist only shortly after the strong interaction ceases, as the induced correlations fade at a geometric rate.

Thus, whereas the last chapter was mostly concerned with stochasticity being instrumental to the geometric reduction of accumulated past errors, this chapter focuses on the *incremental* short-term approximation errors, and how they are affected by the process structure and the strength of the interactions.

6.1 The myth of structure revisited

In this section, we briefly review our previous results on inference in compactly represented dynamic processes, and cast them in a slightly more general setting, which will be useful for our study of the incremental error.

A stochastic dynamic system is defined via a set of states, and a transition model that represents the way in which one state leads to the next. In complex systems, a state is best described using a set of random variables $\mathbf{A}_1, ..., \mathbf{A}_n$. The transition model is described via a 2-TBN $\mathcal{B}$. The network contains anterior nodes $A_1^{\triangleleft}, ..., A_n^{\triangleleft}$ representing the previous state, and ulterior nodes $A_1^{\triangleright}, ..., A_n^{\triangleright}$ representing the present state. Each ulterior node $A_i^{\triangleright}$ has a set of parents $Pa[A_i^{\triangleright}]$; anterior nodes $A_i^{\triangleleft}$ have no parents. The network graph represents the qualitative structure of the transition model, by indicating the anterior and ulterior variables that directly influence the new value of each ulterior variable $A_i^{\triangleright}$. The transition model is made quantitative by associating with each ulterior variable $A_i^{\triangleright}$ a conditional probability distribution $\mathbb{P}[A_i^{\triangleright} | Pa[A_i^{\triangleright}]]$.

Recall that, as the system state is not directly observable, its evolution is tracked using a belief state $\boldsymbol{\sigma}^{(t)}$ over the possible states at the current time t. In principle, maintaining $\boldsymbol{\sigma}^{(t)}$ is straightforward, as described previously. Having computed $\boldsymbol{\sigma}^{(t-1)}$, we propagate it forward using the transition model to obtain the expected next belief state $\boldsymbol{\sigma}^{(\bullet t)}$; we then condition $\boldsymbol{\sigma}^{(\bullet t)}$ on the system response at time t to get $\boldsymbol{\sigma}^{(t)}$.

In Chapter 4, we propose a scalable approximation to this procedure. Algorithms 4.5 and 4.8 maintain approximate belief states that admit a factored representation. Specifically, either variant of the algorithm only considers belief states that fall into some restricted family of distributions Ξ, *e.g.*, ones where certain sets of variables are marginally independent (in the case of Algorithm 4.5) or conditionally independent (in the case of Algorithm 4.8). Let $\boldsymbol{\vartheta}^{(t-1)} \in \Xi$ be the most recently computed approximation to the belief state. Updating it to the next time slice would produce a distribution $\tilde{\boldsymbol{\vartheta}}^{(t)}$ which would typically need to be projected back into Ξ; our algorithm exploits the structure of the process to combine both operations, producing $\boldsymbol{\vartheta}^{(t)}$ in one step.

In this chapter, we seek to refine our analysis of the strength of the various interactions and how it affects the induced correlations. Often, a complex process is architectured as a hierarchy of simpler processes, so that interactions tend to be stronger between the subprocesses that are part of the same substructure. To capture this intuition, we will introduce quantitative notions of weak and sparse interactions, which are based on a hierarchical decomposition of the processes. We will therefore find it useful to consider the inference procedure of Algorithm 4.8, and give a hierarchical process interpretation to the various clusters that appear in the cluster forest selected for the algorithm. We now review a few key aspects of the workings of that algorithm.

Definition 6.1 Given a cluster forest $\mathcal{F}$ over a set of BN or DBN nodes $\{A_1, ..., A_n\}$ (as defined in Definition 4.6), we define $\Xi[\mathcal{F}]$ to be the set of distributions $\boldsymbol{\varphi}$ that are representable over $\mathcal{F}$ (as defined in Definition 4.7).

Algorithm 4.5 takes the approximate belief state $\boldsymbol{\vartheta}^{(t-1)}$ in $\Xi[\mathcal{F}]$, and generates the approximate belief state $\boldsymbol{\vartheta}^{(t)}$ in $\Xi[\mathcal{F}]$, according to an idealized sequence of steps, as follows. In the first phase, the algorithm propagates $\boldsymbol{\vartheta}^{(t-1)}$ to $\hat{\boldsymbol{\vartheta}}^{(t)}$ using the transition model. It then conditions $\hat{\boldsymbol{\vartheta}}^{(t)}$ on the time-t evidence, generating $\tilde{\boldsymbol{\vartheta}}^{(t)}$. Finally, it projects $\tilde{\boldsymbol{\vartheta}}^{(t)}$ into $\Xi[\mathcal{F}]$, resulting in $\boldsymbol{\vartheta}^{(t)}$.

In order for this process to be performed correctly, we require that, if $A_k^{\triangleright} \in Pa[A_l^{\triangleright}]$, *i.e.*, $A_l^{\triangleright}$ has a parent $A_k^{\triangleright}$ in its own time slice, then there must be some cluster F_i such that $A_k^{\triangleright}$ and $A_l^{\triangleright}$ are both in F_i. In other words, it is required that all intra-time-slice edges be contained in some cluster. This assumption allows us to focus attention on inter-time-slice influences. We therefore define the ancestry of an ulterior variable $A_l^{\triangleright}$ to be the set of its parents restricted to the previous time slice, namely $Pa^*[A_l^{\triangleright}] = Pa[A_l^{\triangleright}] \setminus \{A_1^{\triangleright}, ..., A_n^{\triangleright}\}$. The ancestry of an ulterior cluster or set of variables is defined as the union of the individual ancestries, in the obvious way.

The potential problem with this approach is that the repeated approximation at every time slice t could cause errors to accumulate without bound, resulting in a meaningless approximation. In the previous chapter, we provide conditions under which the error remains bounded. The first condition is that the process is sufficiently stochastic, so that the accumulated errors from the past are progressively forgotten. The second condition is that each approximation step does not introduce an excessive amount of fresh error. The rate of decay of past errors depends on the mixing properties of the process, which are most generally characterized as follows.

Definition 6.2 Let $Z^{\triangleright}$ be an ulterior cluster with respect to a 2-TBN model, and $X^{\triangleleft}$ be $Pa^*[Z^{\triangleright}]$. We define the *mixing coefficient* of the stochastic transition $X^{\triangleleft} \to Z^{\triangleright}$ as:

$$\gamma[X^{\triangleleft} \to Z^{\triangleright}] \quad = \quad min_{x_1,x_2} \sum_z min\{\mathbb{P}[Z^{\triangleright} = z | X^{\triangleleft} = x_1], \mathbb{P}[Z^{\triangleright} = z | X^{\triangleleft} = x_2]\} \ .$$

Letting $Y^{\triangleleft}$ and $W^{\triangleleft}$ disjoint such that $X^{\triangleleft} = Y^{\triangleleft} \cup W^{\triangleleft}$, we also define the *mixing coefficient* of the conditional transition $Y^{\triangleleft} \to Z^{\triangleright}$ as the minimal mixing coefficient of the instances of that transition obtained by conditioning over all possible values of $W^{\triangleleft}$:

$$\gamma[Y^{\triangleleft} \to Z^{\triangleright} | W^{\triangleleft}] \quad = \quad min_w min_{y_1,y_2} \sum_z min\{\mathbb{P}[z|y_1,w], \mathbb{P}[z|y_2,w]\} \ .$$

Intuitively, the now familiar mixing coefficient is the minimal amount of mass that two distributions over $Z^{\triangleright}$ are guaranteed to have in common: one is the distribution we would get starting at x_1 and the other the one starting at x_2. The mixing coefficient in the conditional transition is similar, except that the starting points are restricted to be already in agreement about $W^{\triangleleft}$. From here on, we will often drop the explicit reference to $W^{\triangleleft}$ in the notation for mixing coefficient, in which case $W^{\triangleleft}$ is understood to be $Pa^*[Z^{\triangleright}] \setminus Y^{\triangleleft}$. It is noted that the mixing coefficient of compound transitions can be easily bounded in terms of the conditional distributions of individual variables:

Proposition 6.3 *Let* $Z^{\triangleright} = \cup_{i=1}^{k} Z_i^{\triangleright}$*; let* $X_i^{\triangleleft} \subseteq \mathrm{Pa}^*[Z_i^{\triangleright}]$*, and* $X^{\triangleleft} = \cup_{i=1}^{k} X_i^{\triangleleft}$*. Then the minimal mixing coefficient of the compound transition* $X^{\triangleleft} \to Z^{\triangleright}$ *satisfies the inequality:*

$$\gamma[X^{\triangleleft} \to Z^{\triangleright}] \;\; \geq \;\; \prod_{i=1}^{k} \gamma[X^{\triangleleft} \to Z_i^{\triangleright}] \; . \tag{6.1}$$

If furthermore each $X_i^{\triangleleft}$ *is disjoint from* $\mathrm{Pa}^*[Z_j^{\triangleright}]$ *for all* $j \neq i$*, then:*

$$\gamma[X^{\triangleleft} \to Z^{\triangleright}] \;\; \geq \;\; \prod_{i=1}^{k} \gamma[X_i^{\triangleleft} \to Z_i^{\triangleright}] \; . \tag{6.2}$$

Proof We first observe that, since $Z_j^{\triangleright} \notin Pa^*[Z_i^{\triangleright}]$, for $i \neq j$, we have:

$$\boldsymbol{P}[Z_i^{\triangleright} = z_i | X^{\triangleleft} = x, Z_j^{\triangleright} = z_j] \;\; = \;\; \boldsymbol{P}[Z_i^{\triangleright} = z_i | X^{\triangleleft} = x]$$

Regarding Equation 6.1, we then obtain, starting from Definition 6.2:

$$\begin{aligned}
\gamma[X^{\triangleleft} \to Z^{\triangleright}] \;\; &= \;\; min_{x,x'} \sum_{z_1,\ldots,z_k} min\{\prod_{i=1}^{k} \boldsymbol{P}[Z_i^{\triangleright} = z_i | X^{\triangleleft} = x], \prod_{i=1}^{k} \boldsymbol{P}[Z_i^{\triangleright} = z_k | X^{\triangleleft} = x']\} \\
&\geq \;\; min_{x,x'} \sum_{z_1,\ldots,z_k} \prod_{i=1}^{k} min\{\boldsymbol{P}[Z_i^{\triangleright} = z_i | X^{\triangleleft} = x], \boldsymbol{P}[Z_i^{\triangleright} = z_k | X^{\triangleleft} = x']\} \\
&= \;\; min_{x,x'} \prod_{i=1}^{k} \sum_{z_i} min\{\boldsymbol{P}[Z_i^{\triangleright} = z_i | X^{\triangleleft} = x], \boldsymbol{P}[Z_i^{\triangleright} = z_k | X^{\triangleleft} = x']\} \\
&\geq \;\; \prod_{i=1}^{k} min_{x,x'} \sum_{z_i} min\{\boldsymbol{P}[Z_i^{\triangleright} = z_i | X^{\triangleleft} = x], \boldsymbol{P}[Z_i^{\triangleright} = z_k | X^{\triangleleft} = x']\} \\
&= \;\; \prod_{i=1}^{k} \gamma[X^{\triangleleft} \to Z_i^{\triangleright}] \; .
\end{aligned}$$

As for Equation 6.2, it follows from a simple decomposition of the process into independent subprocesses. ◻

As shown in the previous chapter, the mixing coefficient can be used to bound the rate at which errors arising from approximations in the past are forgotten. Let $\{S_1, ..., S_l\}$ be the finest disjoint partition of $\{A_1, ..., A_n\}$ such that each cluster F_i is fully contained in some S_j; *i.e.*, each S_j is one of the connected components in the cluster forest defined by $\mathcal{F}$.

Let r be the maximum inward connectivity of the process relative to the partition, *i.e.*, an upper bound, over all components, on the number of components $S_i^{\triangleleft}$ such that there is an edge from one of $S_i^{\triangleleft}$'s variables to one of $S_j^{\triangleright}$'s variables. Similarly, let q be the maximum outward connnectivity of the process relative to the partition, *i.e.*, an upper bound, over all components, on the number of $S_j^{\triangleright}$ such that there is an edge from $S_i^{\triangleleft}$ to $S_j^{\triangleright}$. Defining γ^*, the (partition dependent) overall contraction ratio of the process,

$$\gamma^* \quad = \quad \left(\frac{1}{r}\, min_l \gamma[S_l^{\triangleleft} \to S_l^{\triangleright}]\right)^q \quad ,$$

we have the contraction property from Theorem 5.11,

$$\boldsymbol{D}_{KL}[\boldsymbol{\sigma}^{(\bullet t)} \,\|\, \hat{\boldsymbol{\vartheta}}^{(t)}] \quad \leq \quad (1-\gamma^*)\, \boldsymbol{D}_{KL}[\boldsymbol{\sigma}^{(t-1)} \,\|\, \boldsymbol{\vartheta}^{(t-1)}] \ .$$

As for the fresh error introduced at every approximation step, defining $\epsilon_{\boldsymbol{\mu}}(\boldsymbol{\varphi} \mapsto \boldsymbol{\psi})$ to be the incurred error of using $\boldsymbol{\psi}$ as an approximation of $\boldsymbol{\varphi}$, with respect to a true distribution $\boldsymbol{\mu}$,

$$\epsilon_{\boldsymbol{\mu}}(\boldsymbol{\varphi} \mapsto \boldsymbol{\psi}) \quad = \quad \boldsymbol{E}_{\boldsymbol{\mu}}[log_2 \frac{\boldsymbol{\varphi}}{\boldsymbol{\psi}}] \ ,$$

and letting ϵ^* be an upper bound on the incurred projection error at all times for the process of interest,

$$\epsilon^* \quad \leq \quad \epsilon_{\boldsymbol{\sigma}^{(t)}}(\tilde{\boldsymbol{\vartheta}}^{(t)} \mapsto \boldsymbol{\vartheta}^{(t)}) \ ,$$

the total expected error remains bounded as follows,

$$\boldsymbol{E}[\boldsymbol{D}_{KL}[\boldsymbol{\sigma}^{(t)} \,\|\, \boldsymbol{\vartheta}^{(t)}]] \quad \leq \quad \epsilon^*/\gamma^* \ .$$

6.2 Intrinsic measures of interaction

Chapter 5 provides an analysis for the contraction ratio γ^*, allowing it to be bounded in terms of parameters of the dynamic system. There is no similar analysis for the implicit approximation error ϵ^*. Rather, it is hypothesized that an approximation that places two subprocesses in different clusters does not incur an exceedingly large incremental error, insofar as the subprocesses do not interact very strongly; indeed, the experimental results of Section 4.3 support this prediction. One of the goals of this chapter is to provide a foundation for such a justification, and relate the approximation error to the structure of the process.

The problem with analyzing the implicit error is, as its name suggests, that it is rather elusive: without knowing the true distribution $\boldsymbol{\sigma}^{(t)}$, the error incurred by the projection of $\tilde{\boldsymbol{\vartheta}}^{(t)}$ to $\boldsymbol{\vartheta}^{(t)}$ cannot be measured. Instead, we will focus on a more manageable, albeit closely related quantity: the intrinsic error of approximating $\tilde{\boldsymbol{\vartheta}}^{(t)}$ using $\boldsymbol{\vartheta}^{(t)}$, as measured by the KL divergence $\boldsymbol{D}_{KL}[\tilde{\boldsymbol{\vartheta}}^{(t)} \,\|\, \boldsymbol{\vartheta}^{(t)}]$, which intrinsically characterizes the error caused by the approximation. The foregoing analysis studies how this intrinsic error is affected by the system structure and dynamics.

Definition 6.4 We define the intrinsic projection error, or simply, *projection error*, of approximating $\boldsymbol{\varphi}$ by $\boldsymbol{\psi}$ as:

$$\begin{aligned} \epsilon[\boldsymbol{\varphi} \mapsto \boldsymbol{\psi}] &= \boldsymbol{D}_{KL}[\boldsymbol{\varphi} \,\|\, \boldsymbol{\psi}] \\ &= \boldsymbol{E}_{\boldsymbol{\varphi}}[log_2 \frac{\boldsymbol{\varphi}[s]}{\boldsymbol{\psi}[s]}] \ . \end{aligned}$$

We now show that the projection error ϵ can be decomposed according to the clustering. The key aspect of this analysis is based on a close relation between the projection error and the mutual information (see Section 2.2) between clusters in the cluster forest $\mathcal{F}$. More precisely, the decomposition relies on an auxiliary structure related to $\mathcal{F}$, which defines a hierarchical grouping of the clusters of $\mathcal{F}$ in a binary tree.

Definition 6.5 Given a cluster forest $\mathcal{F}$, we say that $\mathcal{G}$ is a *cluster hierarchy* of $\mathcal{F}$, when $\mathcal{G}$ is a binary tree of *cluster groups* $\{G_i\}$, or simply groups, satisfying the following conditions:

1. the cluster groups of the leaf nodes are defined as the clusters $F_1, \ldots, F_m$ of $\mathcal{F}$, exactly and uniquely one per leaf node;
2. the cluster groups of the interior nodes are defined as the union of the groups of their two children;
3. for every pair of *sibling groups*, *i.e.*, groups sharing the same parent in $\mathcal{G}$, there is at most one edge in $\mathcal{F}$ that connects a cluster in one group to a cluster in the other group.

We use S_G to denote the set of the $m-1$ undirected pairs $\{G_i, G_j\}$ such that G_i and G_j are siblings. For $\{G_i, G_j\} \in S_G$, we denote by $G_{i\cap j}$ the intersection $G_i \cap G_j$, and by $G_{i\setminus j}$ the difference $G_i \setminus G_j$.

Intuitively, $\mathcal{G}$ is a recursive partition of $\mathcal{F}$, where each split divides up the clusters of a group G_k into a pair of complementary sub-groups G_i and G_j, so that no more than one edge of $\mathcal{F}$ crosses the partition. The crossing edge, if any, corresponds to the intersection of two overlapping clusters in $\mathcal{F}$, *i.e.*, clusters that share some of their variables; the intersection $G_{i\cap j}$ will thus be the set of those variables. Figure 6.1 shows one possible group hierarchy for a given cluster forest.

It is important to emphasize the distinction between the cluster forest $\mathcal{F}$ and the group hierarchy $\mathcal{G}$. The former has a material effect, as it defines the approximation scheme used by the inference algorithm. On the other hand, the group hierarchy $\mathcal{G}$ is merely used for the purpose of analysis, and can be chosen freely given the constraints once $\mathcal{F}$ is fixed. The nature of $\mathcal{F}$ and $\mathcal{G}$ is also different: the clusters in $\mathcal{F}$ may or may not be overlapping, and if they are, they must satisfy the running intersection property; in contrast, some groups in $\mathcal{G}$ are necessarily overlapping, since

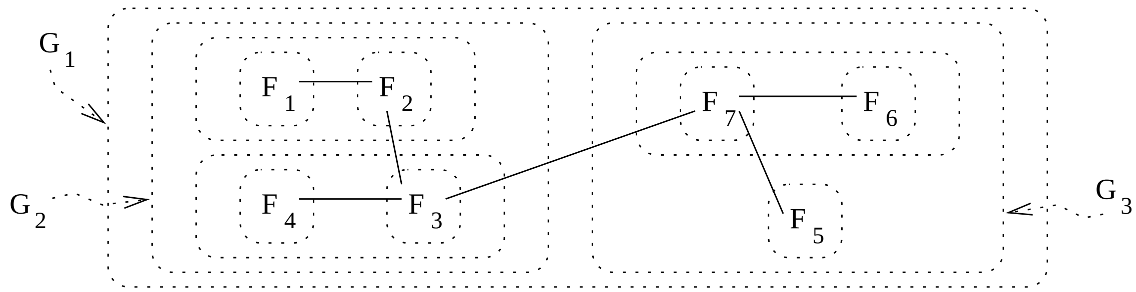


Figure 6.1: Cluster forest and group hierarchy: (solid lines) a cluster forest composed of 7 clusters $F_1, ..., F_7$; (dotted boxes) the 13 groups $G_1, G_2, G_3, ...$ composing one of the possible group hierarchies compatible with the cluster forest.

each group is a proper subset of its parent in the hierarchy. The key insight is that the approximation error for using the clusters in $\mathcal{F}$ decomposes nicely according to the structure of $\mathcal{G}$, as we now show.

Theorem 6.6 *Let* $\mathbf{U}$ *be a set of discrete random variables,* $\mathcal{F}$ *a cluster forest over them, and* $\mathcal{G}$ *a cluster hierarchy as in Definition 6.5. Let* $\boldsymbol{\varphi}$ *be a probability distribution over* $\mathbf{U}$*, and* $\boldsymbol{\psi}$ *its projection onto* $\mathcal{F}$*. Then, the projection error admits the following decomposition:*

$$\epsilon[\boldsymbol{\varphi} \mapsto \boldsymbol{\psi}] \quad = \sum_{\{G_i, G_j\} \in S_G} \mathbb{I}[G_i; G_j | G_{i \cap j}] \ ,$$

where the conditional mutual information under the summation is computed with respect to $\boldsymbol{\varphi}$*.*

Proof The theorem follows easily from an induction argument. Let G_k be any interior node, and denote by G_i and G_j the children of G_k. Since $G_{i \cap j}$ is fully contained in some cluster F_l of $\mathcal{F}$, we have that $\boldsymbol{\psi}[G_{i\cap j}] = \boldsymbol{\varphi}[G_{i\cap j}]$, and, consequently,

$$\begin{aligned} & \mathbb{D}_{KL}[\boldsymbol{\varphi}[G_k] \,||\, \boldsymbol{\psi}[G_k]] \\ = \quad & \mathbb{E}_{\boldsymbol{\varphi}[G_k]} log_2 \frac{\boldsymbol{\varphi}[G_k]}{\boldsymbol{\psi}[G_{i\cap j}] \; \boldsymbol{\psi}[G_i | G_{i\cap j}] \; \boldsymbol{\psi}[G_j | G_{i\cap j}]} \end{aligned}$$

$$
\begin{aligned}
= \quad & \mathbb{E}_{\boldsymbol{\varphi}[G_k]}\Bigg[\, log_2 \frac{\boldsymbol{\varphi}[G_k]}{\boldsymbol{\varphi}[G_{i\cap j}]\, \boldsymbol{\varphi}[G_i|G_{i\cap j}]\, \boldsymbol{\varphi}[G_j|G_{i\cap j}]} \\
& \quad + log_2 \frac{\boldsymbol{\varphi}[G_{i\cap j}]\, \boldsymbol{\varphi}[G_i|G_{i\cap j}]}{\boldsymbol{\psi}[G_{i\cap j}]\, \boldsymbol{\psi}[G_i|G_{i\cap j}]} \\
& \quad + log_2 \frac{\boldsymbol{\varphi}[G_{i\cap j}]\, \boldsymbol{\varphi}[G_j|G_{i\cap j}]}{\boldsymbol{\psi}[G_{i\cap j}]\, \boldsymbol{\psi}[G_j|G_{i\cap j}]}\Bigg] \\
= \quad & \mathbb{E}_{\boldsymbol{\varphi}[G_i,G_j]} log_2 \frac{\boldsymbol{\varphi}[G_i, G_j]}{\boldsymbol{\varphi}[G_{i\cap j}]\, \boldsymbol{\varphi}[G_i|G_{i\cap j}]\, \boldsymbol{\varphi}[G_j|G_{i\cap j}]} \\
& + \mathbb{E}_{\boldsymbol{\varphi}[G_i]} log_2 \frac{\boldsymbol{\varphi}[G_{i\cap j}]\, \boldsymbol{\varphi}[G_i|G_{i\cap j}]}{\boldsymbol{\psi}[G_{i\cap j}]\, \boldsymbol{\psi}[G_i|G_{i\cap j}]} \\
& + \mathbb{E}_{\boldsymbol{\varphi}[G_j]} log_2 \frac{\boldsymbol{\varphi}[G_{i\cap j}]\, \boldsymbol{\varphi}[G_j|G_{i\cap j}]}{\boldsymbol{\psi}[G_{i\cap j}]\, \boldsymbol{\psi}[G_j|G_{i\cap j}]} \\
= \quad & \mathbb{I}[G_i; G_j|G_{i\cap j}] \\
& \quad + \mathbb{D}_{KL}[\boldsymbol{\varphi}[G_i] \,\|\, \boldsymbol{\psi}[G_i]] + \mathbb{D}_{KL}[\boldsymbol{\varphi}[G_j] \,\|\, \boldsymbol{\psi}[G_j]]\ .
\end{aligned}
$$

Thus, the error term restricted to the variables in G_k decomposes into a pair of analogous error terms over the children of G_k, plus the mutual information between those children given their intersection. The claim follows by recursion on the error terms in G_i and G_j, until G_i and G_k reduce to single clusters of $\mathcal{F}$; indeed, since all clusters are atomic with respect to the projection of $\boldsymbol{\varphi}$ into $\boldsymbol{\psi}$, $\mathbb{D}_{KL}[\boldsymbol{\varphi}[F_l] \,\|\, \boldsymbol{\psi}[F_l]] = 0$, for any cluster F_l. To conclude, we note that $\epsilon[\boldsymbol{\varphi} \mapsto \boldsymbol{\psi}] = \mathbb{D}_{KL}[\boldsymbol{\varphi}[G_0] \,\|\, \boldsymbol{\psi}[G_0]]$, where $G_0 = \mathbf{U}$ is the root of the group hierarchy. ◻

The key to exploiting this decomposition is as follows. Recall that $\hat{\boldsymbol{\vartheta}}^{(t)}$ is obtained from propagating a distribution $\boldsymbol{\vartheta}^{(t-1)}$, where the distribution $\boldsymbol{\vartheta}^{(t-1)}$ is in the restricted space $\Xi[\mathcal{F}]$, *i.e.*, it satisfies the independence assumptions defined by $\mathcal{F}$. Intuitively, there should be a limit to the number of dependencies introduced by a factored stochastic transition on a factored distribution, which should result in bounds on each of the mutual information terms that appear in the theorem. The coming sections will be devoted to formalizing this intuition.

It turns out that the notion of mixing coefficient, which captures the extent to which information is retained from a set of anterior variables $X^{\triangleleft}$ to a set of ulterior variables $Y^{\triangleright}$ through a stochastic transformation, can also be viewed as representing the extent to which a set of variables influences another. In particular, we are interested in the extent to which one group G_i at time $t-1$ influences another group G_j at time t. We therefore introduce γ_{ij}, based on the conditional mixing coefficient of Definition 6.2:

$$\gamma_{ij} \quad = \quad \gamma[(G_i^{\triangleleft} \setminus G_{i\cap j}^{\triangleleft}) \rightarrow (G_j^{\triangleright} \setminus G_{i\cap j}^{\triangleright})] \ ,$$

where, as in the Definition 6.2, the dependence on other parents of $G_j^{\triangleright}$ is implicit.

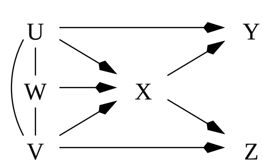
In the following sections, we study the connection between intrinsic errors and mixing properties for the various components of the process.

6.3 Weakly interacting processes

The first simple scenario we study is that of two groups taken by the approximation to be completely independent, *i.e.*, groups contained in different connected components of the cluster forest. Intuitively, the projection error in this case would be expected to depend on the extent to which these two groups interact. In particular, if the system is such that the variables interact only weakly across the two groups, the error incurred by keeping their belief states independent should be small.

We first state a central lemma for this kind of analysis.

Lemma 6.7 *Let $\mathcal{P}$ be a probability distribution over six sets of random variables, denoted U, V, W, X, Y, Z, with the following dependency structure:*



i.e., such that:

$$\mathcal{P}[U,V,W,X,Y,Z] \quad = \quad \mathcal{P}[U,V,W]\,\mathcal{P}[X\,|U,V,W]\,\mathcal{P}[Y\,|X,U]\,\mathcal{P}[Z|X,V] \ .$$

Then, writing γ_{XY} as an abbreviation for $\gamma[X \to Y]$, etc., it holds that:

$$\begin{aligned} \boldsymbol{I}[Y; Z|W] &\leq (1-\gamma_{UY})\,(1-\gamma_{VZ})\,\boldsymbol{I}[U; V|W] \\ &\quad + (2-\gamma_{XY}-\gamma_{XZ}) \log_2 [\#X] \ , \end{aligned}$$

where the mutual informations are computed with respect to $\mathcal{P}$.

The proof of this lemma makes use of the two following claims.

Claim 6.8 *Let $\mathcal{P}[Y|U, X]$ be a conditional probability distribution with mixing coefficients $\gamma[U \to Y|X] = \gamma_{UY}$, $\gamma[X \to Y|U] = \gamma_{XY}$, and $\gamma[(U, X) \to Y] = \gamma_Y$. Also let $\boldsymbol{\varphi}[U, X]$ and $\boldsymbol{\psi}[U, X]$ be two arbitrary distributions over U and X. Then, there exists a conditional distribution $\mathcal{Q}[A, Y', Y'', Y|U, X]$, defined over three additional random variables A, Y', Y'', such that:*

1. *$\mathcal{Q}$ can be factored as follows:*

$$\begin{aligned} &\mathcal{Q}[A, Y', Y'', Y|U, X] \\ &\quad = \ \mathcal{Q}[A]\, \mathcal{Q}[Y'|A, U]\, \mathcal{Q}[Y''|A, X]\, \mathcal{Q}[Y|Y', Y''] \ . \end{aligned}$$

2. *The domain of Y' replicates the domain of U, augmented with one additional state $\mathbf{c}_U$, the contraction state. Similarly, the domain of Y'' replicates that of X with one additional contraction state $\mathbf{c}_X$. Each of Y' and Y'' either copies the value of its parent, or contracts to $\mathbf{c}_U$ or $\mathbf{c}_X$, depending on the value of A. A randomly takes one of four values, respectively triggering the contraction for Y', Y'', both, or none; these outcomes are distributed with probabilities:*

$$\begin{aligned} \mathcal{Q}[A = \text{“both } Y' \text{ and } Y'' \text{ contract”}] &= \gamma_Y \ , \\ \mathcal{Q}[A = \text{“only } Y' \text{ contracts”}] &= (\gamma_{UY} - \gamma_Y) \ , \\ \mathcal{Q}[A = \text{“only } Y'' \text{ contracts”}] &= (\gamma_{XY} - \gamma_Y) \ , \\ \mathcal{Q}[A = \text{“neither } Y' \text{ nor } Y'' \text{ contracts”}] &= (1 - \gamma_{UY} - \gamma_{XY} + \gamma_Y) \ . \end{aligned}$$

3. *The conditional distribution $\mathcal{Q}$ defines a stochastic transition that maps $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ to the same distributions of Y as $\mathcal{P}$ does. Formally:*

$$\begin{aligned} \bar{\boldsymbol{\varphi}}[Y] &= \sum_{u,x} \boldsymbol{\varphi}[u,x]\, \mathcal{P}[Y|u,x] \\ &= \sum_{u,x} \boldsymbol{\varphi}[u,x]\, \mathcal{Q}[Y|u,x] \ , \end{aligned}$$

$$\begin{aligned} \bar{\boldsymbol{\psi}}[Y] &= \sum_{u,x} \boldsymbol{\psi}[u,x]\, \mathcal{P}[Y|u,x] \\ &= \sum_{u,x} \boldsymbol{\psi}[u,x]\, \mathcal{Q}[Y|u,x] \ . \end{aligned}$$

Proof The argument is a generalization of Lemma 5.5 in Section 5.2.

Since the transition $\langle U, X\rangle \to Y$ has mixing coefficient γ_Y, at least a fraction γ_Y of the mass of $\bar{\boldsymbol{\varphi}}[Y]$ and $\bar{\boldsymbol{\psi}}[Y]$ is distributed identically. Let $\eta[Y]$ be the measure of this mass in agreement (it sums up to γ_Y, and always has a solution, which need not be uniquely determined). Thus, $\bar{\boldsymbol{\varphi}}[Y]$ and $\bar{\boldsymbol{\psi}}[Y]$ both contain a contribution $\eta[Y]$, which is independent of U and X under the assumption that $\langle U, X\rangle$ is distributed per either $\boldsymbol{\varphi}$ or $\boldsymbol{\psi}$. To capture the contribution of η in the new model, we let Y' and Y'' simultaneously enter their respective contraction states $\mathbf{c}_U$ and $\mathbf{c}_X$ with probability γ_Y, and we define the distribution of $\{Y|Y' = \mathbf{c}_U, Y'' = \mathbf{c}_X\}$ to be $\eta[Y]/\gamma_Y$.

Similarly, since the conditional transformation $U \to Y|X$ has mixing coefficient γ_{UY}, we have that $\bar{\boldsymbol{\varphi}}[Y]$ and $\bar{\boldsymbol{\psi}}[Y]$ must agree for an additional fraction $(\gamma_{UY} - \gamma_Y)$ of their mass when X is fixed. We collect this additional contribution, whose distribution depends on X or equivalently on Y'', in the distributions of $\{Y|Y' = c, Y'' = x\}$ for all $x \in dom[X]$; as for Y', we let it enter its contraction state alone with probability $(\gamma_{UY} - \gamma_Y)$. The case where Y'' but not Y' contracts is analogous.

Finally, the remainder of the mass, which can be shown to be non-negative everywhere, is collected is the conditional distribution of $\{Y|Y', Y''\}$ for $Y' \neq c$ and $Y'' \neq c$. ◻

Claim 6.9 *In the setting of Claim 6.8 above, the following contraction property holds:*

$$\begin{aligned}
&\boldsymbol{D}_{KL}[\bar{\varphi}[Y] \,\|\, \bar{\boldsymbol{\psi}}[Y]] \\
&\quad \leq \quad (1-\gamma_{UY})\, \boldsymbol{D}_{KL}[\varphi[U] \,\|\, \boldsymbol{\psi}[U]] + (1-\gamma_{XY})\, \boldsymbol{E}_{\varphi[U]}[\boldsymbol{D}_{KL}[\varphi[X|U] \,\|\, \boldsymbol{\psi}[X|U]]] \ .
\end{aligned}$$

Proof Let $\mathcal{Q}$ be the transition model as constructed in Claim 6.8. We know that $\mathcal{Q}$ maps $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ respectively to $\bar{\varphi}$ and $\bar{\boldsymbol{\psi}}$. For notational convenience, let us extend the distributions $\bar{\boldsymbol{\varphi}}[Y]$ and $\bar{\boldsymbol{\psi}}[Y]$ over U, X, A, Y', Y'', Y, the obvious way:

$$\begin{aligned}
\bar{\boldsymbol{\varphi}}[U,X,A,Y',Y'',Y] &= \boldsymbol{\varphi}[U,X] \otimes \mathcal{Q}[A,Y',Y'',Y|U,X] \ , \\
\bar{\boldsymbol{\psi}}[U,X,A,Y',Y'',Y] &= \boldsymbol{\psi}[U,X] \otimes \mathcal{Q}[A,Y',Y'',Y|U,X] \ .
\end{aligned}$$

By construction of $\mathcal{Q}$, the only variables Y directly depends upon are Y' and Y''. Therefore, the transition $\langle Y', Y''\rangle \to Y$ is Markovian, and we get, taking all expectations relative to $\bar{\boldsymbol{\varphi}}$:

$$\begin{aligned}
&\boldsymbol{D}_{KL}[\bar{\varphi}[Y] \,\|\, \bar{\boldsymbol{\psi}}[Y]] \\
&\quad \leq \quad \boldsymbol{D}_{KL}[\bar{\boldsymbol{\varphi}}[Y'] \,\|\, \bar{\boldsymbol{\psi}}[Y']] + \boldsymbol{E}_{Y'}[\boldsymbol{D}_{KL}[\bar{\varphi}[Y''|Y'] \,\|\, \bar{\boldsymbol{\psi}}[Y''|Y']]] \ .
\end{aligned}$$

For convenience, define A' and A'' to be the functions of A respectively indicating whether Y' and Y'' are entering their contraction states. The first term appearing in the above expression can be bounded as follows:

$$\begin{aligned}
&\boldsymbol{D}_{KL}[\bar{\boldsymbol{\varphi}}[Y'] \,\|\, \bar{\boldsymbol{\psi}}[Y']] \\
&\quad \leq \quad \boldsymbol{D}_{KL}[\bar{\boldsymbol{\varphi}}[A',Y'] \,\|\, \bar{\boldsymbol{\psi}}[A',Y']] \\
&\quad = \quad \boldsymbol{D}_{KL}[\bar{\boldsymbol{\varphi}}[A'] \,\|\, \bar{\boldsymbol{\psi}}[A']] + \boldsymbol{E}_{A'}[\boldsymbol{D}_{KL}[\bar{\varphi}[Y'|A'] \,\|\, \bar{\boldsymbol{\psi}}[Y'|A']]] \\
&\quad = \quad 0 + \mathcal{Q}[A' = \mathit{true}]\, \boldsymbol{D}_{KL}[\bar{\boldsymbol{\varphi}}[Y'|A' = \mathit{true}] \,\|\, \bar{\boldsymbol{\psi}}[Y'|A' = \mathit{true}]] \\
&\qquad\quad + \mathcal{Q}[A' = \mathit{false}]\, \boldsymbol{D}_{KL}[\bar{\boldsymbol{\varphi}}[Y'|A' = \mathit{false}] \,\|\, \bar{\boldsymbol{\psi}}[Y'|A' = \mathit{false}]] \\
&\quad = \quad 0 + 0 + (1-\gamma_{UY})\, \boldsymbol{D}_{KL}[\bar{\boldsymbol{\varphi}}[Y'|A' = \mathit{false}] \,\|\, \bar{\boldsymbol{\psi}}[Y'|A' = \mathit{false}]] \\
&\quad = \quad (1-\gamma_{UY})\, \boldsymbol{D}_{KL}[\bar{\varphi}[U] \,\|\, \bar{\boldsymbol{\psi}}[U]] \ ,
\end{aligned}$$

where we have used the fact that $Y' = c$ deterministically when $A' = true$, and that $Y' = U$ when $A' = false$.

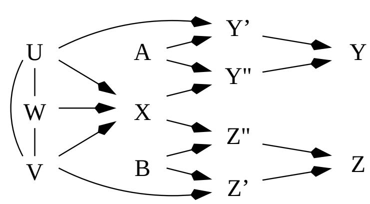
Analogously, the second term inside the expectation is bounded as follows:

$$\begin{aligned}
&\mathbb{D}_{KL}[\bar{\varphi}[Y''|U] \,\|\, \bar{\boldsymbol{\psi}}[Y''|U]] \\
&\quad \leq \; \mathbb{D}_{KL}[\bar{\varphi}[A'', Y''|U] \,\|\, \bar{\boldsymbol{\psi}}[A'', Y''|U]] \\
&\quad = \; \mathbb{D}_{KL}[\bar{\varphi}[A''|U] \,\|\, \bar{\boldsymbol{\psi}}[A''|U]] + \mathbb{E}_{A''|U}[\mathbb{D}_{KL}[\bar{\varphi}[Y''|A'', U] \,\|\, \bar{\boldsymbol{\psi}}[Y''|A'', U]]] \\
&\quad = \; 0 + \mathcal{Q}[A'' = true]\, \mathbb{D}_{KL}[\bar{\varphi}[Y''|A'' = true, U] \,\|\, \bar{\boldsymbol{\psi}}[Y''|A'' = true, U]] \\
&\qquad\;\; + \mathcal{Q}[A'' = false]\, \mathbb{D}_{KL}[\bar{\varphi}[Y''|A'' = false, U] \,\|\, \bar{\boldsymbol{\psi}}[Y''|A'' = false, U]] \\
&\quad = \; 0 + 0 + (1 - \gamma_{XY})\, \mathbb{D}_{KL}[\bar{\varphi}[Y''|A'' = false, U] \,\|\, \bar{\boldsymbol{\psi}}[Y''|A'' = false, U]] \\
&\quad = \; (1 - \gamma_{XY})\, \mathbb{D}_{KL}[\bar{\varphi}[X|U] \,\|\, \bar{\boldsymbol{\psi}}[X|U]] \; .
\end{aligned}$$

Since $\varphi[U, X] = \bar{\varphi}[U, X]$ and $\boldsymbol{\psi}[U, X] = \bar{\boldsymbol{\psi}}[U, X]$, the claim follows easily. ◻

Proof of Lemma 6.7 We pose $\gamma_Y = \gamma[\langle U, X\rangle \to Y]$ and $\gamma_Z = \gamma[\langle V, X\rangle \to Z]$, and, as before, $\gamma_{UY} = \gamma[U \to Y|X]$, *etc.*Unless otherwise specified, all expectations and mutual informations are defined with respect to the distribution $\mathcal{P}[U, V, W, X, Y, Z]$.

Intuitively, we consider a new distribution with the following structure:



We show that (suitable fragments of) the new distribution can be parameterized to behave like the original distribution, in a well defined sense. We exploit this fact to show a contraction property for the Y-transition and the Z-transition, in sequence. We then show that, taken together, they give the desired result.

The Y-transition Our objective here is to show that:

$$\boldsymbol{I}[Y; V|W] \stackrel{?}{\leq}$$

$$(1-\gamma_{UY})\,\boldsymbol{I}[U;V|W]+(1-\gamma_{XY})\,log_2\,[\#X]\ .$$

Fix $v \in dom[V]$ and $w \in dom[W]$, and pose, writing $\mathcal{P}[\cdot|w]$ for $\mathcal{P}[\cdot|W=w]$:

$$\begin{aligned}\boldsymbol{\varphi}[U,X] &= \mathcal{P}[U,X|v,w]\ ,\\ \boldsymbol{\psi}[U,X] &= \mathcal{P}[U,X|w]\ .\end{aligned}$$

Let $\mathcal{Q}[A,Y',Y'',Y|U,X]$ be the structured conditional distribution constructed as in Claim 6.8 to emulate $\mathcal{P}[Y|U,X,v,w]$ for $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$. Then define the following distributions:

$$\begin{aligned}\bar{\boldsymbol{\varphi}}[U,X,A,Y',Y'',Y] &= \boldsymbol{\varphi}[U,X]\otimes\mathcal{Q}[A,Y',Y'',Y|U,X]\ ,\\ \bar{\boldsymbol{\psi}}[U,X,A,Y',Y'',Y] &= \boldsymbol{\psi}[U,X]\otimes\mathcal{Q}[A,Y',Y'',Y|U,X]\ .\end{aligned}$$

By Claim 6.8, it holds that:

$$\boldsymbol{D}_{KL}[\mathcal{P}[Y|v,w]\,||\,\mathcal{P}[Y|w]] \;=\; \boldsymbol{D}_{KL}[\bar{\boldsymbol{\varphi}}[Y]\,||\,\bar{\boldsymbol{\psi}}[Y]]\ .$$

And then, by Claim 6.9:

$$\begin{aligned}&\boldsymbol{D}_{KL}[\mathcal{P}[Y|v,w]\,||\,\mathcal{P}[Y|w]]\\ &\quad\leq\ (1-\gamma_{UY})\,\boldsymbol{D}_{KL}[\boldsymbol{\varphi}[U]\,||\,\boldsymbol{\psi}[U]]\\ &\qquad\quad+(1-\gamma_{XY})\,\boldsymbol{E}_{\boldsymbol{\varphi}[U]}[\boldsymbol{D}_{KL}[\boldsymbol{\varphi}[X|U]\,||\,\boldsymbol{\psi}[X|U]]]\ ,\end{aligned}$$

where it is noted that the above expectation is taken with respect to $\boldsymbol{\varphi}$. Replacing $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$ by their definitions wherever they appear, it follows that:

$$\begin{aligned}&\boldsymbol{D}_{KL}[\mathcal{P}[Y|v,w]\,||\,\mathcal{P}[Y|w]]\\ &\quad\leq\ (1-\gamma_{UY})\,\boldsymbol{D}_{KL}[\mathcal{P}[U|v,w]\,||\,\mathcal{P}[U|w]]\\ &\qquad\quad+(1-\gamma_{XY})\,\boldsymbol{E}_{\mathcal{P}[U|v,w]}[\boldsymbol{D}_{KL}[\mathcal{P}[X|U,v,w]\,||\,\mathcal{P}[X|U,w]]]\ .\end{aligned}$$

Therefore, by taking the expectation over V and W:

$$\begin{aligned}
&\boldsymbol{E}_{\mathcal{P}[V,W]}[\boldsymbol{D}_{KL}[\mathcal{P}[Y|V,W] \,\|\, \mathcal{P}[Y|W]]] \\
&\quad\leq\ (1-\gamma_{UY})\,\boldsymbol{E}_{\mathcal{P}[V,W]}[\boldsymbol{D}_{KL}[\mathcal{P}[U|V,W] \,\|\, \mathcal{P}[U|W]]] \\
&\qquad +(1-\gamma_{XY})\,\boldsymbol{E}_{\mathcal{P}[U,V,W]}[\boldsymbol{D}_{KL}[\mathcal{P}[X|U,V,W] \,\|\, \mathcal{P}[X|U,W]]]\ .
\end{aligned}$$

On the other hand, the following identities are seen to hold universally:

$$\begin{aligned}
&\boldsymbol{I}[Y;V|W] \\
&\quad=\ \boldsymbol{E}_{\mathcal{P}[W]}[\boldsymbol{D}_{KL}[\mathcal{P}[Y,V|W] \,\|\, \mathcal{P}[Y|W]\otimes\mathcal{P}[V|W]]] \\
&\quad=\ \boldsymbol{E}_{\mathcal{P}[W]}[\boldsymbol{D}_{KL}[\mathcal{P}[V|W] \,\|\, \mathcal{P}[V|W]]] + \boldsymbol{E}_{\mathcal{P}[V,W]}[\boldsymbol{D}_{KL}[\mathcal{P}[Y|V,W] \,\|\, \mathcal{P}[Y|W]]] \\
&\quad=\ \boldsymbol{E}_{\mathcal{P}[V,W]}[\boldsymbol{D}_{KL}[\mathcal{P}[Y|V,W] \,\|\, \mathcal{P}[Y|W]]]\ , \\
&\boldsymbol{I}[U;V|W] \\
&\quad=\ \boldsymbol{E}_{\mathcal{P}[V,W]}[\boldsymbol{D}_{KL}[\mathcal{P}[U|V,W] \,\|\, \mathcal{P}[U|W]]]\ , \\
&\boldsymbol{I}[X;V|U,W] \\
&\quad=\ \boldsymbol{E}_{\mathcal{P}[U,V,W]}[\boldsymbol{D}_{KL}[\mathcal{P}[X|U,V,W] \,\|\, \mathcal{P}[X|U,W]]]\ .
\end{aligned}$$

By substituting each of these in the previous result, and noticing that $\boldsymbol{I}[X;\cdot|\cdot] \leq log_2\,[\#X]$, the first part of the claim is established.

The Z-transition Here, the objective is to show:

$$\begin{aligned}
&\boldsymbol{I}[Y;Z|W]\ \overset{?}{\leq} \\
&\quad (1-\gamma_{VZ})\,\boldsymbol{I}[Y;V|W] + (1-\gamma_{XZ})\,log_2\,[\#X]\ .
\end{aligned}$$

The argument is essentially the same as for the Y-transition, where the variables are substituted as follows, the role of each variable on the left-hand side below now being assumed by the variable on the right-hand side:

$$\begin{array}{rcl} U & \to & V \\ V & \to & Y \\ X & \to & X \\ W & \to & W \ , \end{array} \qquad\qquad \begin{array}{rcl} A & \to & B \\ Y' & \to & Z' \\ Y'' & \to & Z'' \\ Y & \to & Z \ . \end{array}$$

Combined transition Putting the two above intermediate results together, we get:

$$\begin{aligned} &\boldsymbol{I}[Y; Z|W] \\ &\quad \leq \ (1-\gamma_{UY})\,(1-\gamma_{VZ})\,\boldsymbol{I}[U;V|W] \\ &\qquad +(1-\gamma_{VZ})\,(1-\gamma_{XY})\,log_2\,[\#X] + (1-\gamma_{XZ})\,log_2\,[\#X] \\ &\quad \leq \ (1-\gamma_{UY})\,(1-\gamma_{VZ})\,\boldsymbol{I}[U;V|W] \\ &\qquad +(2-\gamma_{XY}-\gamma_{XZ})\,log_2\,[\#X] \ , \end{aligned}$$

as required.

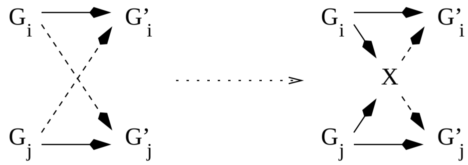
$\square$

We are now ready to prove our theorem for weakly interacting processes.

Theorem 6.10 *Let $\mathcal{F}$ and $\mathcal{G}$ be a cluster forest and group hierarchy, and G_i and G_j two siblings with an empty intersection, i.e., having no cluster of $\mathcal{F}$ in common. Let $\tilde{\boldsymbol{\mu}}$ be a distribution factored according to $\mathcal{F}$. Let $\boldsymbol{\varphi}$ be obtained from $\tilde{\boldsymbol{\mu}}$ by propagation through the given transition model. Then, with respect to $\boldsymbol{\varphi}$,*

$$\boldsymbol{I}[G_i^{\triangleright}; G_j^{\triangleright}] \ \leq \ (2-\gamma_{ij}-\gamma_{ji})\,(log_2\,[\#G_i]\,log_2\,[\#G_j]) \ .$$

Proof Consider the transition model involving G_i and G_j at the anterior and ulterior time slices. The proof is based on the construction of a related model that behaves equivalently on $\tilde{\boldsymbol{\mu}}$ and φ, and that contains a "mediator" variable X through which the cross-interaction between G_i and G_j is funneled.



Specifically, the transformed model contains the original random variables $G_i^{\triangleleft}$, $G_j^{\triangleleft}$, $G_i^{\triangleright}$, $G_j^{\triangleright}$, and also the new mediator variable X, whose domain has the cardinality of $G_i \times G_j$. X simply copies the values of $G_i^{\triangleleft}$ and $G_j^{\triangleleft}$, and directly influences $G_i^{\triangleright}$ and $G_j^{\triangleright}$ in a way that reproduces the previous cross-dependences $G_i^{\triangleleft} \to G_j^{\triangleright}$ and $G_j^{\triangleright} \to G_i^{\triangleleft}$ in the original model, which no longer exist in the new model. Notice that γ_{Xi} and γ_{Xj} in the new model are respectively equal to γ_{ji} and γ_{ij} in the original one. An application of Lemma 6.7 to the new structure now gives:

$$\begin{aligned}
&\boldsymbol{I}[G_i^{\triangleright}; G_j^{\triangleright}] \\
&\quad \leq \quad (2 - \gamma_{Xi} - \gamma_{Xj})\, log_2\, [\#X] + c\, \boldsymbol{I}[G_i^{\triangleleft}; G_j^{\triangleleft}] \\
&\quad = \quad (2 - \gamma_{Xi} - \gamma_{Xj})\, log_2\, [\#X]\ ,
\end{aligned}$$

where the term in $\boldsymbol{I}[G_i^{\triangleleft}; G_j^{\triangleleft}]$ vanishes as G_i and G_j are assumed independent in the anterior belief state $\tilde{\boldsymbol{\mu}}$. The claim follows, as $\#X = \#(G_i \times G_j) = (\#G_i\, \#G_j)$. ◻

Note that, as G_i and G_j are disjoint, we have $G_{i\cap j} = \emptyset$. Thus, the term $\boldsymbol{I}[G_i^{\triangleright}; G_j^{\triangleright}]$ bounded in this theorem is precisely the term $\boldsymbol{I}[G_i^{\triangleright}; G_j^{\triangleright} | G_{i\cap j}^{\triangleright}]$ that appears in Theorem 6.6. In other words, Theorem 6.10 gives us a bound on the error introduced by maintaining a factored belief state on two specific groups G_i and G_j. To get the overall bound on the error, we would simply add the contributions of all pairs of siblings in the group hierarchy $\mathcal{G}$, as per Theorem 6.10.

The bound for G_i and G_j closely matches our intuition regarding their interaction strength. Consider the term γ_{YX} for two groups X and Y that are weakly interacting. In this case, Y would not have a strong influence on X, which is to say that $\mathcal{P}[X^{\triangleright} | X^{\triangleleft} = x, Y^{\triangleleft} = y_1]$ is close to $\mathcal{P}[X^{\triangleright} | X^{\triangleleft} = x, Y^{\triangleleft} = y_2]$ for any x, y_1, and y_2. It follows that:

$$\sum_{x'} min\{\mathcal{P}[X^{\triangleright} = x' | X^{\triangleleft} = x, Y^{\triangleleft} = y_1], \mathcal{P}[X^{\triangleright} = x' | X^{\triangleleft} = x, Y^{\triangleleft} = y_2]\} \quad \lesssim \quad 1\ ,$$

for all x, y_1, y_2; hence γ_{YX} is also close to one. If both γ_{ij} and γ_{ji} are close to one, the error bound given by Theorem 6.10 will be close to zero.

To illustrate, consider the process composed of a number of cars on a freeway. In normal circumstances, the cars interact weakly with each other, so we want to place each car in a separate cluster F_i in our belief state representation. We can use the above theorem to justify this, as the weak interaction between the cars will ensure that each $\gamma_{ij} \simeq 1$ in any group hierarchy $\mathcal{G}$ that we choose. In fact, since the choice of $\mathcal{G}$ is arbitrary given a clustering $\mathcal{F}$, we can choose $\mathcal{G}$ to maximize the various γ_{ij}. In the example of the cars on a freeway, it is reasonable to assume that only neighboring vehicles may experience any kind of interaction. We can maximize γ_{ij} by minimizing the number of neighboring cars belonging to any two siblings G_i and G_j, which corresponds to the intuition that vehicles should be grouped according to their proximity.

It should however be noted that, contrarily to $\mathcal{F}$, the choice of $\mathcal{G}$ does not affect the approximation. Thus, a judicious group hierarchy will only lead to a tighter bound on the quality of the approximation, which itself depends on the actual clusters in $\mathcal{F}$.

6.4 Conditionally weak interactions

The previous section analyzed the error of approximating clusters of variables as completely independent. However, as experimentally observed in the last chapter, the tracking error can sometimes be substantially lowered by the use of conditionally independent clusters. For example, it may be much more reasonable to maintain an approximate distribution where the states of individual cars on a freeway are conditionally independent given the overall traffic conditions, as opposed to being fully independent. In this particular instance, the cluster forest $\mathcal{F}$ would be composed of as many clusters as there are vehicles, all of which share a common random variable describing the traffic conditions, in addition to vehicle-specific variables which are not shared. This kind of conditional independence relation causes some overlap among the clusters of $\mathcal{F}$, and, in turn, among some pairs of siblings in the group hierarchy $\mathcal{G}$. For Theorem 6.6 to remain of any use in this situation, we therefore need to analyze and bound the intrinsic error for two sibling groups that need not be disjoint.

Lemma 6.11 *Let $\boldsymbol{\varphi}[W, X, Y, Z]$ be an arbitrary probability distribution over the discrete variables W, X, Y, Z. Then, regardless of the dependency structure of the variables as defined by $\boldsymbol{\varphi}$,*

$$| \, \mathbb{I}[X;Y|Z] - \mathbb{I}[X;Y|W] \, | \quad \leq \quad \mathbb{H}[Z|W] + \mathbb{H}[W|Z] \ .$$

Proof We start by decomposing $\mathbb{I}[X;W,Y|Z]$ in two different ways, giving the identity:

$$\mathbb{I}[X;Y|Z] + \mathbb{I}[X;W|Y,Z] \quad = \quad \mathbb{I}[X;W|Z] + \mathbb{I}[X;Y|W,Z] \ .$$

From this equality, we successively derive:

$$\begin{aligned}
\mathbb{I}[X;Y|Z] + 0 &\leq (\mathbb{H}[W|Z] - \mathbb{H}[W|X,Z]) + \mathbb{I}[X;Y|W,Z] \ , \\
\mathbb{I}[X;Y|Z] + 0 &\leq (\mathbb{H}[W|Z] - 0) + \mathbb{I}[X;Y|W,Z] \ , \\
\mathbb{I}[X;Y|Z] - \mathbb{I}[X;Y|W,Z] &\leq \mathbb{H}[W|Z] \ .
\end{aligned}$$

From the same identity as before, we also derive:

$$\begin{aligned}
\mathbb{I}[X;Y|Z] - \mathbb{I}[X;Y|W,Z] &= \mathbb{H}[W|X,Y,Z] - \mathbb{H}[W|X,Z] \\
&\quad +(\mathbb{H}[W|Z] - \mathbb{H}[W|Y,Z]) \ , \\
\mathbb{I}[X;Y|Z] - \mathbb{I}[X;Y|W,Z] &= \mathbb{H}[W|X,Y,Z] - \mathbb{H}[W|X,Z] + \mathbb{I}[W;Y|Z] \ , \\
\mathbb{I}[X;Y|Z] - \mathbb{I}[X;Y|W,Z] &\geq 0 - \mathbb{H}[W|X,Z] + 0 \ , \\
\mathbb{I}[X;Y|Z] - \mathbb{I}[X;Y|W,Z] &\geq -\mathbb{H}[W|Z] \ .
\end{aligned}$$

Putting together the results of last two derivations, we obtain:

$$| \, \mathbb{I}[X;Y|Z] - \mathbb{I}[X;Y|W,Z] \, | \quad \leq \quad \mathbb{H}[W|Z] \ .$$

Similarly, by simple permutation of W and Z, we also have:

$$| \boldsymbol{I}[X;Y|W] - \boldsymbol{I}[X;Y|W,Z] | \quad \leq \quad \boldsymbol{H}[Z|W] \ .$$

The claim now follows from a direct application of the triangle inequality to the two above relations. $\boxed{}$

Theorem 6.12 *In the context of the previous definitions, let $\mathcal{F}$ be a cluster forest over some set of variables, $\mathcal{G}$ a cluster hierarchy compatible with $\mathcal{F}$, and G_i and G_j two sibling groups in $\mathcal{G}$. Let also $\boldsymbol{\varphi}$ and $\tilde{\boldsymbol{\mu}}$ be the exact and approximate distributions defined as in Theorem 6.10. Then, with respect to $\boldsymbol{\varphi}$, we have:*

$$\begin{aligned} &\boldsymbol{I}[G_i^{\triangleright}; G_j^{\triangleright} | G_{i\cap j}^{\triangleright}] \\ &\quad \leq \quad (2 - \gamma_{ij} - \gamma_{ji}) \, log_2 \left[\# G_{i\setminus j} \, \# G_{j\setminus i}\right] \\ &\qquad\quad + \boldsymbol{H}[G_{i\cap j}^{\triangleleft} | G_{i\cap j}^{\triangleright}] + \boldsymbol{H}[G_{i\cap j}^{\triangleright} | G_{i\cap j}^{\triangleleft}] \ . \end{aligned}$$

Proof The proof is based on a similar construction to that in Theorem 6.10, introducing a mediator variable X to capture the cross-interactions between $G_{i\setminus j}$ and $G_{j\setminus i}$. Using Lemma 6.7, we obtain:

$$\boldsymbol{I}[G_i^{\triangleright}; G_j^{\triangleright} | G_{i\cap j}^{\triangleleft}] \quad \leq \quad (2 - \gamma_{ij} - \gamma_{ji}) \, log_2 \left[\# G_{i\setminus j}^{\triangleleft} \, \# G_{j\setminus i}^{\triangleleft}\right] \ .$$

Applying Lemma 6.11, we get:

$$\boldsymbol{I}[G_i^{\triangleright}; G_j^{\triangleright} | G_{i\cap j}^{\triangleright}] \quad \leq \quad \boldsymbol{I}[G_i^{\triangleright}; G_j^{\triangleright} | G_{i\cap j}^{\triangleleft}] + \boldsymbol{H}[G_{i\cap j}^{\triangleleft} | G_{i\cap j}^{\triangleright}] + \boldsymbol{H}[G_{i\cap j}^{\triangleright} | G_{i\cap j}^{\triangleleft}] \ ,$$

hence the claim. $\boxed{}$

The following proposition offers an easy bound of the conditional entropy terms in the case where the transition $G_{i\cap j}^{\triangleleft} \to G_{i\cap j}^{\triangleright}$ is doubly stochastic for all values of $Pa^*[G_{i\cap j}^{\triangleright}] \setminus G_{i\cap j}^{\triangleleft}$. This assumption is likely to be met whenever the variables in $G_{i\cap j}$ evolve slowly.

Proposition 6.13 *If a process $M^{\triangleleft} \to M^{\triangleright}$ is* doubly stochastic*, i.e., such that every row and column of the transition matrix $\boldsymbol{P}[M^{\triangleright}|M^{\triangleleft}]$ sums to one, then:*

$$\begin{aligned}\boldsymbol{H}[M^{\triangleright}|M^{\triangleleft}] + \boldsymbol{H}[M^{\triangleleft}|M^{\triangleright}] &\leq 2\max_{m^{\triangleleft}} \boldsymbol{H}[M^{\triangleright}|M^{\triangleleft} = m^{\triangleleft}] \\ &= -2\max_{m^{\triangleleft}} \sum_{m^{\triangleright}} \boldsymbol{P}[m^{\triangleright}|m^{\triangleleft}] \log_2 \boldsymbol{P}[m^{\triangleright}|m^{\triangleleft}] \ .\end{aligned}$$

Proof By definition, we have:

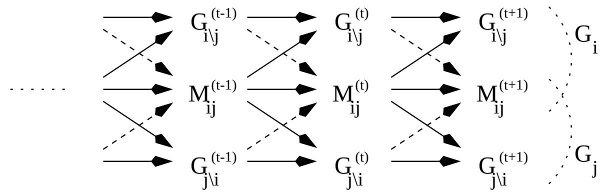
$$\begin{aligned}\boldsymbol{H}[M^{\triangleleft}|M^{\triangleright}] &= \boldsymbol{H}[M^{\triangleleft}] - \boldsymbol{I}[M^{\triangleleft}; M^{\triangleright}] \\ &= \boldsymbol{H}[M^{\triangleleft}] - \boldsymbol{H}[M^{\triangleright}] + \boldsymbol{H}[M^{\triangleright}|M^{\triangleleft}] \ .\end{aligned}$$

Now, if $M^{\triangleleft} \to M^{\triangleright}$ is doubly stochastic, this process is entropy-increasing toward a limiting uniform stationary distribution [CT91, page 35]. Therefore, $\boldsymbol{H}[M^{\triangleleft}] \leq \boldsymbol{H}[M^{\triangleright}]$, and we have:

$$\boldsymbol{H}[M^{\triangleright}|M^{\triangleleft}] + \boldsymbol{H}[M^{\triangleleft}|M^{\triangleright}] \leq 2\,\boldsymbol{H}[M^{\triangleright}|M^{\triangleleft}] \ ,$$

from which the claim follows. ◻

Let us return to the upper bound of the error provided by Theorem 6.12, and examine its expression from an intuitive standpoint. The first term was already present in Theorem 6.10, and represents the amount of correlation introduced by the weak interaction. The second term is new: it represents the amount by which conditioning on $G^{\triangleright}_{i\cap j}$ instead of $G^{\triangleleft}_{i\cap j}$ might change the mutual information. Intuitively, if $G^{\triangleright}_{i\cap j}$ is a faithful (deterministic) copy of $G^{\triangleleft}_{i\cap j}$, then conditioning on one or the other should not make any difference. In this case, indeed, we would have both conditional entropies equal to zero. This behavior generalizes to more realistic situations, where $G_{i\cap j}$ does evolve over time, but more slowly than the two clusters it separates. More precisely, let us assume that G_i and G_j interact only through $G_{i\cap j}$, and that $G_{i\cap j}$ tends to preserve its value from one step to the next. In particular, this implies that all external influences on $G_{i\cap j}$ are weak.



The subprocesses G_i and G_j are conditionally independent given the entire sequence $G^{(\cdot)}_{i\cap j}$. Assuming that $G_{i\setminus j}$ and $G_{j\setminus i}$ are mixing fast enough, $G^{(t)}_i$ and $G^{(t)}_j$ will be approximately independent given the values of $G^{(t')}_{i\cap j}$ in a vicinity of t. If $G_{i\cap j}$ evolves slowly, *i.e.*, if $\mathbb{H}[G^{\triangleleft}_{i\cap j}|G^{\triangleright}_{i\cap j}] \simeq 0$ and $\mathbb{H}[G^{\triangleright}_{i\cap j}|G^{\triangleleft}_{i\cap j}] \simeq 0$, then the values of $G^{(t')}_{i\cap j}$ in the vicinity of t will be approximately determined by the knowledge of $G^{(t)}_{i\cap j}$, so that $G^{(t)}_i$ and $G^{(t)}_j$ are approximately independent given the single point $G^{(t)}_{i\cap j}$. The same analysis holds if $G_{i\setminus j}$ and $G_{j\setminus i}$ do interact directly, but only weakly.

In the real world, there are many examples of processes whose primary interaction is via some more slowly evolving process. Our freeway example is a typical one: we can refine the model of the previous section if we consider external influences that affect some or all cars in mostly the same way, such as road maintenance or bad weather. A similar situation occurs in the stock market: the price trend of different stocks is clearly correlated, but they are reasonably modeled as conditionally independent given the current market trends. In both cases, the conditioning variables fluctuate more slowly than the dependent subprocesses. In these examples, our approximation model will contain a number of clusters $F_1, \ldots, F_k$ that all contain some variable W, which will therefore appear in the $G_{i\cap j}$ at one or more levels in the group hierarchy.

6.5 Sparsely interacting processes

As we have argued thus far, many systems are composed of interacting subprocesses. However, the assumption of weak interaction throughout the entire lifetime of the system is sometimes an unwarranted idealization. In many domains, while the various subprocesses would interact weakly, if at all, most of the time, they may have an occasional strong interaction. In our example of cars on a freeway, the interaction of one car with another is very weak most of the time; however, there are momentary situations when the interaction becomes quite strong, *e.g.*, when one car makes a sudden maneuver in the close vicinity of another.

The interaction structure of the system is very different in these two situations. As long as the subprocesses interact weakly, an approximation of the belief state as independent or conditionally independent components is justified. However, should a strong interaction occur, this approximation would incur a large error. A naive solution would be to keep correlated the partial belief states of any two processes that may interact strongly. Unfortunately, in many systems, this solution greatly reduces the effectiveness of selecting clusters of small sizes, as required to achieve computational efficiency. In our freeway example, this is to say that any cluster of cars that are not too far apart from each other should be tracked as a whole, in the off chance that one of the cars suddenly interacts strongly with another. Not only does this strategy cause a great increase in the computational effort, it is also impractical, since cars moving at different speeds will need to be reassigned to different clusters.

An alternative solution is to adapt the approximation to the context. When two processes have a momentary strong interaction, one should avoid decoupling them in the belief state. In fact, care must be taken, as the strong correlation between the two processes usually lasts for more than one time slice after the strong interaction. However, as the system evolves, the two processes return to their standard mode of weak interaction, allowing the correlation to decay. After some amount of time, it will be possible again to return to an approximation that decouples the states of the two processes.

We now proceed to extend both the inference algorithm and the analysis to account for this type of *sparse interaction*. First, we define the notion of sparse interaction. The idea is to consider two separate transition models, which will be applicable respectively in the standard and exceptional mode of interaction. Concretely, if X and Y interact sparsely, we define a binary random variable B_{XY} which will be a parent of both $X^{\triangleright}$ and $Y^{\triangleright}$, and will select their mode of interaction. We will speak of the *weak interaction model* and the *strong interaction model* to designate the portions of the conditional probability distributions of $X^{\triangleright}$ and $Y^{\triangleright}$ that are relevant to either value of B_{XY}.

The extended algorithm uses a different cluster forest $\mathcal{F}^{(t)}$ at each time slice t, which is constructed in terms of a given baseline cluster forest $\mathcal{F}_0$ and associated group hierarchy $\mathcal{G}_0$. In the absence of any strong interaction or correlation between any of the clusters at time t, $\mathcal{F}^{(t)} = \mathcal{F}_0$ is used for the approximation. If at time t a strong interaction is detected between variables in two distinct clusters of $\mathcal{F}_0$, the algorithm switches to a forest $\mathcal{F}^{(t)}$ in which those clusters are coupled; this temporary coupling is achieved by identifying the two sibling groups of $\mathcal{G}_0$ that each contained one of the interacting variables, and merging them into a temporary super-cluster, to obtain $\mathcal{F}^{(t)}$. If no more strong interaction subsequently occurs for some number of time slices d, the algorithm returns to the original approximation.

More precisely, at each time t, we maintain a set $C^{(t)} \subseteq S_G$ of *couplings*, which indicates which among the pairs of siblings $\{G_i, G_j\}$ are currently coupled. The cluster forest $\mathcal{F}^{(t)}$ is derived from the baseline cluster forest $\mathcal{F}_0$, as follows. Each cluster $F_k^{(t)}$ in $\mathcal{F}^{(t)}$ is either a cluster of $\mathcal{F}_0$, or is the union of two siblings G_i and G_j; in other terms, each cluster F_l in $\mathcal{F}_0$ is assigned to some cluster $F_k^{(t)}$, in which case $F_l \subseteq F_k^{(t)}$. The requirement is that, if F_l is assigned to $F_k^{(t)}$, while $F_l \subseteq G_i$, and $\{G_i, G_j\} \in C^{(t)}$, then $G_i \cup G_j \subseteq F_k^{(t)}$.

We remark that, in general, the algorithm might not be able to observe directly the existence of a strong correlation. In some circumstances, there may be certain tests that are reliable indicators of such an event, *e.g.*, a proximity radar mounted on a car signalling that another car is dangerously close. In other domains, the strong interaction may be due to an action taken by the very agent tracking the system, in which case, correlations can be predicted. One general, albeit expensive, approach for discovering strong interactions between two clusters, is to compute a rough estimate of the error that would be incurred by taking two clusters to be independent, *e.g.*, by computing the statistical correlation between pairs of clusters, or similar heuristics. The assumption that we are making here, is that the occurrence of a strong interaction between two clusters X and Y, as signaled by the random variable B_{XY} can readily be detected, even though no further information may necessarily be observed beyond the binary value of B_{XY}. In fact, the only requirement is that occurrences of strong interaction be reliably detected, even though false positives are tolerated.

6.5.1 The independent case

We begin by analyzing the error of the adaptive filtering algorithm in the case where the groups are disjoint. Let thus G_i and G_j be two siblings such that $G_i \cap G_j = \emptyset$. The question is, what does this pair of groups contribute to the projection error at time t, in the decomposition of Theorem 6.6? Clearly, as long as G_i and G_j belong to the same super-cluster of $\mathcal{F}^{(t)}$, the contribution is null, as there is no approximation. We therefore assume that G_i and G_j are decoupled in $\mathcal{F}^{(t)}$, or, equivalently, that $\{G_i, G_j\} \notin C^{(t)}$, which implies a regime of weak interaction at time t.

There are two cases of interest. If G_i and G_j were also decoupled at time $t-1$, then the local situation for the most recent transition was that of a weak interaction under a static approximation. In this case, the analysis reduces to that of Theorem 6.10. In the converse situation, the siblings G_i and G_j were coupled in $\mathcal{F}^{(t-1)}$, and it was chosen at this time slice to decouple them. This indicates that these groups have been coupled for some number d of time slices, during which period no error has been incurred by the pair, following a strong interaction. The following theorem provides an estimate of the extent to which the strong correlation that was present d time slices ago has attenuated. The bound depends only on the properties of the weak interaction model, and makes no assumption on the strength of the correlation that was present at time $t - d$.

Theorem 6.14 *Let G_i and G_j be two disjoint sibling groups in $\mathcal{G}$. Let also $\gamma_{ij}^w = \gamma[G_i^{\triangleleft} \to G_j^{\triangleright}]$ be the mixing coefficient of the stochastic transition under the weak mode of interaction. Then, the projection error incurred for not coupling G_i and G_j in the forest $\mathcal{F}^{(t)}$ is bounded as follows:*

1. *If G_i and G_j were decoupled at time $t-1$, and no strong interaction has occurred between them at time t, then,*

$$\boldsymbol{I}[G_i^{(t)}; G_j^{(t)}] \le (2 - \gamma_{ij}^w - \gamma_{ji}^w) \log_2 \#(G_i \times G_j) \ .$$

2. *If G_i and G_j were coupled at all times $t' \in \{t - d + 1, ..., t - 1\}$, and no strong*

interaction has occurred between them since time $t-d$, then,

$$\boldsymbol{I}[G_i^{(t)}; G_j^{(t)}] \;\le\; log_2 \#(G_i \times G_j) \left((1-\gamma_{ii}^w)^d \, (1-\gamma_{jj}^w)^d + \frac{2-\gamma_{ij}^w-\gamma_{ji}^w}{\gamma_{ii}^w+\gamma_{jj}^w-\gamma_{ii}^w\,\gamma_{jj}^w} \right) \; .$$

Proof The first case follows from Theorem 6.10. The general case is obtained by applying Lemma 6.7 d times in sequence, where, for $t' = t-d+1, ..., t$, we map $W \leftarrow \emptyset$, $U \leftarrow G_i^{(t'-1)}$, $V \leftarrow G_j^{(t'-1)}$, $Y \leftarrow G_i^{(t')}$, $Z \leftarrow G_j^{(t')}$, and $X \leftarrow \langle G_i^{(t'-1)}, G_j^{(t'-1)} \rangle$. We obtain:

$$\begin{aligned} \boldsymbol{I}[G_i^{(t)}; G_j^{(t)}] \;\le\;& \\ & (1-\gamma_{ii}^w)^d \, (1-\gamma_{jj}^w)^d \, \boldsymbol{I}[G_i^{(t-d)}; G_j^{(t-d)}] \\ & + (2-\gamma_{ij}^w-\gamma_{ji}^w) \, log_2 \#(G_i \times G_j) \sum_{k=0}^{d-1} (1-\gamma_{ii}^w)^k \, (1-\gamma_{jj}^w)^k \; . \end{aligned}$$

It is noted that all the mixing coefficients above refer to the weak interaction model, since it is assumed that no strong interaction has occurred since epoch $t-d$. Then, the claim follows from the inequalities $\boldsymbol{I}[G_i^{(t-d)}; G_j^{(t-d)}] \le min\{\boldsymbol{H}[G_i^{(t-d)}], \boldsymbol{H}[G_j^{(t-d)}]\} \le log_2 \#(G_i \times G_j)$, and the fact that, for $x < 1$, $\sum_{k=0}^{d-1} x^d \le 1/(1-x)$. ◻

In summary, the projection error induced by decoupling two groups with a delay of d time slices after a strong correlation decreases exponentially with d. The theorem also provides a way to calculate a priori how long a pair of sibling groups should be kept coupled in order to guarantee a chosen bound on the error.

To see how this theorem can be used, let us go back to the freeway example, and assume that we observe a strong interaction between two vehicles at time t, such as one cutting in front of the other. In response to this observation, the algorithm would then couple the sibling groups G_i and G_j to which these vehicles belong, which results in the creation of a super-cluster of all the variables in G_i and G_j. This coupling goes on for a certain number of time slices, until it can be guaranteed that the correlation induced by the strong interaction has sufficiently decayed, so that the groups can be decoupled without incurring too much error. Theorem 6.14 guarantees that this eventually happens, and gives an upper bound on the delay until it does.

6.5.2 The conditionally independent case

The following theorem states the general result for sparsely interacting overlapping clusters.

Theorem 6.15 *Let G_i and G_j be two sibling groups in $\mathcal{G}$, and $G_{i\cap j}$ their intersection. Then, the projection error incurred for not coupling G_i and G_j in the forest $\mathcal{F}^{(t)}$ is bounded as follows:*

1. *If G_i and G_j were decoupled at time $t-1$, and no strong interaction has occurred between them at time t, then,*

$$\boldsymbol{I}[G_i^{(t)}; G_j^{(t)} | G_{i\cap j}^{(t)}] \leq (2 - \gamma_{ij}^w - \gamma_{ji}^w) \log_2 \#(G_i \cup G_j) + \boldsymbol{H}[G_{i\cap j}^{(t-1)} | G_{i\cap j}^{(t)}] + \boldsymbol{H}[G_{i\cap j}^{(t)} | G_{i\cap j}^{(t-1)}] \ .$$

2. *If G_i and G_j were coupled at all times $t' \in \{t-d+1, ..., t-1\}$, and no strong interaction has occurred between them since time $t-d$, then,*

$$\begin{aligned}\boldsymbol{I}[G_i^{(t)}; G_j^{(t)} | G_{i\cap j}^{(t)}] \leq \; & \log_2 \#(G_{i\setminus j} \cup G_{j\setminus i}) \, (1-\gamma_{ii}^w)^d \, (1-\gamma_{jj}^w)^d \\ & + \sum_{k=0}^{d-1} (1-\gamma_{ii}^w)^k \, (1-\gamma_{jj}^w)^k \begin{pmatrix} (2-\gamma_{ij}^w - \gamma_{ji}^w) \log_2 \#(G_{i\setminus j} \cup G_{j\setminus i}) \\ + \boldsymbol{H}[G_{i\cap j}^{(t-k-1)} | G_{i\cap j}^{(t-k)}] + \boldsymbol{H}[G_{i\cap j}^{(t-k)} | G_{i\cap j}^{(t-k-1)}] \end{pmatrix} .\end{aligned}$$

Proof The proof is analogous to that of Theorem 6.14, making use of d sequential applications of Lemma 6.7, where, for $t' = t-d+1, ..., t$, we map $W \leftarrow G_{i\cap j}^{(t'-1)}$, $U \leftarrow G_{i\setminus j}^{(t'-1)}$, $V \leftarrow G_{j\setminus i}^{(t'-1)}$, $Y \leftarrow G_i^{(t')}$, $Z \leftarrow G_j^{(t')}$, and $X \leftarrow \langle G_i^{(t'-1)}, G_j^{(t'-1)} \rangle$. ◻

6.6 Toward devising approximation heuristics

The results in this chapter provide a useful set of rules for understanding how the interaction structure affects how the information propagates within a complex stochastic process. Naturally, the same results may also serve as a basis for operational tools for deciding on a suitable approximation scheme for a particular process, whether statically ahead of time, or even adaptively as the observation data pour in.

The rules that govern the construction of a good approximation can be inferred from the various results in this chapter. One of the difficulties faced by any algorithmic determination of an optimal approximation strategy, *i.e.*, in the form of a static or adaptive clustering scheme, is that there is no absolute optimality criterion. At one end of the spectrum, the optimal accuracy is achieved by not doing any approximation whatsoever; at the other end, the most efficient clustering is one that places each variable in its own cluster. Between these extremes, the space of possibilities is typically much too large to afford an exhaustive search. However, a number of heuristics can be used in order to devise adequate approximations for a particular set of requirements driven by the application domain.

Non-overlapping clusterings, *i.e.*, approximations in which the connectivity of the cluster forest $\mathcal{F}$ is equal to the number of clusters it contains, are relatively straightfoward to assemble algorithmically. One idea is to construct a group hierarchy $\mathcal{G}$ from the bottom up, starting from the individual variables, in a greedy fashion similar to hierarchical agglomerative clustering. Each variable is first assigned to a group of its own. Then, for as long as more than one group remains, the algorithm selects two groups from the current pool, and combines them into a new group—the parent in the emerging hierarchy. Let thus $\mathcal{G}'$ be the resulting binary hierarchy. The cluster forest $\mathcal{F}$ is constructed as a collection of groups, elements of $\mathcal{G}'$, that perfectly cover all the variables without overlap.

In the above, the two decision points that require heuristics are: (1) the selection of the pair of groups to be combined during the elaboration of $\mathcal{G}'$, and (2) the selection of the groups from $\mathcal{G}'$ that are to constitute the clusters of $\mathcal{F}$. In both cases the goal is to maximize some local objective function, which balances the computational cost of trading two sibling groups for their parent, against the expected reduction in error caused by such a trade, which can be estimated, *e.g.*, using the results of Section 6.3.

Similar greedy heuristics can be used in the overlapping case, *i.e.*, where it is desired that the cluster count of $\mathcal{F}$ be larger than its connectivity. For this, the greedy construction of $\mathcal{G}'$ needs to be modified in order to allow sibling groups to overlap, in conjunction with an objective function based on the notions from Section 6.4.

All these ideas also fit nicely with the insights from Section 6.5 regarding dynamic approximations. In this case, it is assumed that the aggressive, efficient default approximation used for inference under ordinary circumstances, is ready to be converted into a more conservative approximation at the onset of a strong interaction between variables in multiple clusters. To facilitate the conversion, the approximation-devising algorithm could be instructed to keep in relative proximity any set of variables likely to experience an occasional strong interaction.

Part III

Learning

Chapter 7

Learning under uncertainty

The techniques presented in the previous chapters provide an effective way of reasoning about the evolution of a stochastic process, provided that a model of the system is available. When the only available information is a stream of historical observations of the process behavior, it becomes necessary to estimate the system itself, in addition to its current state. This is commonly referred to as the *learning* task.

Sometimes, there may be some information available about the system dynamics, although not enough to conduct inference and tracking using the techniques previously described. For example, in the case of structured systems, a qualitative description of the dependency structure may be at hand, albeit without any quantitative probability model. In other cases, the nature of the state variables may be known, without even a qualitative interaction model. In extreme situations, there is simply no information beyond the observed variables themselves, so that even the mere existence of hidden state variables has to be inferred.

Recent works have contributed significant progress on various aspects of learning Bayesian networks from partially observed data; these include Heckerman's analysis on BN learning from a mathematical perspective [Hec99], and the efficient Structural-EM algorithm of Friedman [Fri97] which extends to structural learning Lauritzen's parametric EM algorithm for BN [Lau95], itself a particularization of the celebrated EM algorithm [DLR77]. Other works have addressed the problem of learning DBNs *per se*; Friedman *et al.* [FMR98] provide a detailed description of how BN learning can be applied to DBNs for fully observed data.

The general approach for learning stochastic models from partial data uses a hill climbing strategy, whereby an initial candidate model is iteratively refined and improved, using the observation data. The particulars vary according to which of the above situations is faced: for example, if the qualitative structure is known, all the information contained in the data may be used to refine the qualitative model parameters, based on statistics about the process trajectory. In all cases, probabilistic inference plays the key role in estimating those statistics from the noisy data provided by the stream of partial observations.

This chapter reviews the state-of-the-art techniques involved in learning the parameters and the structure of partially observable stochastic process models from data, with an emphasis on structured models such as DBNs. As those techniques are best understood relatively to the fully observable case, we shall make a short digression on this simpler problem.

7.1 Learning from time series

Recall from Section 3.5 that a DBN represented by a 2-TBN over a set of variables $\mathbf{U}$ defines a distribution over infinite trajectories of states defined by $\mathbf{U}$. In practice, we reason only about a finite time interval $0, ..., T$. To do this reasoning, we can notionally unroll the DBN structure into a long BN over $\mathbf{U}^{(0)}, ..., \mathbf{U}^{(T)}$, where the conditional probability distributions are duplicated from the 2-TBN, except for the variables in time slice 0, which must be assigned an initial state distribution. We will denote the entire DBN model by $\mathbf{B}_\star = \langle \mathbf{B}_0, \mathbf{B}_\rightarrow \rangle$, where $\mathbf{B}_0$ is the initial distribution over $\mathbf{U}^{(0)}$, and $\mathbf{B}_\rightarrow = \langle \mathbf{G}, \Theta \rangle$ is a 2-TBN with graphical structure $\mathbf{G}$ and numerical parameterization Θ, that represents the conditional probability model $\boldsymbol{P}[\mathbf{U}^\triangleright | \mathbf{U}^\triangleleft]$, which, for all $t > 0$, defines a stochastic transition from $\mathbf{U}^{(t-1)}$ to $\mathbf{U}^{(t)}$. Thus, a DBN $\mathbf{B}_\star = \langle \mathbf{B}_0, \mathbf{B}_\rightarrow \rangle$ over the variables $\mathbf{U}$ represents, for any T, the the joint distribution over $\mathbf{U}^{(0)}, ..., \mathbf{U}^{(T)}$, given by:

$$\mathbf{B}_\star[\mathbf{u}^{(0)}, ..., \mathbf{u}^{(T)}]$$

$$= \mathbf{B}_0[\mathbf{u}^{(0)}] \prod_{t=1}^{T} \mathbf{B}_{\rightarrow}[\mathbf{U}^{\triangleright} = \mathbf{u}^{(t)} | \mathbf{U}^{\triangleleft} = \mathbf{u}^{(t-1)}] \ .$$

We now consider the task of learning the transition model $\mathbf{B}_{\rightarrow}$ of a DBN from data. For simplicity of notation, we assume that our data set $\mathbf{D}$ is a single finite-length trajectory $\mathbf{d}^{(0)}, ..., \mathbf{d}^{(T)}$ through the system; in this section, we assume that this trajectory is fully observable. For simplicity, we also ignore the task of learning the prior network $\mathbf{B}_0$. Given a training sequence, the learning task is to find the network $\mathbf{B}_{\rightarrow}$ that provides a best match, or best explanation, for $\mathbf{D}$. The notion of best match is defined using a scoring function. Several different scoring functions have been proposed in the literature. The most frequently used score functions are the *Bayesian information criterion (BIC)* [Sch78] and the *Bayesian score (BDe)* [HGC95]. Both of these scores combine a measure of fit to data with some penalty relating to the complexity of the network. For ease of presentation, we will focus on the BIC score.

In the BIC score, the term that represents the fit of $\mathbf{B}_{\rightarrow}$ to the data $\mathbf{D}$ is the *log-likelihood function*, defined as $\ell[\mathbf{B}_{\star} : \mathbf{D}] = log_2 \mathbb{P}[\mathbf{D}|\mathbf{B}_{\star}]$. This function measures the extent to which the data set is likely given a candidate model $\mathbf{B}_{\star}$; it is thus an estimate of how well a given candidate model fits the empirical data. The log-likelihood can be computed from the *sufficient statistics*, which summarize the observed frequencies of the relevant events in the data. The BDe score contains a similar *marginal log-likelihood* term, which can also be efficiently computed from sufficient statistics summarizing the data.

Definition 7.1 Let E denote any event over the anterior and ulterior variables $\langle \mathbf{U}^{\triangleleft}, \mathbf{U}^{\triangleright} \rangle$ in the transition model, *i.e.*, $E \subseteq dom[\langle \mathbf{U}^{\triangleleft}, \mathbf{U}^{\triangleright} \rangle]$. Let $E^{(t)}$ denote the event over $\langle \mathbf{U}^{(t-1)}, \mathbf{U}^{(t)} \rangle$, obtained by fixing the anterior and ulterior time slices of E to correspond to times $t-1$ and t respectively. Let also $\iota[E^{(t)}|\mathbf{D}]$ be an indicator function which takes on the value 1 if the event E over $\langle \mathbf{U}^{\triangleleft}, \mathbf{U}^{\triangleright} \rangle$ holds for the joint instantiation $\mathbf{U}^{\triangleleft} = \mathbf{d}^{(t)}$ and $\mathbf{U}^{\triangleright} = \mathbf{d}^{(t+1)}$, and 0 otherwise. Then, the *cumulative* *frequency* of event E in a data stream $\mathbf{D} = \{\mathbf{d}^{(0)}, ..., \mathbf{d}^{(T)}\}$, is defined as:

$$N_E \;=\; \sum_{t=1}^{T} \iota[E^{(t)}|\mathbf{D}] \; .$$

Letting $U_1, ..., U_n$ be the n variables composing $\mathbf{U}$, the log-likelihood can now be characterized as:

$$\ell[\mathbf{B}_{\rightarrow} : \mathbf{D}] \;=\; \sum_{i=1}^{n} \sum_{u_i^{\triangleright} \in dom[U_i^{\triangleright}]} \sum_{\mathbf{x} \in dom[Pa[U_i^{\triangleright}]]} N_{u_i^{\triangleright},\mathbf{x}} \, log_2 \, \Theta_{u_i^{\triangleright}|\mathbf{x}} \; ,$$

where $\Theta_{u_i^{\triangleright}|\mathbf{x}} = \mathbf{B}_{\rightarrow}[u_i^{\triangleright}|\mathbf{x}]$ denotes the parameters of $\mathbf{B}_{\rightarrow}$. The BIC score is simply the log-likelihood plus a penalty term for network complexity:

$$\varsigma_{BIC}[\mathbf{B}_{\rightarrow}] \;=\; \ell[\mathbf{B}_{\rightarrow} : \mathbf{D}] - \frac{log_2 \, T}{2} \, dim[\mathbf{B}_{\rightarrow}] \; ,$$

where $dim[\mathbf{B}_{\rightarrow}]$ is the *dimension* of $\mathbf{B}_{\rightarrow}$, which in the case of complete data is simply the number of independent, real-valued parameters in the conditional probability distributions of $\mathbf{B}_{\rightarrow}$.

The goal is to find the network that maximizes this score. For a fixed structure, the parameters that maximize the score are exactly the maximum likelihood parameters, which simply mirror the frequencies in the data:

$$\hat{\Theta}_{y_i|\mathbf{x}} \;=\; \frac{N_{y_i|\mathbf{x}}}{N_{\mathbf{x}}} \; .$$

This is particularly easy, as the only statistics $N_{\bullet}$ needed are the frequencies over the various families in the network structure $\mathbf{G}$, where a family is the set composed of a node and all its parents.

Finding the highest scoring network structure is NP-hard [CGH95]. In practice, one usually resorts to greedy local search procedures [Bun91, HGC95] that gradually improve a candidate structure by applying local structural transformation: adding, deleting, or reversing an edge. These transformations are usually applied in a greedy fashion, with occasional random steps to deal with local maxima and plateaux. Two crucial properties of the BIC score greatly facilitate this procedure. First, the score of a network can be written as a sum of terms, where each term determines the score of a particular choice of parents for a particular variable. Thus, a local change to one family $U_i^\triangleright, Pa[U_i^\triangleright]$, such as the addition or removal of an arc, affects only one of these terms. As a consequence, the incremental value of any change to another family in the network remains unchanged. Hence, to determine the values of all local changes to the current network structure, it is only necessary to re-evaluate changes to the family of $U_i^\triangleright$. Second, the term that evaluates the family of $X_i^\triangleright$ is a function only of the sufficient statistics for $X_i^\triangleright$ and its parents $Pa[U_i^\triangleright]$. These sufficient statistics are the only aspects of the data about which statistics need to be collected. However, as the families $\{U_i^\triangleright\} \cup Pa[U_i^\triangleright]$ will likely differ for the various candidate models considered at any iteration, it is necessary to collect statistics on a subtantially larger set of events than was needed for parametric learning only. In general, evaluation of local changes in the network topology usually involves the computation of new sufficient statistics, followed by an evaluation of the score of the new models using those statistics.

The Bayesian score is somewhat more complex, and involves taking a prior distribution over model structures and parameters into account. Without delving into much detail, we note that for suitable choices of priors, such as the BDe priors of [HGC95], the two key features above are preserved: the score decomposes in to a sum of terms, and depends only on the sufficient statistics collected from data. Although the Bayesian score and the Bayesian information criterion are asymptotically equivalent, for small sample sizes the Bayesian score often performs better, at the expense of requiring a prior distribution over both structures and parameters.

7.2 The Expectation-Maximization algorithm

The main difficulty with learning from partial observations is that some of the counts in the data are not available, preventing the computation of sufficient statistics from which a model can be inferred. One way around this is to estimate the missing data from the current model and the data we have, and use the completed data to obtain the sufficient statistics. However, since statistics computed this way depend on the learned model itself, a circular dependency arises between learned model and estimated data. A related difficulty is that neither the BIC nor the BDe score of a learned model given incomplete data decomposes into separate components corresponding to individual families.

The most common solution to the missing data estimation problem is provided by the *Expectation-Maximization (EM)* algorithm [DLR77, Lau95]. The EM algorithm is an iterative procedure that searches for a parameter vector Θ^* which is a local maximum of the likelihood function. It starts with some initial, usually random, parameter vector Θ, and then repeatedly executes a two-phase procedure, composed of the *E-step* and the *M-step*. In the E-step, the current parameters are used to *complete* the data by estimating unobserved values using their expected values given the available data and the current parameters. In the M-step, the completed data set is used as if it were real, to update the model parameters in a maximum likelihood estimation step similar to the complete case. Both steps are detailed below.

E-step In the expectation step, the algorithm computes the *expected sufficient statistics (ESS)* for the data $\mathbf{D}$, relative to the current model structure $\mathbf{G}$ and parameters Θ. The ESS take the form of expected cumulative event frequencies, for any event of interest, which are obtained as follows:

$$\begin{aligned}
\bar{N}_E &= \boldsymbol{E}_{\langle \mathbf{G},\Theta\rangle}[N_E] \\
&= \sum_{t=1}^{T} \boldsymbol{E}_{\langle \mathbf{G},\Theta\rangle}[\iota[E^{(t)}|\mathbf{D}]] \\
&= \sum_{t=1}^{T} \boldsymbol{P}[E^{(t)}|\mathbf{D}, \langle \mathbf{G},\Theta\rangle] \ . \qquad (7.1)
\end{aligned}$$

M-step In the maximization step, the new set of parameters Θ' are estimated as in the case of parametric learning from complete data, except that the ESS $\bar{N}_\bullet$ are used in place of the unobserved actual frequencies $N_\bullet$:

$$\Theta'_{y_i|\mathbf{x}} = \frac{\bar{N}_{y_i|\mathbf{x}}}{\bar{N}_{\mathbf{x}}} \ .$$

The alternation of the E and M steps repeats until some convergence criterion has been met, at which point the last computed parameterization—or, as usual in machine learning, the parameterization that attained the best score on a separate validation data set sampled from the same distribution as $\mathbf{D}$—is retained. Termination may be triggered, *e.g.*, by detecting that the rate of improvement of the likelihood function has fallen below some absolute or relative threshold.

The algorithm's effectiveness rests on a fundamental theorem which states that each EM cycle is guaranteed not to degrade the likelihood of the data $\mathbf{D}$ given the model $\langle \mathbf{G}, \Theta \rangle$. In other words, the trajectory of the model parameters Θ, as updated by the EM algorithm, is guaranteed to converge to a local extremum of the likelihood function, for any given data sequence $\mathbf{D}$ and fixed model structure $\mathbf{G}$. In general, however, there are no guarantees regarding global convergence properties.

7.3 The Structural-EM algorithm

EM has been traditionally viewed as a method for adjusting the parameters of a fixed model structure. Friedman's *Structural EM (SEM)* algorithm [Fri97] extends it to the task of structure learning. The SEM algorithm has the same E-step as EM, completing the data by computing expected counts based on the current structure and parameters. In addition to re-estimating the parameters, the M-step of SEM uses the ESS, computed according to the current structure, to score other candidate structures. Essentially, the algorithm uses the current network structure to compute ESS not only on the current families (as would be required for merely updating the parameters), but also on supersets of those families. The algorithm then conducts a greedy structure search in the vicinity of the current structure, as in the complete data case, using those ESS. The structure search stops after some number of steps, at which point the process repeats using the current candidate network.

Friedman [Fri98] shows that, for a large family of scoring rules, the network resulting from this inner loop must have a higher score than the original. This is true even though the ESS used in evaluating the new candidate structures are computed using the structure from the previous step. More precisely, Friedman defines a notion of *expected score*, which is the expectation of the score for different completions $\mathbf{D}^+$ of the data $\mathbf{D}$, where the probability of a completion $\mathbf{D}^+$ is $\mathbb{P}[\mathbf{D}^+|\mathbf{D}, \mathbf{B}_\star]$. For a large class of scores, such as BIC or BDe, it can be shown that if a change is made to the network structure that increases the expected score, then the true score increases by at least as much. The crucial property of the expected score is that, like the actual score in the case of complete data, it decomposes into a sum of local terms. In particular, the expected BIC score is simply the BIC score applied to the expected sufficient statistics. This property reinstates both of the key requirements previously encountered in the case of structure search for complete data: the ability to evaluate a structural change by considering only the families affected by the change, and the ability to summarize an arbitrarily large data sequence into expected sufficient statistics of predetermined size, for the purpose of computing the score.

We now give a succinct overview of the innards of the SEM algorithm; see [Fri97, Fri98] for a complete description. The process is iterative, and consists of an alternation of an E-step and a M-step, updating the current model structure and parameters in each round, until some convergence criterion is satisfied. The E-step and M-step are described next.

7.3.1 The E-step

As in the regular EM algorithm, the purpose of the estimation step in SEM is to obtain a set of expected sufficient statistics. First, the missing data is completed by computing its expectation given the current model and the available data. Then, the completed data is summarized into ESS (which simply consist of frequency counts for all events of interest, and are thus trivial to obtain from the completed data).

Candidate determination

The main difference with regular (parametric) EM, is that we need more ESS in order to evaluate different candidate structures in the M-step. Precisely, a number of candidate structures are (implicitly) generated by considering a number of small modifications to the current model, such as the addition and deletion of a few edges, according to some suitability criterion involving, among other things, the cost of obtaining ESS for the various candidates. Since the ESS are defined differently in the various structures, the algorithm needs to ensure that it will collect enough information during the inference step, in order to later compute all the ESS it needs. Thus, in SEM, there is an extra step taking place before inference in the E-step, in which the algorithm creates a list of clusters of variables over which to gather statistics.

Statistics collection

Once the list of needed statistics is created, probabilistic inference is used to compute the expected values of the missing data. During this step, marginals over all required clusters of variables are computed, using an inference algorithm such as that of Section 3.4; these marginals directly translate into expected frequency counts. It is emphasized that all statistics are computed with respect to the current model structure.

7.3.2 The M-step

In the M-step, the implicit class of candidate structures considered in the E-step is searched for a model structure (and corresponding parameters) that maximizes the score.

Greedy search and scoring

The search proceeds in a greedy fashion, starting from the current model, by evaluating one-step structure changes in each family, using the best set of parameters as determined by the relevant ESS in each case. The local score gain or loss is evaluated for all the changes, and the best option is selected. Since the BIC and BDe scores decompose nicely according to the families of variables in the network; only the local contribution to the total score needs to be recomputed for each local change.

Termination criterion

At some point, when the greedy search reaches the boundary of the available ESS, the M-step terminates, leading to a new iteration unless the algorithm deems that the convergence criterion is satisfied, such as, among other things, the stagnation of the scoring metric for a number of interations.

7.4 The Forward-Backward algorithm

Whereas the EM algorithm provides an effective method for dealing with incomplete data, as encountered when learning partially observable processes, the main difficulty consists in computing various expected sufficient statistics $\mathbb{E}_{\langle \mathbf{G},\Theta \rangle}[E|\mathbf{D}]$ from the partially observed data sequence $\mathbf{D}$, in the E-step of the algorithm. This computation, as shown in Equation 7.1, is done using probabilistic inference for all instantiations $E^{(t)}$ of any event E of interest. In general, such inference has to be conducted on the entire trajectory $\mathbf{D}$, since the distribution of any event $E^{(t)}$ involving hidden state variables is generally going to depend on all observations, whether they occurred in the past or in the future of $E^{(t)}$. *A fortiori*, the generic event E, as integrated over the entire sequence, is also going to depend on the entire data set.

The classical forward-backward algorithm for HMMs [Bau72, RJ86] performs this operation in an elegant and efficient way, by allowing all expected sufficient statistics of interest to be computed using only a single traversal of the data sequence. As usual, this algorithm is based on Aström's principle that, at any point in time, the belief state of a Markovian process captures all the information that connects the past history and its future evolution of the process [Ast65].

At a high level, the algorithm propagates *forward messages* $\boldsymbol{\alpha}^{(t)}$ from the start of the sequence forward, gathering evidence along the way; it uses a similar process to propagate *backward messages* $\boldsymbol{\beta}^{(t)}$ in the reverse direction. Let $S^{(t)}$ and $R^{(t)}$ denote the random variables for the state and the response of the process at time t, respectively. Let also $r^{(1)}, ..., r^{(T)}$ be the observed sequence of responses in the data. With respect to $\mathbf{B}_\star$, the forward and backward messages at time t are defined as the measures over

$dom[S^{(t)}]$ obtained from the following conditional distributions:

$$\begin{aligned}
\boldsymbol{\alpha}^{(t)}[S^{(t)}] &= \boldsymbol{P}_{\mathbf{B}_\star}[S^{(t)}|r^{(1)},...,r^{(t)}] \\
\boldsymbol{\beta}^{(t)}[S^{(t)}] &= \boldsymbol{P}_{\mathbf{B}_\star}[r^{(t+1)},...,r^{(T)}|S^{(t)}] \ .
\end{aligned}$$

It is also useful to define a decomposition of $\mathbf{B}_\rightarrow$ into a Markovian state transition model $\mathcal{S}$ and an instantaneous observation model $\mathcal{R}$, such that:

$$\mathbf{B}_\rightarrow[S^\triangleright, R^\triangleright|S^\triangleleft] = \mathcal{S}[S^\triangleright|S^\triangleleft]\,\mathcal{R}[R^\triangleright|S^\triangleright] \ .$$

Then, the update rules for the $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ messages are easily obtained:

$$\begin{aligned}
&\boldsymbol{\alpha}^{(t)}[s] \\
&\quad= k_{\boldsymbol{\alpha}}^{(t)} \sum_{s' \in dom[S^{(t-1)}]} \boldsymbol{\alpha}^{(t-1)}[s']\,\mathbf{B}_\rightarrow[R^{(t)} = r^{(t)}, S^{(t)} = s|S^{(t-1)} = s'] \\
&\quad= k_{\boldsymbol{\alpha}}^{(t)} \sum_{s' \in dom[S^{(t-1)}]} \boldsymbol{\alpha}^{(t-1)}[s']\,\mathcal{S}[S^{(t)} = s|S^{(t-1)} = s']\,\mathcal{R}[R^{(t)} = r^{(t)}|S^{(t)} = s] \ ,
\end{aligned}$$

$$\begin{aligned}
&\boldsymbol{\beta}^{(t)}[s] \\
&\quad= k_{\boldsymbol{\beta}}^{(t)} \sum_{s' \in dom[S]} \boldsymbol{\beta}^{(t+1)}[s']\,\mathbf{B}_\rightarrow[R^{(t+1)} = r^{(t+1)}, S^{(t+1)} = s'|S^{(t)} = s] \\
&\quad= k_{\boldsymbol{\beta}}^{(t)} \sum_{s' \in dom[S]} \boldsymbol{\beta}^{(t+1)}[s']\,\mathcal{S}[R^{(t+1)} = r^{(t+1)}|S^{(t+1)} = s']\,\mathcal{R}[S^{(t+1)} = s'|S^{(t)} = s] \ ,
\end{aligned}$$

where the constants $k_{\boldsymbol{\alpha}}^{(t)}$ and $k_{\boldsymbol{\beta}}^{(t)}$ are chosen so that $\boldsymbol{\alpha}^{(t)}$ and $\boldsymbol{\beta}^{(t)}$ are normalized to sum to one. From there, the posterior distribution $\boldsymbol{\mu}^{(t)}[S^{(t)}] = \boldsymbol{P}_{\mathbf{B}_\star}[S^{(t)}|\mathbf{D}]$ over the states at time t given the entire data sequence is now simply given by:

$$\boldsymbol{\mu}^{(t)}[s] \propto \boldsymbol{\alpha}^{(t)}[s]\,\boldsymbol{\beta}^{(t)}[s] \ ,$$

where the proportionality relation $\propto$ hides a normalization constant, as above. Similarly, the joint posterior $\boldsymbol{\mu}^{(t,t+1)}[S^{(t-1)}, S^{(t)}] = \boldsymbol{P}_{\mathbf{B}_\star}[S^{(t-1)}, S^{(t)}|\mathbf{D}]$ over any pair of states at consecutive times $t-1$ and t, is:

$$\begin{aligned}&\boldsymbol{\mu}^{(t-1,t)}[s^{\triangleleft}, s^{\triangleright}]\\ &\quad\propto \quad \boldsymbol{\alpha}^{(t-1)}[s^{\triangleleft}]\,\boldsymbol{\beta}^{(t)}[s^{\triangleright}]\,\mathbf{B}_{\rightarrow}[R^{(t)} = r^{(t)}, S^{(t)} = s^{\triangleright}|S^{(t-1)} = s^{\triangleleft}]\ .\end{aligned} \tag{7.2}$$

In summary, the forward-backward algorithm[1] works by propagating a pair of messages $\boldsymbol{\alpha}$ and β from either end of the data sequence, computing the intermediate messages $\boldsymbol{\alpha}^{(t)}$ and $\boldsymbol{\beta}^{(t)}$ for each t along the way. These messages are cached in memory; when the propagation passes are complete, the messages are merged as above, to obtain the posteriors $\boldsymbol{\mu}^{(t,t+1)}[S^{(t-1)}, S^{(t)}]$, from which the probability of the various instantaneous events $E^{(t)}$ can be easily computed by marginalization, and accumulated into the desired expected sufficient statistics E. In practice, only one message sequence needs to be pre-cached, say $\boldsymbol{\alpha}$, as the computation of the various $\boldsymbol{\mu}^{(t-1,t)}$ may immediately follow that of the corresponding $\boldsymbol{\beta}^{(t)}$, for $t = 1, ..., T$.

This procedure offers an effective way to compute all the desired ESS in only two inference passes over the data, one going forward, the other going backward, and is the implementation of choice of the E-step of the parametric and structural EM algorithm for dynamic systems.

[1]For the sake of completeness, it should be noted that the foregoing description differs somewhat from the usual formalization of the algorithm. In the traditional descriptions of Baum and Welch [Bau72] and Rabiner and Juang [RJ86], the forward message is defined as the joint distribution $\boldsymbol{P}[S^{(t)}, r^{(1)}, ..., r^{(t)}]$ instead of the conditional $\boldsymbol{P}_{\mathbf{B}_\star}[S^{(t)}|r^{(1)}, ..., r^{(t)}]$, which we will find more convenient for our purpose. The two definitions lead to slightly different but equivalent algorithms.

Chapter 8

Efficient learning

One major difficulty of the learning procedure outlined in the previous chapter, is its dependence on the lengthy statistics estimation procedure in the case of partially observable systems; indeed, this procedure typically needs to be carried out not once but during several iterations of the EM algorithm, each iteration requiring a complete forward and backward pass over the entire data sequence. This is expensive on two counts. As should be obvious from Section 7.4, the forward and backward message propagation is nothing other than probabilistic inference on a Markov chain, which, as we have seen in Section 4.1, is almost invariably intractable for large complex systems, such as Markovian DBNs with many state variables. The second source of inefficiency is not nearly as dramatic, although its scope is slightly more subtle to analyze. The complexity of a single iteration of EM is seen to increase linearly with the size of the learning sequence, which may be a problem for voluminous data sets. As a mitigating factor, one would hope that the more accurate statistics resulting from a massive data set would lead to more effective M-steps, thereby reducing the total number of iterations of EM until convergence. Unfortunately, this is not quite the case, as an abundance of data typically does not help much until the very last few iterations, when it allows fine-tuning a model that has already almost converged.

The main focus of this chapter is to study how the contraction phenomenon in stochastic systems provides the key to overcoming both sources of inefficiency, and achieving efficient learning of structured stochastic processes—either by approximating the costly inference procedure itself, or by neglecting some of the data when warranted, or both.

8.1 Approximate parametric learning

A natural approach to the problem of the E-step intractability is to substitute our approximate inference algorithm for the propagation of the forward and backward messages. As described in Chapter 4, this algorithm avoids the problem of explicitly maintaining distributions over large spaces, by maintaining approximate belief states from a class of compactly representable probability distributions.

8.1.1 Approximate forward-backward propagation

The applicability of approximate inference to forward message propagation is immediate: similarly to the forward propagation principle, our approximate inference algorithm propagates a time-$(t-1)$ approximate distribution through the transition model, and conditions it on the evidence at time t; the resulting time-t distribution is approximated to produce a compactly representable time-t approximate distribution, allowing the algorithm to continue. In the case of DBNs, the approximation at each step consists of a simple projection onto a set of selected marginals or clusters, which determine a factored representation of the approximate belief state.

Forward propagation is easy. As previously detailed in Section 4.2, the approximate propagation algorithm can be implemented efficiently using the junction tree algorithm [LS88]. To compute $\tilde{\boldsymbol{\alpha}}^{(t)}$ from $\tilde{\boldsymbol{\alpha}}^{(t-1)}$, a junction tree over these two time slices is generated from the current DBN model, ensuring that each selected approximation cluster at time $t-1$ and time t appears as a subset of some clique in the junction tree. The previous message $\tilde{\boldsymbol{\alpha}}^{(t-1)}$ is incorporated into the tree, simply by multiplying the relevant cliques by the clusters of $\tilde{\boldsymbol{\alpha}}^{(t-1)}$. The next message $\tilde{\boldsymbol{\alpha}}^{(t)}$ is then easily obtained by calibrating the tree and reading off the relevant marginals from the appropriate cliques of the tree, $\tilde{\boldsymbol{\alpha}}^{(t)}$ being implicitly defined as their product.

This algorithm is directly applicable to the learning task, as the belief state used for inference corresponds exactly to the forward message used in the forward-backward algorithm. In turn, the contraction theorems give us confidence that the approximate forward messages will indefinitely stay close to the true forward messages.

The case of backward propagation is slightly more subtle. In itself, the above technique does not completely resolve the computational problems associated with the E-step, as the backward propagation is as expensive as the forward one. Fortunately, the computation of backward messages can be approximated similarly to that of forward messages, and carries the same computational benefits. It is also straightforward to adapt the approximate (forward) inference mechanism of Algorithm 4.5 or 4.8 to backward propagation, literally by taking the mirror image, substituting the anterior time slice for the ulterior one, and *vice versa*.

Specifically, to compute the backward message $\tilde{\boldsymbol{\beta}}^{(t)}$ from $\tilde{\boldsymbol{\beta}}^{(t+1)}$, we first construct a junction tree Υ from the current DBN model, giving indices t and $t+1$ to the anterior and ulterior sides of Υ, respectively. The tree Υ is constructed so that each cluster of $\tilde{\boldsymbol{\beta}}^{(t)}$ and $\tilde{\boldsymbol{\beta}}^{(t+1)}$ is fully contained in at least one clique of the tree. (For details, see the description of Algorithms 4.5 and 4.8 in Chapter 4, where an analogous condition appears). The second operation consists of incorporating the incoming message $\tilde{\boldsymbol{\beta}}^{(t+1)}$ into the ulterior $(t+1)$ side of Υ, to give Υ_t. For non-overlapping clusters, this is simply done by multiplying each cluster of $\tilde{\boldsymbol{\beta}}^{(t+1)}$ into any compatible clique of Υ. For overlapping clusters, the procedure must also deal with the sepset potentials present in $\tilde{\boldsymbol{\beta}}^{(t+1)}$, and is in all respects analogous to that described in Algorithm 4.8. Then, the resulting tree Υ_t is calibrated. Finally, the desired approximate backward message $\tilde{\boldsymbol{\beta}}^{(t)}$ is obtained in factored form simply by extracting the relevant marginals of Υ_t.

To reiterate, the only difference with the forward case is that, for backward propagation, the input and output sides of the junction tree are reversed. Considering a transition model with an anterior and an ulterior time slice, whereas in forward propagation we multiply out the transition model with the previous message on the anterior side, and read out the new message from the ulterior side, in backward propagation we multiply out the incoming message into the ulterior side of the transition model, calibrate the junction tree, and read out the new message from the anterior side by marginalization.

8.1.2 Bidirectional contraction

Extending the contraction analysis is not as immediate. It is not completely straightforward to apply the techniques of Chapter 5 to obtain relative entropy bounds for the backward message, which is not defined as a conditional distribution. Furthermore, even if bounds on relative entropy error were available for both the forward and backward messages, error bounds for the combined approximate posterior $\tilde{\boldsymbol{\mu}}^{(t)}$ would not necessarily follow. In fact, counterexamples may be found, in which the relative entropy error of the approximate forward and backward messages may be as low as we wish, and still combine into an approximate posterior that has arbitrarily large error with respect to its exact counterpart.

Example 8.1 Consider a process with state space $\mathbf{S} = \{s_0, s_1, s_2\}$, and let the true and approximate forward and backward messages be as follows:

$$\begin{aligned} \boldsymbol{\alpha}^{(t)} &= [0, 1-\epsilon, \epsilon] & \qquad \boldsymbol{\beta}^{(t)} &= [0, \epsilon, 1-\epsilon] \\ \tilde{\boldsymbol{\alpha}}^{(t)} &= [\epsilon, 1-\epsilon-\epsilon^3, \epsilon^3]\ , & \qquad \tilde{\boldsymbol{\beta}}^{(t)} &= [\epsilon, \epsilon^3, 1-\epsilon-\epsilon^3]\ . \end{aligned}$$

The relative entropy error in the forward and backward case is, respectively:

$$\mathbb{D}_{KL}[\boldsymbol{\alpha}^{(t)} \,\|\, \tilde{\boldsymbol{\alpha}}^{(t)}] \simeq \epsilon\ , \qquad \mathbb{D}_{KL}[\boldsymbol{\beta}^{(t)} \,\|\, \tilde{\boldsymbol{\beta}}^{(t)}] \simeq \epsilon\ .$$

Now, if we merge the forward and backward messages, we obtain, after normalization:

$$\begin{aligned} \boldsymbol{\mu}^{(t)} &= [0, \frac{1}{2}, \frac{1}{2}] \\ \tilde{\boldsymbol{\mu}}^{(t)} &\simeq [1, \epsilon, \epsilon]\ . \end{aligned}$$

The relative entropy error that results amounts to:

$$\mathbb{D}_{KL}[\boldsymbol{\mu}^{(t)} \,\|\, \tilde{\boldsymbol{\mu}}^{(t)}] \simeq log_2\,[\frac{1}{2\,\epsilon}]\ .$$

In conclusion, the arbitrarily small forward and backward error has turned into an arbitrarily large one for the combined belief state.

∎

The above phenomenon occurs because the combination operation, as described in Section 7.4, is akin to Bayesian conditioning; and, while relative entropy will not increase on expectation under Bayesian conditioning, nothing can be said *a priori* when conditioning on arbitrary events. In spite of this, the posterior distribution resulting from the forward-backward procedure is still subject to useful contraction properties, albeit not nearly as strong ones as those associated with filtering.

The key to analyzing this form of contraction turns out to be found in an alternative notion of error, called *projective distance*, which combines additively under Bayesian updating, and is defined as follows.

Definition 8.2 Let φ and ψ be two measures over the same discrete space Ω. Their *projective distance* is defined as:

$$\mathbb{D}_{proj}[\boldsymbol{\varphi} \,\|\, \boldsymbol{\psi}] \quad = \quad \max_{\omega_i, \omega_j \in \Omega} \, log_2 \frac{\boldsymbol{\varphi}[\omega_i] \cdot \boldsymbol{\psi}[\omega_j]}{\boldsymbol{\varphi}[\omega_j] \cdot \boldsymbol{\psi}[\omega_i]} \; .$$

Projective distance bears some resemblance to relative entropy, with the main difference of having a worst-case semantics attached to it, which contrasts with the average-case flavor of relative entropy. We also note the following facts.

Fact 8.3 *For any two distributions φ and ψ over the same discrete space Ω:*

$$\mathbb{D}_{KL}[\boldsymbol{\varphi} \,\|\, \boldsymbol{\psi}] \quad \geq \quad \mathbb{D}_{proj}[\boldsymbol{\varphi} \,\|\, \boldsymbol{\psi}] \; .$$

Fact 8.4 *Projective distance is scale-invariant, i.e., $\forall x > 0, y > 0$:*

$$\mathbb{D}_{proj}[x\,\boldsymbol{\varphi} \,\|\, y\,\boldsymbol{\psi}] \quad = \quad \mathbb{D}_{proj}[\boldsymbol{\varphi} \,\|\, \boldsymbol{\psi}] \; .$$

As previously mentioned, projective distance combines additively under Bayesian conditioning, leading to the following lemma.

Lemma 8.5 *For forward, backward, and combined messages and belief states as previously defined:*

$$\mathbb{D}_{proj}[\boldsymbol{\mu}^{(t)} \,\|\, \tilde{\boldsymbol{\mu}}^{(t)}] \;\leq\; \mathbb{D}_{proj}[\boldsymbol{\alpha}^{(t)} \,\|\, \tilde{\boldsymbol{\alpha}}^{(t)}] + \mathbb{D}_{proj}[\boldsymbol{\beta}^{(t)} \,\|\, \tilde{\boldsymbol{\beta}}^{(t)}] \;.$$

Proof The result follows easily from the definition of projective distance; *in extenso*:

$$\begin{aligned}
\mathbb{D}_{proj}[\boldsymbol{\mu}^{(t)} \,\|\, \tilde{\boldsymbol{\mu}}^{(t)}] &= \mathbb{D}_{proj}[\boldsymbol{\alpha}^{(t)} \otimes \boldsymbol{\beta}^{(t)} \,\|\, \tilde{\boldsymbol{\alpha}}^{(t)} \otimes \tilde{\boldsymbol{\beta}}^{(t)}] \\
&= \max_{\omega_i,\omega_j} \log_2 \frac{\boldsymbol{\alpha}^{(t)}[\omega_i] \cdot \boldsymbol{\beta}^{(t)}[\omega_i] \cdot \tilde{\boldsymbol{\alpha}}^{(t)}[\omega_j] \cdot \tilde{\boldsymbol{\beta}}^{(t)}[\omega_j]}{\boldsymbol{\alpha}^{(t)}[\omega_j] \cdot \boldsymbol{\beta}^{(t)}[\omega_j] \cdot \tilde{\boldsymbol{\alpha}}^{(t)}[\omega_i] \cdot \tilde{\boldsymbol{\beta}}^{(t)}[\omega_i]} \\
&\leq \max_{\omega_i,\omega_j} \log_2 \frac{\boldsymbol{\alpha}^{(t)}[\omega_i] \cdot \tilde{\boldsymbol{\alpha}}^{(t)}[\omega_j]}{\boldsymbol{\alpha}^{(t)}[\omega_j] \cdot \tilde{\boldsymbol{\alpha}}^{(t)}[\omega_i]} + \max_{\omega_i,\omega_j} \log_2 \frac{\boldsymbol{\beta}^{(t)}[\omega_j] \cdot \tilde{\boldsymbol{\beta}}^{(t)}[\omega_i]}{\boldsymbol{\beta}^{(t)}[\omega_j] \cdot \tilde{\boldsymbol{\beta}}^{(t)}[\omega_i]} \\
&\leq \mathbb{D}_{proj}[\boldsymbol{\alpha}^{(t)} \,\|\, \tilde{\boldsymbol{\alpha}}^{(t)}] + \mathbb{D}_{proj}[\boldsymbol{\beta}^{(t)} \,\|\, \tilde{\boldsymbol{\beta}}^{(t)}] \;.
\end{aligned}$$

$\square$

Based on the results of [AL95], we show that the projective distance contracts under message propagation through the stochastic transition as above in either direction. In other words, the accumulated error, quantified as a projective distance to the exact message, undergoes an exponential decay. The contraction ratio depends on the minimal ergodicity properties of the stochastic transition matrix $\mathcal{S}$ representing the entire transition model, as follows.

Lemma 8.6 *[AL95, Theorem 2.1]*

Let $\mathcal{S}$ be a stochastic transition matrix each of whose columns has at least one non-zero element. Define:

$$\kappa_{\mathcal{S}} \;=\; 2\,\frac{k}{1+k} \;,$$

where

$$k \;=\; \min_{\{i,j,i',j' : \mathcal{S}_{i,j}\,\mathcal{S}_{i',j'} \neq 0\}} \sqrt{(\mathcal{S}_{i,j'}\,\mathcal{S}_{i',j})/(\mathcal{S}_{i,j}\,\mathcal{S}_{i',j'})} \;.$$

Then, for $\boldsymbol{\alpha}^{(t)}$ *and* $\boldsymbol{\beta}^{(t)}$ *as previously defined:*

$$\mathbb{D}_{proj}[\boldsymbol{\alpha}^{(t)} \,\|\, \tilde{\boldsymbol{\alpha}}^{(t)}] \;\leq\; (1-\kappa_{\mathcal{S}})\, \mathbb{D}_{proj}[\boldsymbol{\alpha}^{(t-1)} \,\|\, \tilde{\boldsymbol{\alpha}}^{(t-1)}] \;,$$

and

$$\mathbb{D}_{proj}[\boldsymbol{\beta}^{(t)} \,\|\, \tilde{\boldsymbol{\beta}}^{(t)}] \;\leq\; (1-\kappa_{\mathcal{S}})\, \mathbb{D}_{proj}[\boldsymbol{\beta}^{(t+1)} \,\|\, \tilde{\boldsymbol{\beta}}^{(t+1)}] \;.$$

Proof The lemma is a restatement of [AL95, Theorem 2.1]. ◻

We can now combine Lemmas 8.5 and 8.6 with the facts that the projective distance obeys the triangle inequality and does not increase under Bayesian conditioning, to show that, if our approximations do not introduce too large an error, then the expected sufficient statistics will remain close to their correct values.

Theorem 8.7 *Let* $\bar{N}$ *and* $\tilde{N}$ *be the expected sufficient statistics as respectively computed using exact and approximate forward and backward propagation, on the same model and with the same data. If each forward and backward approximation step is guaranteed to increase the projective error by, respectively, at most* $\epsilon_{\boldsymbol{\alpha}}$ *and* $\epsilon_{\boldsymbol{\beta}}$*, then:*

$$\mathbb{D}_{proj}[\bar{N} \,\|\, \tilde{N}] \;\leq\; \frac{\epsilon_{\boldsymbol{\alpha}} + \epsilon_{\boldsymbol{\beta}}}{\kappa_{\mathcal{S}}} \;.$$

Proof Applying an argument similar to the proof of Theorem 5.14 to Lemma 8.6, using the fact that the projective error does not increase under Bayesian conditioning on observation, it is easy to see that the total projective error for the forward and backward messages is indefinitely bounded by, respectively:

$$\begin{aligned}
\mathbb{D}_{proj}[\boldsymbol{\alpha}^{(t)} \,\|\, \tilde{\boldsymbol{\alpha}}^{(t)}] &\;\leq\; \frac{\epsilon_{\boldsymbol{\alpha}}}{\kappa_{\mathcal{S}}} \\
\mathbb{D}_{proj}[\boldsymbol{\beta}^{(t)} \,\|\, \tilde{\boldsymbol{\beta}}^{(t)}] &\;\leq\; \frac{\epsilon_{\boldsymbol{\beta}}}{\kappa_{\mathcal{S}}} \;.
\end{aligned}$$

Combining the above results as per Lemma 8.5, the claim follows easily. ◻

Since the projective distance is an upper bound on relative entropy, the next corollary follows easily.

Corollary 8.8 *Under the above assumptions, and further assuming that $\bar{N}$ and $\tilde{N}$ are normalized to form valid distributions:*

$$\boldsymbol{D}_{KL}[\bar{N} \,\|\, \tilde{N}] \quad \leq \quad \frac{\epsilon_{\boldsymbol{\alpha}} + \epsilon_{\boldsymbol{\beta}}}{\kappa_{\mathcal{S}}} \ .$$

It is emphasized that the above results should be interpreted carefully. First, the contraction ratio for the forward-backward procedure is not nearly as strong as the one obtained in Chapter 5 for the filtering task. As already noted, this is due to the fact that the error is measured as a projective distance, which captures the largest discrepancy between all states, even if the probability of reaching those states is minute. This worst-case behavior is needed to cope with the bidirectional nature of the forward-backward procedure, which may amplify the probability of some states by an arbitrary amount, as we have seen in Example 8.1. It should also be noted that the KL divergence bound given in Corollary 8.8, being numerically equal to the projective distance bound of Theorem 8.7, is actually less useful. Second, the ESS guarantee of accuracy as provided by Theorem 8.7 does not necessarily translate to a similar guarantee of accuracy for the entire learning process. Indeed, even small fluctuations in the sufficient statistics can potentially cause the EM algorithm to reach a substantially different local maximum, for better or for worse. For all those reasons, the above results should be interpreted as a justification of principle for our approximate learning algorithm, as opposed to an analytical characterization of its quality.

8.1.3 Experimental resutls

Two series of experiments were conducted to assess the effectiveness of learning using approximate inference. In a first set of experiments, we provide a more qualitative comparison between approximate and exact inference, and show that, even though the former approach is orders of magnitude faster, the learning curves are almost identical in both cases. In a second set of experiments, we further strengthen the statistical significance of the conclusions herein, using results from a larger pool of learning instances.

In this first series of experiments, we evaluated our algorithm on the task of learning the parameters of the previously encountered BAT network [FHKR95], shown in Figure 4.2. The task was to learn the parameters of the network, given the correct network structure and a sequence of synthetic data composed of observations sampled from the correct network. The training set was a fixed sequence of 1000 time slices, sampled from the correct model. An independent sequence of 50 time slices was also generated for the test set. Our evaluation metric was defined as the average log-likelihood of the test sequence, per time slice, with respect to the learned model.

The objective of the experiments was to evaluate various kinds of structural approximation schemes for forward-backward message propagation, in comparison with the standard exact algorithm. As in our previous inference experiments with the BAT network from Section4.3, we selected four different structural approximations, as shown in Figure 4.2: (i) one single cluster corresponding to exact propagation; (ii) a 5+5 clustering of the ten state variables; (iii) a 3+2+4+1 clustering of the state variables; (iv) each state variable in a separate cluster, corresponding to the most aggressive approximation. For each approximation, we learned the parameters of the transition model given the correct network structure and the same learning sequence, by applying the EM algorithm until perceived convergence. To mitigate the impact of the sensitivity of EM to initial conditions, three different sets of random starting values were generated, on which each of the four experiments was then conducted. For each approximation, the quality of the learned parameters was evaluated after each iteration of the EM algorithm. The resulting learning curves, for one of the three sets of initial parameters, are plotted in Figure 8.1.

As can be observed from Figure 8.1, the learning curves for exact learning and the various approximations are almost identical; even severe structural approximation is seen to have negligible impact on learning accuracy. Qualitatively similar learning curves were obtained for the other two sets of starting parameters. In all cases, the various approximations tracked the exact algorithm very closely, and the largest

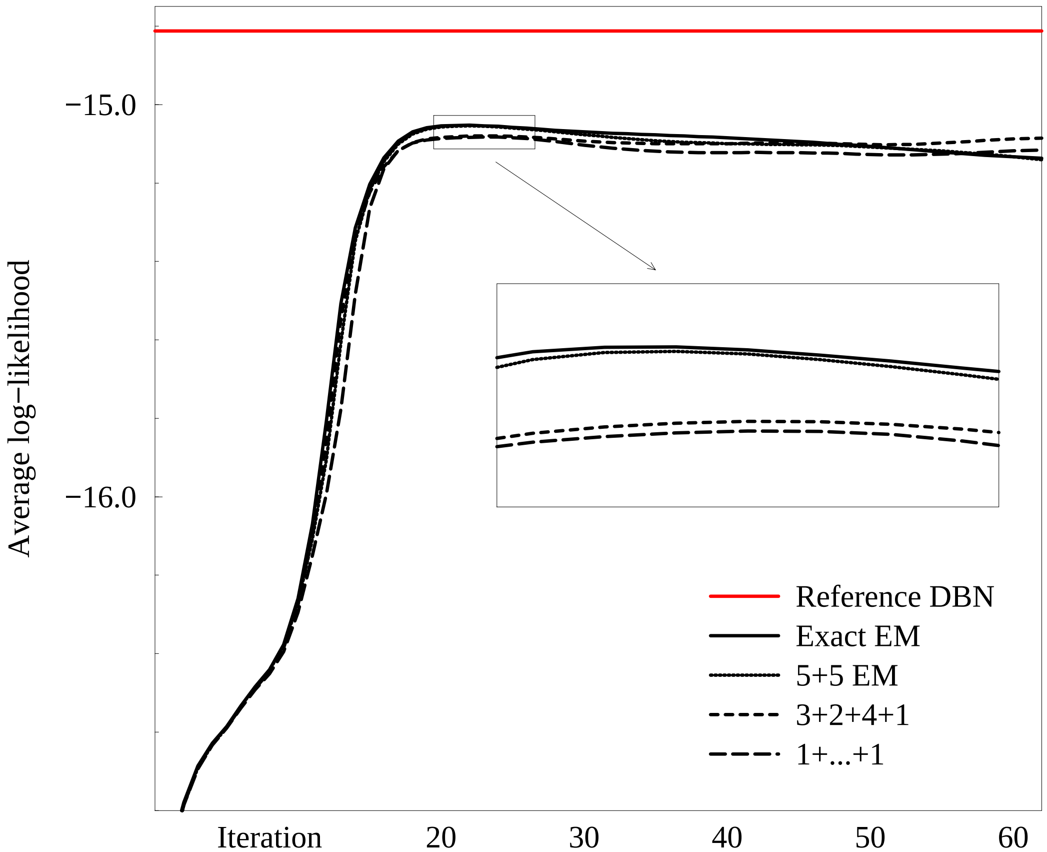


Figure 8.1: Compared quality of various structural approximations for parametric learning with EM. The horizontal line represents the quality of the original model used for sampling the data.

difference in the peak log-likelihood was much smaller than 0.1 bit per slice. This phenomenon is rather remarkable, especially in view of the substantial savings due to the approximations: in our implementation, a 23-fold speedup over exact learning was achieved using the 5+5 clustering, and over 27-fold with the other two, more aggressive approximations.

In the second series of experiments, we conducted a large number of automated learning tests for various approximation schemes, and differing random data sets and starting model parameters. We conducted two such sets of experiments, one with the now familiar BAT network, and another with the WATER network modified for partial observability as in Section 4.3. For both BAT and WATER, we randomly generated between 10 and 20 learning instances. Each instance was composed of a set of starting model parameters, and three independent data sequences sampled from the reference BAT and WATER networks: namely, a 1000-point training set, a 100-point validation set, and a 100-point independent test set. The training set was used for learning, using the previously described algorithm, for up to 50 iterations of EM. At the end of each iteration, the quality of the then-current model was evaluated using the validation set. At the end of the learning process, all models produced at the end of the various EM iterations were evaluated on the validation set, and the best model selected. The best model was then evaluated using the independent test data set, producing a single "figure of merit" for that learning instance. This was done for all approximation schemes using the same data sets and starting parameters. The process was then repeated for all learning instances.

Table 8.1 details the results of our experiments with 20 independent learning instances for the BAT network, using 7 different approximation configurations. In each case, a sequence of models is learned from a first training set; the best model is determined using a second validation set; the accuracy is then evaluated on a third independent test set. Each learning instance corresponds to a different collection of random starting parameters, and independently sampled data sets using the original network parameters. Referring to Figure 4.2 in Section 4.3, the approximation schemes used in the present experiments are as follows:

- One exact inference scheme, denoted "Exact": this case is obtained by requiring all 10 state variables to belong to a single cluster;
- Three approximate schemes with non-overlapping clusters: these are the '5+5' and '3+2+4+1' clusters represented on Figure 4.2, and the trivial '1+1+...+1' clustering which places each state variable in a cluster of its own;
- Three approximate schemes with overlapping clusters, as follows:
  - '7+7' has two overlapping clusters of 7 variables each, namely:

Table 8.1: Fitness of learned BAT models on independent test data, for various combinations of approximation schemes and data sets. The numbers reported are the negative log-likelihood in bits/time slice (lower is better), along with the iteration number of the model selected by the algorithm.

BAT network	Approximate inference clustering scheme						
	Exact	Overlapping clusters			Non-overlapping clusters		
Instance	10	7+7	7+6+1	6+5	5+5	3+...+1	1+...+1
#1	0.752 @ 17	0.750 @ 17	0.756 @ 17	0.753 @ 17	0.758 @ 17	0.746 @ 17	0.746 @ 17
#2	1.164 @ 49	1.220 @ 20	1.220 @ 20	1.221 @ 20	1.220 @ 20	1.223 @ 20	1.083 @ 41
#3	0.363 @ 19	0.370 @ 19	0.365 @ 19	0.370 @ 19	0.370 @ 19	0.365 @ 19	0.376 @ 19
#4	0.122 @ 26	0.129 @ 26	0.132 @ 26	0.125 @ 26	0.128 @ 26	0.135 @ 27	0.156 @ 27
#5	0.336 @ 28	0.322 @ 29	0.321 @ 29	0.319 @ 29	0.321 @ 29	0.309 @ 31	0.323 @ 31
#6	0.168 @ 16	0.170 @ 15	0.170 @ 15	0.165 @ 16	0.168 @ 15	0.168 @ 15	0.172 @ 14
#7	0.384 @ 50	0.281 @ 35	0.253 @ 27	0.369 @ 50	0.250 @ 26	0.288 @ 24	0.378 @ 39
#8	0.732 @ 25	0.732 @ 25	0.743 @ 26	0.710 @ 22	0.710 @ 22	0.713 @ 22	0.713 @ 23
#9	0.667 @ 50	0.586 @ 50	0.726 @ 50	0.694 @ 50	0.616 @ 50	0.833 @ 50	1.224 @ 50
#10	0.343 @ 29	0.337 @ 26	0.343 @ 26	0.347 @ 28	0.340 @ 26	0.370 @ 27	0.369 @ 27
#11	0.809 @ 13	0.858 @ 12	0.858 @ 12	0.809 @ 13	0.858 @ 12	0.817 @ 13	0.825 @ 13
#12	0.137 @ 28	0.137 @ 29	0.130 @ 28	0.126 @ 28	0.209 @ 39	0.079 @ 28	0.117 @ 29
#13	0.987 @ 35	1.018 @ 36	1.007 @ 35	0.984 @ 34	0.979 @ 34	0.830 @ 27	0.860 @ 30
#14	0.733 @ 23	0.734 @ 23	0.734 @ 23	0.734 @ 23	0.733 @ 23	0.723 @ 23	0.721 @ 22
#15	0.676 @ 15	0.680 @ 15	0.681 @ 15	0.676 @ 15	0.680 @ 15	0.691 @ 14	0.683 @ 15
#16	0.595 @ 25	0.594 @ 25	0.594 @ 25	0.596 @ 25	0.593 @ 24	0.575 @ 25	0.568 @ 25

Table 8.2: Summary statistics for the results of Table 8.1. For each approximation, aggregates are computed over all instances, using the difference between the negative log-likelihoods for the approximation and the exact method, for matching instances. (Negative averages indicate better-performing approximations.)

BAT network		Approximate inference clustering scheme					
	Exact	Overlapping clusters			Non-overlapping clusters		
Instance	10	7+7	7+6+1	6+5	5+5	3+...+1	1+...+1
	actual data	pairwise differences Δ(approximate − exact)					
average	0.5605	-0.0031	0.0041	0.0019	-0.0022	-0.0064	0.0216
std. dev.	0.3080	0.0399	0.0423	0.0185	0.0463	0.0685	0.1482

(LeftClr–RightClr–LatAction–Xdot–InLane–FwdAction–Ydot), and
(Xdot–InLane–FwdAction–Ydot–Stopped–EngStatus–FrontBackStatus);

– ‘7+6+1’ has three clusters, including two that overlap:
(LeftClr–RightClr–LatAction–Xdot–InLane–FwdAction–Ydot),
(Xdot–InLane–FwdAction–Ydot–Stopped–EngStatus), and
(FrontBackStatus);

– ‘6+5’ has two clusters with a one-variable overlap:
(LeftClr–RightClr–LatAction–Xdot–InLane–FrontBackStatus), and
(FwdAction–Ydot–Stopped–EngStatus–FrontBackStatus).

Summary statistics for the results shown in Table 8.1 are provided in Table 8.2. The ‘Exact’ column shows the mean and (sample) standard deviation of the negative log-likelihood for learning using exact inference. The other columns show the means and standard deviations of the differences between the each approximation case and the exact case; a negative value of the average difference in a given column indicates that the corresponding approximation produced models more acccurate than the exact learner, on independent test data. A first observation is that all approximation methods, with the possible exception of the ‘1+1+...+1’ clustering, produce models that are almost exactly as accurate as the exact method. The ‘1+1+...+1’ clustering appears to behave slightly less predictably. Although the magnitude of the average

Table 8.3: Fitness of learned WATER models on independent test data, for various combinations of approximation schemes and data sets. The numbers reported are the negative log-likelihood in bits/time slice, and the iteration number of the model selected by the algorithm. (Experiments using the extremely expensive exact inference were only conducted on a subset of the learning instances.)

WATER network	Approximate inference clustering scheme			
Instance (seed / data sets)	Exact	2+4+2	2+2+2+2	1+1+...+1
#1	0.426 @ 18	0.650 @ 46	0.633 @ 45	0.478 @ 49
#2	0.555 @ 19	0.536 @ 50	0.565 @ 50	0.492 @ 50
#3	0.324 @ 17	0.313 @ 50	0.224 @ 50	0.194 @ 50
#4	0.214 @ 22	0.216 @ 50	0.205 @ 48	0.167 @ 50
#5	0.291 @ 18	0.360 @ 50	0.153 @ 50	0.174 @ 50
#6	0.191 @ 17	0.369 @ 50	0.100 @ 50	0.161 @ 50
#7	0.706 @ 14	0.764 @ 14	0.762 @ 14	0.766 @ 14
#8	1.164 @ 19	1.032 @ 17	1.045 @ 17	1.019 @ 16
#9	0.417 @ 28	0.467 @ 50	0.602 @ 50	0.563 @ 50
#10	0.409 @ 27	0.215 @ 35	0.191 @ 50	0.234 @ 49
#11	n/a	0.633 @ 50	0.511 @ 50	0.504 @ 50
#12	n/a	0.781 @ 22	0.817 @ 19	0.801 @ 19
#13	n/a	0.337 @ 50	0.322 @ 50	0.264 @ 50
#14	n/a	0.922 @ 16	0.845 @ 16	0.853 @ 15
#15	n/a	0.793 @ 50	0.534 @ 50	0.491 @ 50
#16	n/a	0.258 @ 50	0.271 @ 50	0.252 @ 50
#17	n/a	0.286 @ 50	0.276 @ 50	0.286 @ 50
#18	n/a	0.772 @ 20	0.763 @ 21	0.789 @ 23
#19	0.574 @ 22	0.681 @ 19	0.817 @ 15	0.843 @ 15
#20	0.606 @ 20	0.435 @ 27	0.584 @ 41	0.626 @ 40

difference in accuracy with respect to the exact method remains small compared to its own standard deviation, a close inspection of Table 8.1 reveals that 2 out of the 16 cases are outliers: in case #2, the '1+1+...+1' appears to perform somewhat better than the competition, but fares significantly worse in case #9.

Table 8.3 shows our experimental results for the WATER network, on 20 independent learning instances; Table 8.4 gives a succinct summary of those results. As before, the model parameters are learned from a first training set; the best model is determined using a second validation set; the reported accuracy is then evaluated

Table 8.4: Summary statistics for the results of Table 8.3. For each approximation, aggregates are computed over all instances, using the difference between the negative log-likelihoods for the approximation and the exact method, for matching instances. (Negative averages indicate better-performing approximations.)

WATER network	Approximate inference clustering scheme			
Instance (seed / data sets)	Exact	2+4+2	2+2+2+2	1+1+...+1
	actual data	pairwise differences Δ(approximate − exact)		
average	0.4898	0.0134	0.0003	-0.0133
standard deviation	0.2653	0.1302	0.1474	0.1306

on a third independent test set. For this network, the approximation schemes are as follows:

- One exact scheme, denoted 'Exact', where all 8 state variables to belong to a single cluster (A-B-C-D-E-F-G-H);
- Three approximate schemes with non-overlapping clusters:
  - '2+4+2' corresponds to the three clusters (A-B) (C-D-E-F) (G-H);
  - '2+2+2+2' corresponds to the four clusters (A-B) (C-D) (E-F) (G-H), as represented on Figure 4.3;
  - '1+1+...+1' corresponds to the maximal approximation where each state variable is in its own cluster.

It is noted that, due to their computational cost, no overlapping cluster approximation experiments were conducted on this network, and exact inference experiments were restricted to a smaller number of learning instances.

Summary statistics for the results of Table 8.3 are given in Table 8.2. As with the BAT network, we observe that all three approximations seem to closely match the performance of the exact learner. In addition, the standard deviations of the accuracy differences with the exact methods are significantly smaller than the standard deviation of the exact method's own accuracy.

In conclusion, even though the test accuracies are not strictly equivalent across the board in response to a change of clustering, the fluctuations are small and appear non-systematic. In particular, the fluctuations caused by the clustering are completely dwarfed by the fluctuations caused by the randomization of the data sets, even though these are fairly large and sampled from the same distributions. This lends further credibility to the proposed approach.

8.2 Online learning

In addition to the subtantial computational gains already provided by approximate message propagation, our contraction analysis also gives us the tools to address another important problem with learning dynamic models: the need to reason about the entire temporal sequence at once.

8.2.1 Forgetting the distant future

From a computational perspective, reasoning with a large sequence of observations poses a number of problems. If the sequence is too large to keep in main memory, the execution of the forward-backward process can become seriously impaired. More importantly, in an online setting, the data set takes the form of a stream of observations as the process unfolds, so that at no point is the entire sequence ever available.

There have been several attempts to deal with either or both of these problems. Binder *et al.* [BMR97] address the memory problem with the help of a time-space tradeoff: some of the messages are cached, while the others are recomputed as needed. This approach, however, still requires a forward-backward propagation over the entire sequence, making it inapplicable to the online learning task. Neal and Hinton [NH98] address the online learning problem with an incremental update procedure for the sufficient statistics and to the model parameters. However, their solution applies only to independent data sets, where the inference step can be carried out separately on each independent data set, as it becomes available. This technique is very suited to learning static models from a large number of independent observations; however, when learning temporal models from a single ongoing data stream, the entire sequence would still be required for the E-step. In this section, we build on the intuition arising from our contraction analysis, to offer an approach that addresses both difficulties.

One consequence of the contraction phenomenon is that the error incurred by using approximations far away in the sequence decays exponentially with the time difference, whether in the past or in the future; in particular, the impact of an approximation which completely ignores all data far away in the future is also limited. Consequently, applying the forward-backward procedure to compute a posterior distribution at time t using only a small subset of the data sequence in the vicinity of t, should still provide fairly accurate results. More precisely, assume that we are considering a window of size w on both sides of the target time slice. A very bad approximation of the backward message at time $t + w$ that ignores all data beyond $t + w$, is given by the uniform message; however, as we propagate this approximate message backward from $t + w$ to t, the initial error decays exponentially in the width of the window w, and so will its impact on the posterior at time t. This observation, supported by our previous results, leads to a simple yet effective class of online learning algorithms for stochastic processes.

8.2.2 Incremental updates

An efficient approach to online learning is based on the *incremental EM* method of Neal and Hinton [NH98], in which the ESS are not recomputed *tabula rasa* during the E-step, but merely updated with the new data. Various strategies may be used for the update, one of the simplest and most effective being the exponential decay schedule, in which the new data are blended in fixed proportion with the previous ESS to yield updated ESS at each iteration. The M-step is unchanged; the model parameters are recomputed from the ESS by the usual formula, either at every time step, or, to save time, at larger intervals.

The main problem with frequent parameter updates in an online setting is that they require a recomputation of old messages each time the model parameters are updated. For long sequences, the computational cost of such recomputations would be prohibitive. Even with the windowed approach, recomputing $2\,w$ messages for each incremental ESS update remains costly.

Our algorithms use a more aggressive strategy to avoid unnecessary message computations. In essence, assuming a window width w, we maintain the "present" approximate posterior $\tilde{\boldsymbol{\mu}}^{(t-1,t)}$ so that it lags the received evidence by a delay of up to w. That is, we may postpone the computation of $\tilde{\boldsymbol{\mu}}^{(t-1,t)}$ until the evidence at time $t+w$ is observed. This allows us to employ efficient caching techniques in order to avoid recomputing too many messages at each time step. This is especially important for backward messages, as they cannot easily be updated when new evidence pours in.

8.2.3 Smart message caching

We now describe the strategies we employ to reduce the amortized number of message propagations per time slice, in order to speed up the learning process.

Forward messages are computed once and for all, and are never recomputed, even when the model parameters are updated. Instead, forward message propagation is performed lazily using a just-in-time strategy, so that the most recent parameters are used to obtain $\tilde{\boldsymbol{\mu}}^{(t)}$ from $\tilde{\boldsymbol{\mu}}^{(t-1)}$, itself computed one step earlier using the then-current parameters, and so on. This approach is justified by the contraction phenomenon and the slow parameter evolution from one step of incremental EM to the next, which together tend to ensure that the total error on forward messages remains bounded under this approximation.

Backward messages require a slightly different approach, since it is not possible to update a message $\tilde{\boldsymbol{\beta}}^{(t)}$ propagated backward from time $t+w-1$, to account for a new observation at time $t+w$. However, some savings can be made, provided that the window w is sufficiently large, and the rate of change in the model parameters sufficiently slow. Let t' be such that $t < t' \ll t+w$, and assume that a sequence of backward messages $\tilde{\boldsymbol{\beta}}^{(\cdot)}$ from time $t+w$ down to time t was computed when the time-$(t+w)$ observation became available. Under the above assumptions, the time-t' message $\tilde{\boldsymbol{\beta}}^{(t')}$ from this sequence still reasonably approximates the optimal backward message at time t', which suggests that a backward sequence, even when restricted to a window of limited size, does not have to be recomputed every single time a new observation is obtained. We experimented with two types of caching strategies, as follows.

Static backward message cache

The static caching strategy is fairly straightforward; for a window size w, we call it *static-w*. The principle is to maintain a cache of up to w backward messages, which is initially empty. When the cache is found empty at the beginning of an EM iteration, the algorithm pauses to compute all backward messages from up to w steps ahead of the current time, using the w evidence points in the lookahead buffer. This precomputation is done periodically, at any time τ such that $\tau \equiv 0 \pmod{w}$, using the current learned model at time τ. Then, during the E-step at any time t, the forward message is propagated normally from the time-$(t-1)$ message, using the latest model. However, the backward message is not recomputed; instead, it is simply pulled from the cache.

The main advantage of the static strategy is that, similar to the computation of forward messages, it only requires one backward propagation step on average per time slice of data. Its main drawback is that most backward messages will no longer correspond to a fresh model when they are actually used, in incremental EM. This is especially true for time indices t such that $t \bmod w \gg 0$. A second difficulty is that the few farthest messages in the cache (those at times $t \approx \tau + w$ in a series from τ to $\tau + w$) will be inaccurate, being based on very little evidence data. Unlike the first problem, the latter is more of an issue with small window sizes.

In our experiments with static caching, we used $w \in \{0, 1000\}$ with regular (non-incremental) EM, and $w \in \{0, 100, 1000\}$ with incremental EM as in Section 8.2.2.

Dynamic backward message cache

The dynamic caching strategy is an attempt to cure both shortcomings of the static cache, without resorting to recomputing w backward messages every time the model is updated in incremental EM. The general idea is to maintain a $\log w$-sized cache of backward messages, containing cached messages at geometrically increasing distances ahead of the current time, computed using models that are geometrically increasingly older–*i.e.*, the closer the cached message is with respect to the present time, the more recently it will have been computed. Thus, the invariant of the cache is that, at any time t, the cache contains cached messages for the times $t_i = 2^i \lceil t/2^i \rceil$, for $i = 0, ..., \lfloor \log w \rfloor$. Then, at every step t of incremental EM, the cached message closest to t is retrieved from the cache (*i.e.*, this is the message at time $t_i > t$ for the smallest available i), and is propagated backward using the current model parameters in order to obtain the needed backward message at time t. During this process, the cache is also updated to satisfy the invariant. In addition to the above, we introduce a supplemental "hold-off" parameter g, and require that only messages satifying $t_i > t + g$ be pulled from the cache. This is in order to guard against the few occasions where an ancient message (*i.e.*, for one of the larger values of i) would be pulled from the cache at a time t_i uncomfortably close to the current time t. In case the model parameters are only updated every f steps in incremental EM, the selection criterion becomes $t_i > t+g+f$, and only needs to be satisfied at the update times.

It is easy to see that this strategy incurs an amortized cost of about $1 + 1/f$ backward propagations per time slice, where f is the number of time slices between each model update in incremental EM. With the hold-off parameter, the cost factor becomes $1+(1+g)/f$. On the upside, this rather complex strategy combines efficiency with accuracy, and is a realistic alternative to a full update of backward messages at every step of incremental EM.

In our experiments with the dynamic caching strategy, we used the values $f = g = 4$ and $w \in \{4, 40\}$, with incremental EM.

8.2.4 Experimental results

The purpose of our first set of experiments is to get a sense of how the various update strategies described in Section 8.2.3 stack against each other for online learning. Here, we are equally interested in true online learning, in which each data point of a potentially unbounded sequence is seen at most once, as in what we refer to as offline learning, in which multiple passes are made over a given finite sequence, in a data stream fashion. Such a hybrid of online and offline algorithm is considered in, *e.g.*, [GMMO00].

We compare the following instances of the update strategies: *batch EM*, or basic EM without incremental update, as used in the experiments of Section 8.1.3; *dynamic-1000*, as our most careful approximation of a full update of the backward messages at every step, being both accurate and adaptive; *static-1000*, as an accurate but not very responsive backward message approximation; *static-4*, as the reciprocal embodiment of a moderately accurate but very responsive approximation; and finally, *static-0*, in which no lookahead is used at all, effectively disabling any future evidence in the computation of the joint posteriors.

In the world of Kalman filtering, the latter approach corresponds to pure real-time filtering, and is often used for the online learning of process parameters with Kalman filters. The other approaches use some amount of lookahead data, and implement some sort of *smoothing*, in the Kalman filtering terminology. By slightly delaying the estimator in order to permit a peek into its future, smoothing is known to enhance the accuracy of state estimates in Kalman filters, and has been used as a heuristic in many applications of Kalman filters, including parameter estimation [WH89, CC91].

To minimize the computational burden, all tests were conducted using the 5+5 structural approximation, which has been previously seen as providing an extremely close approximation to exact EM. In those conditions, The compared running times for the various algorithms are: $0.4s$ per slice for batch EM; $1.4s$ for dynamic-1000; $0.5s$ for static-1000; $0.5s$ for static-4; and $0.3s$ for static-0. We evaluated these temporal approximations in an offline and an online setting.

The purpose of the offline learning experiments was to provide a benchmark in which our various online approximations could be adequately compared to standard EM with full forward-backward. We used the same sampled sequence of 1000 data points as in Section 8.1.3; for the online algorithms, an endless stream of data is simulated by repetitively looping the sequence onto itself. The results are shown in Figure 8.2. We see that the dynamic-1000 algorithm reaches the same quality model

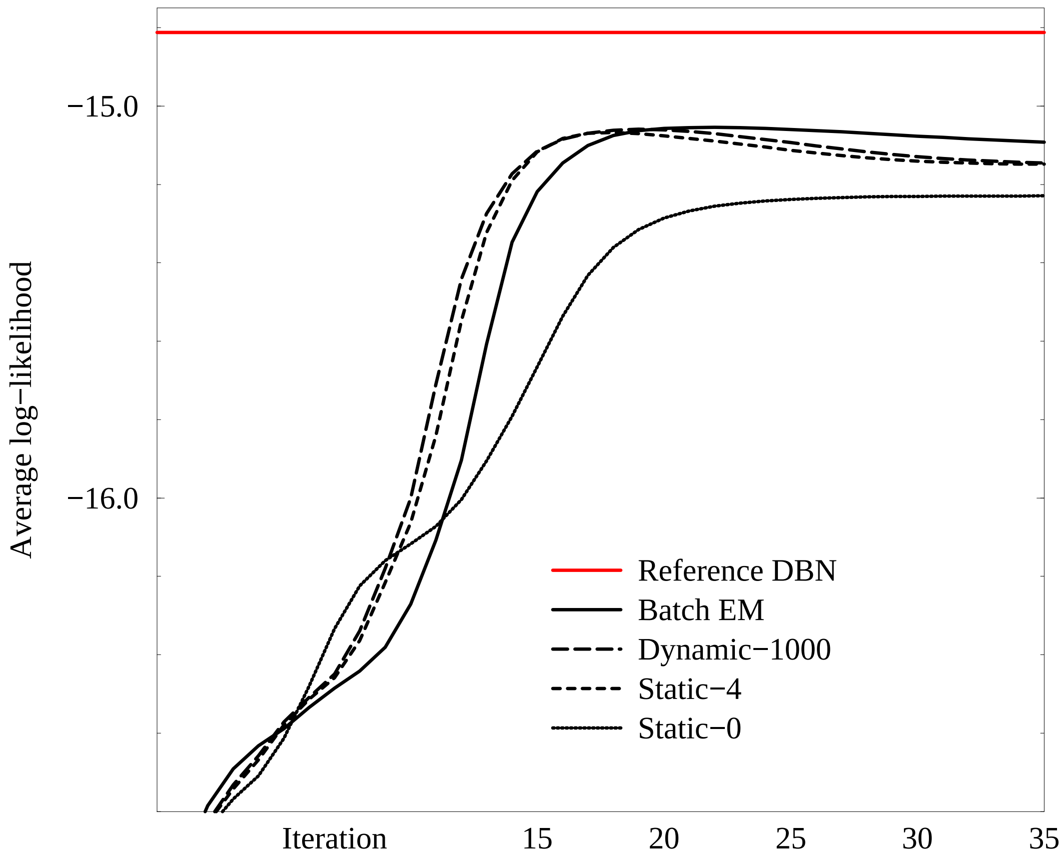


Figure 8.2: Compared quality of various temporal approximations for offline parameter learning.

as standard batch EM, but converges sooner. As observed with incremental EM in atemporal settings [NH98], the accelerated convergence is due to the frequent update of the sufficient statistics based on more accurate parameters.

More interestingly, we see that the static-4 algorithm, which uses a lookahead window of only 4 time slices, also reaches the same accuracy. These results illustrate the adequacy of our strategy of ignoring evidence far in the future, even for a very weak notion of "far". By contrast, we see that the quality reached by the static-0 approach remains significantly lower, presumably because the expected sufficient statistics made available to the M-step are consistently worse, as they ignore all future

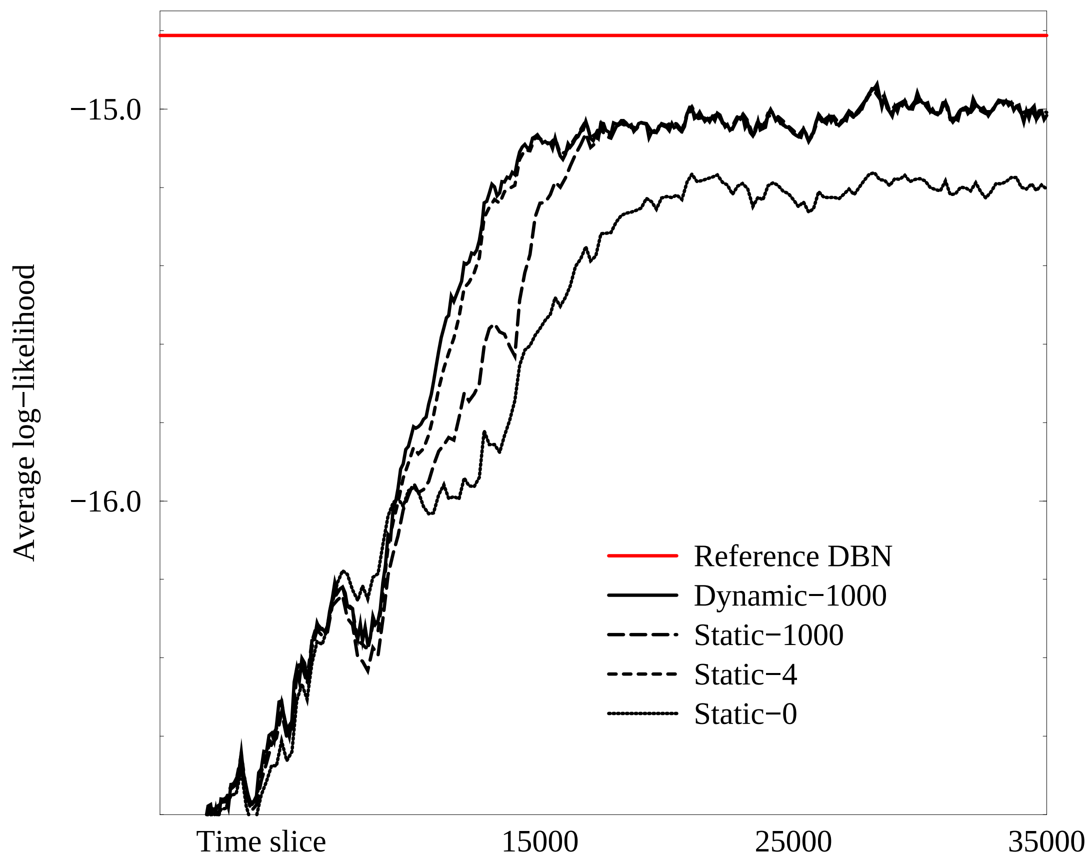


Figure 8.3: Compared quality of various temporal approximations for online parameter learning.

evidence. Thus, for this network structure and data, a lookahead is indispensable, yet even a very short one of 4 time slices performs as well as a full forward-backward update.

Our online learning experiments used a single long sequence of 40000 slices sampled from the reference network, which was more than long enough to reach a plateau in the learning curves of all the considered algorithms. The results are shown in Figure 8.3. Again, we see that the static-4 approach is almost indistinguishable in terms of accuracy from the dynamic-1000 approach. Both converge more rapidly than the static-1000 algorithm, in accordance with our expectations. This illustrates the superiority of frequent updates of short windows over infrequent updates of longer windows. Finally, we note that the static-0 algorithm converges to a hypothesis of significantly lower quality, compared to all the other algorithms. As in the offline setting, the presence of a lookahead window appears to be a necessity, even if a short one will often suffice, in accordance with the theory.

To complement the above observations, we conducted a large number of additional experiments to further support our claims on the temporal approximation, both in an offline data stream setting and in a true online setting. Each set of experiments was conducted on both the BAT and the WATER networks, using 16 and 20 independent data sets respectively, in each case a 1000-point training set, a 100-point validation set, and a 100-point independent test set, all sampled from the reference networks. To save computing time, a fixed approximation clustering was used to speed up inference: the '5+5' clustering from Section 4.3 was used for BAT, while the '2+4+2' clustering was retained for WATER. The following temporal approximation strategies are compared:

- Strategies based on regular (non-incremental) EM:
  - No backward messages, *i.e.*, no lookahead, or 'nolook': only the forward messages are used for learning;
  - Static-1000 backward message pre-caching, or 'stat1000': a 1000-point window of backward messages is statically pre-cached, and updated only when it is completely exhausted, as explained in Section 8.2.3 (in the case of a 1000-point offline training set, this strategy boils down to regular batch EM; this strategy was not tested for online learning);
- Strategies based on incremental EM with exponential decay [NH98]:
  - Learning without lookahead ('nolook');
  - Static-1000 backward message pre-caching, or 'stat1000';
  - Static-100 backward message pre-caching, or 'stat100';
  - Static-4 backward message pre-caching, or 'stat4';

– Dynamic-4 backward message pre-caching, or 'dyna4': a 4-point window of backward messages is pre-cached, and periodically updated using a schedule as explained in Section 8.2.3;

– Dynamic-40 backward message pre-caching, or 'dyna40';

Tables 8.5 and 8.6 gather the results of the offline experiments for the BAT and WATER networks, respectively. The results of the online experiments using the same networks are detailed in Tables 8.7 and 8.8.

The test accuracy of the various learned models is observed to fluctuate somewhat as a function of the random data sets and starting model parameters (*e.g.*, instance #15 in Table 8.8 presents an unusually high loss for all approximation schemes, compared with other instances). In spite of this, it is readily apparent from these tables that all flavors of lookahead enable the learning of high-quality models. Interestingly, the approximations without lookahead appear to perform consistently worse, whether one uses a regular EM update stragegy (as tested in the offline case only), or one based on incremental EM; this seems to confirm our previous observations. Similarly to what we previously observed, the individual results of the various experiments seem to fluctuate much more due to the mere randomization of the data sets than as a result of the approximations.

Table 8.5: Offline learning loss for various approximations on independent test data, for the BAT model using the '5+5' clustering.

	Temporal approximation method (offline case)							
	regular EM		exponential-decay incremental EM					
	nolook	stat1000	nolook	stat1000	stat100	stat4	dyna4	dyna40
BAT network (using the '5+5' clustering)								
#1	0.773	0.758	0.575	0.726	0.543	0.537	0.542	0.544
#2	1.203	1.220	1.229	0.998	1.076	1.087	1.002	1.074
#3	0.479	0.369	0.671	0.406	0.381	0.382	0.380	0.379
#4	0.487	0.128	0.522	0.200	0.175	0.170	0.170	0.176
#5	0.502	0.321	0.480	0.300	0.294	0.308	0.303	0.294
#6	0.523	0.168	0.559	0.183	0.174	0.181	0.174	0.173
#7	0.408	0.250	0.392	0.253	0.241	0.248	0.243	0.245
#8	0.564	0.710	0.534	0.471	0.456	0.460	0.462	0.461
#9	0.814	0.616	0.742	0.965	0.823	1.346	0.763	0.778
#10	0.454	0.340	0.474	0.487	0.427	0.477	0.417	0.425
#11	0.940	0.858	0.884	0.851	0.808	0.804	0.808	0.808
#12	0.404	0.209	0.424	0.240	0.221	0.298	0.211	0.223
#13	0.405	0.979	0.549	0.836	0.796	0.771	0.744	0.804
#14	0.844	0.733	0.840	0.747	0.736	0.758	0.743	0.737
#15	0.764	0.680	0.475	0.314	0.333	0.299	0.303	0.304
#16	0.697	0.593	0.725	0.674	0.681	0.667	0.674	0.687
Summary statistics								
avg.	0.6413	0.5583	0.6297	0.5407	0.5103	0.5496	0.4962	0.5070
dev.	0.2313	0.3175	0.2154	0.2852	0.2776	0.3358	0.2622	0.2760

Table 8.6: Offline learning loss for various approximations on independent test data, for the WATER model using the '2+4+2' clustering.

	Temporal approximation method (offline case)							
	regular EM		exponential-decay incremental EM					
	nolook	stat1000	nolook	stat1000	stat100	stat4	dyna4	dyna40
WATER network (using the '2+4+2' clustering)								
#1	0.391	0.408	0.292	0.347	0.378	0.372	0.372	0.378
#2	0.789	0.907	0.909	0.742	0.711	0.694	0.734	0.712
#3	0.453	0.379	0.360	0.351	0.360	0.337	0.347	0.363
#4	0.431	0.364	0.488	0.409	0.510	0.382	0.354	0.573
#5	0.481	0.412	0.437	0.455	0.434	0.381	0.431	0.481
#6	0.780	0.657	0.790	0.787	0.708	0.719	0.681	0.705
#7	0.634	0.570	0.617	0.490	0.471	0.530	0.467	0.469
#8	0.489	0.304	0.454	0.286	0.299	0.263	0.297	0.300
#9	0.613	0.648	0.733	0.868	0.474	0.681	0.525	0.468
#10	0.392	0.202	0.463	0.205	0.287	0.317	0.260	0.170
#11	0.697	0.801	0.584	0.612	0.682	0.564	0.666	0.679
#12	0.315	0.211	0.307	0.228	0.157	0.117	0.234	0.184
#13	0.581	0.367	0.598	0.336	0.367	0.399	0.373	0.368
#14	0.516	0.473	0.618	0.515	0.509	0.521	0.536	0.509
#15	1.629	1.550	1.489	1.317	1.315	0.959	1.057	1.077
#16	0.637	0.407	0.515	0.401	0.369	0.367	0.372	0.370
#17	0.681	0.595	0.904	0.756	0.751	0.737	0.755	0.754
#18	0.298	0.217	0.271	0.359	0.310	0.282	0.310	0.309
#19	0.553	0.571	0.616	0.566	0.639	0.592	0.655	0.648
#20	0.311	0.299	0.293	0.285	0.237	0.268	0.237	0.237
Summary statistics								
avg.	0.5836	0.5171	0.5869	0.5158	0.4984	0.4741	0.4832	0.4877
dev.	0.2874	0.3100	0.2863	0.2702	0.2566	0.2080	0.2158	0.2254

Table 8.7: Online learning loss for various approximations on independent test data, for the BAT model using the '5+5' clustering.

	Temporal approximation method (online case)						
	regular EM	exponential-decay incremental EM					
	nolookahead	nolook	stat1000	stat100	stat4	dyna4	dyna40
BAT network (using the '5+5' clustering)							
#1	1.167	0.793	0.854	0.831	0.832	0.829	0.830
#2	1.344	1.165	1.037	1.112	1.122	1.112	1.112
#3	0.467	0.456	0.347	0.343	0.344	0.342	0.352
#4	0.467	0.471	0.105	0.088	0.089	0.087	0.088
#5	0.368	0.342	0.258	0.248	0.249	0.248	0.248
#6	0.564	0.368	0.061	0.065	0.068	0.067	0.065
#7	0.377	0.389	0.165	0.157	0.177	0.170	0.158
#8	0.697	0.389	0.282	0.335	0.328	0.336	0.333
#9	0.403	0.337	0.244	0.242	0.236	0.242	0.188
#10	0.414	0.347	0.217	0.210	0.214	0.210	0.210
#11	0.853	0.811	0.352	0.332	0.296	0.345	0.333
#12	0.281	0.199	0.037	0.029	0.088	0.029	0.029
#13	0.100	0.388	0.710	0.709	0.632	0.633	0.709
#14	0.968	1.034	0.706	0.708	0.762	0.707	0.708
#15	0.290	0.271	0.164	0.164	0.165	0.164	0.164
#16	0.576	0.572	0.475	0.473	0.470	0.473	0.473
Summary statistics							
avg.	0.5835	0.5208	0.3759	0.3779	0.3759	0.3746	0.3750
dev.	0.3407	0.2809	0.2988	0.3097	0.3059	0.3038	0.3114

Table 8.8: Online learning loss for various approximations on independent test data, for the WATER model using the '2+4+2' clustering.

	Temporal approximation method (online case)						
	regular EM	exponential-decay incremental EM					
	nolookahead	nolook	stat1000	stat100	stat4	dyna4	dyna40
WATER network (using the '2+4+2' clustering)							
#1	0.238	0.227	0.184	0.126	0.137	0.125	0.126
#2	0.573	0.521	0.567	0.469	0.467	0.458	0.467
#3	0.074	0.035	0.069	0.066	0.063	0.061	0.065
#4	0.503	0.415	0.208	0.224	0.160	0.223	0.219
#5	0.195	0.238	0.223	0.209	0.119	0.205	0.205
#6	0.791	0.785	0.600	0.601	0.617	0.598	0.601
#7	0.449	0.416	0.386	0.395	0.356	0.375	0.396
#8	0.097	0.091	0.096	0.096	0.115	0.093	0.096
#9	0.585	0.522	1.004	0.990	0.628	0.985	0.993
#10	0.266	0.205	0.092	0.076	0.059	0.070	0.077
#11	0.282	0.345	0.111	0.294	0.102	0.104	0.265
#12	0.318	0.241	0.042	0.068	0.050	0.038	0.069
#13	0.444	0.390	0.356	0.352	0.157	0.356	0.353
#14	0.520	0.497	0.449	0.386	0.396	0.395	0.389
#15	2.007	1.926	1.438	1.853	1.659	1.850	1.853
#16	0.441	0.429	0.294	0.269	0.283	0.277	0.270
#17	0.559	0.600	0.512	0.491	0.409	0.491	0.491
#18	0.086	0.078	0.158	0.162	0.113	0.164	0.162
#19	0.162	0.123	0.137	0.194	0.123	0.195	0.193
#20	0.302	0.278	0.104	0.155	0.113	0.138	0.153
Summary statistics							
avg.	0.4446	0.4181	0.3515	0.3738	0.3063	0.3601	0.3722
dev.	0.4159	0.4037	0.3502	0.4136	0.3669	0.4186	0.4144

Chapter 9

Toward structure discovery

The usefulness of parameter learning from data given a fixed model structure is directly dependent on the availability of an accurate qualitative description of the interactions taking place among the system variables. Many situations of practical interest do not strike such a middle ground: sometimes a complete quantitative model of the system is available from the onset, in which case no learning is necessary; at other times very little reliable prior knowledge is assumed, and both the model structure and parameters must be learned from the observations alone.

In this chapter, we discuss some of the problems encountered in the task of learning the structure of an unknown process. We suggest a possible approach, supported by a few preliminary experiments.

9.1 Approximate expected sufficient statistics

The approximate message propagation algorithm provides an efficient way of inferring statistics from partial data in large systems. As shown in the previous chapter, this algorithm offers an appropriate solution for parametric learning of Markovian processes of given structure, where the set of needed statistics is known and invariable, and their complexity is directly commensurate with that of the model template. The situation becomes more delicate when the model structure is not known and has to be learned together with the model parameters. As outlined in Section 7.3, this situation often calls for a local structure search, which cannot be executed without an expanded set of statistics. Contrarily to the parametric learning case, where the statistics are directly patterned on a fixed model structure, here the expanded statistics merely reflect speculations on what portion of the space of structures should be available for exploration during the upcoming search step. Not only are such expanded statistics not as straightforward to compute given the current model structure, their collection from partially observed data would require expensive inference operations on a potentially very large number of different candidate structures.

9.1.1 Methods and limitations

Recall from Chapter 7 that ESS are derived from the joint marginals over sets of state variables, as in Equation 7.1. Given a fixed structure $\mathbf{B}_{\rightarrow}$, computing ESS is easy: the only required statistics are the ones over the families of the individual variables in $\mathbf{B}_{\rightarrow}$, and the approximate forward-backward message propagation of Section 8.1.1 can be readily used for their estimation. Based on Algorithm 4.5, the method of Section 8.1.1 uses a junction tree Υ, derived from the transition model, to propagate messages in either direction, as well as to combine forward and backward messages into posterior distributions, as required by Equation 7.2. In order to learn the parameters of a fixed structure $\mathbf{B}_{\rightarrow}$, those posteriors, and the ESS which derive from them, must be detailed enough to cover each variable family in the transition model; *i.e.*, for each ulterior variable $X^{\triangleright}$, a set of statistics $N[X^{\triangleright}|Pa[X^{\triangleright}]]$ should be available. For a fixed structure, the junction tree is guaranteed to have this property: the construction outlined in Algorithm 4.5 ensures that each family is contained in full in some clique of Υ. Furthermore, not only can Υ be directly used to combine a forward message $\tilde{\boldsymbol{\alpha}}^{(t-1)}$ and a backward message $\tilde{\boldsymbol{\beta}}^{(t)}$ to produce the full approximated joint posterior $\tilde{\boldsymbol{\mu}}^{(t-1,t)}$, as above, but $\tilde{\boldsymbol{\mu}}^{(t-1,t)}$ directly admits a compact representation as the calibrated junction tree $\Upsilon^{(t)}$ itself (where $\Upsilon^{(t)}$ instantiates Υ to cover the transition from $t-1$ and t). It is then easy to extract the needed marginals from the appropriate cliques and integrate them over the entire sequence to obtain the required ESS, as detailed earlier.

For structure learning, this easy solution is no longer applicable, since the current structure $\mathbf{B}_{\rightarrow}$ is used to compute statistics for a variety of other candidate structures: in general, a tree Υ constructed from $\mathbf{B}_{\rightarrow}$ will not spontaneously cover the families of the other candidate structures. Forcing such coverage for all the families involved is not an option either, as this will almost invariably result in the degenerate tree where all the variables belong to a single, intractably large clique. Thus, the main challenge amounts to estimating a large set of complex statistics, without resorting to doing full inference for each of them, either separately (using many small tailored junction trees), or all at once (using a huge degenerate junction tree).[1]

Let $\mathbf{Y}$ be a set of variables for which joint statistics are desired; *e.g.*, $\mathbf{Y}^{(t)}$ could be the family of $Y^{(t)}$ in a potential candidate structure. The problem is to compute ESS over $\mathbf{Y}$ when $\mathbf{Y}$ is not contained in full in any clique of Υ. A naive approach to this problem is to compute the necessary posterior $\boldsymbol{P}_{\mathbf{B}_{\rightarrow}}[\mathbf{Y}^{(t)}|\mathbf{D}]$ by performing a special-purpose inference step over $\Upsilon^{(t)}$, tailored to $\mathbf{Y}$. Unfortunately, this operation can be very expensive as $\mathbf{Y}$ contains variables which are "distant" in the current structure. It also needs to be performed a great many times every time slice: once for each family of statistics of interest. Finally, this approach is almost invariably intractable for seeking statistics of higher Markovian order, *i.e.*, where $\mathbf{Y}^{(t)}$ spans the time slices from $t-d$ to t, with $d>1$. Although they are not conventionally used in structure learning, the availability of such statistics will soon prove to be the key to discovering hidden variables, see Section 9.2.

9.1.2 Scalable approximation

Since the determination of the required statistics is intractable in most cases of interest, we propose an approximate solution, in the same spirit as the decomposed message propagation introduced in the previous chapters, and related to the variational approximation of [GJ96]. Instead of computing the joint distribution $\boldsymbol{P}[\mathbf{Y}^{(t)}]$,

[1]It is noted that the difficulty of collecting joint statistics over "distant" variables does not arise in the work of Friedman *et al.* [FMR98], as they essentially rely on a degenerate junction tree to provide access to all possible ESS in models that are small enough to allow it. It is our requirement to provide a more flexible representation and finer grained decomposition of the junction tree, for the sake of scalability, that brings up the issue.

we approximate it as a product of independent marginals over the individual variables (or sets thereof) of $\mathbf{Y}$.

Let Y_i be an instantaneous variable in $\mathbf{Y}$. Since $\mathbf{Y}^{(t)}$ may span several time slices from $t-d$ to t, the notation $Y_i^{(t_i)}$ is used to indicate that the actual time slice at which Y_i is evaluated in $\mathbf{Y}^{(t)}$ is $t_i \in \{t-d, ..., t\}$. Then, our approximation scheme assesses $\boldsymbol{P}_{\mathbf{B}_\rightarrow}[\mathbf{Y}^{(t)}|\mathbf{D}]$ as:

$$\boldsymbol{P}_{\mathbf{B}_\rightarrow}[\mathbf{Y}^{(t)}|\mathbf{D}] \;\approx\; \prod_i \boldsymbol{P}_{\mathbf{B}_\rightarrow}[Y_i^{(t_i)}|\mathbf{D}] \;,$$

where, for each Y_i, the posterior $\boldsymbol{P}_{\mathbf{B}_\rightarrow}[Y_i^{(t_i)}|\mathbf{D}]$ is computed by marginalizing some clique of $\Upsilon^{(t_i)}$ in which $Y_i^{(t_i)}$ is present. The various junction trees $\Upsilon^{(t_i)}$ used for this operation result from the approximate forward-backward propagation, at almost no additional cost. From there, the approximate ESS $\tilde{N}_{\mathbf{Y}}$ are computed as in Equation 7.1, by accumulating the individual posteriors over t.

The whole process requires a pass over the junction tree to perform the marginalization, and then a simple aggregation of the marginals into accumulators, which requires linear time in the number of sufficient statistics that we are maintaining.[2]

As implied by the above description, this approach readily applies to the task of computing ESS for events $\mathbf{Y}$ that span several time slices: all that is needed is to extract the marginals over the various $Y_i^{(t_i)}$ from more than one junction tree, as needed. For example, joint ESS over an event $\mathbf{Y} = \langle X_{i_1}^{(t)}, X_{i_2}^{(t-2)} \rangle$ could be approximated by extracting $\boldsymbol{P}[X_{i_1}^{(t)}]$ from $\Upsilon^{(t)}$ and $\boldsymbol{P}[X_{t-2}^{(t-2)}]$ from $\Upsilon^{(t-2)}$, and multiplying the two marginals. The ESS for the event are obtained by repeating this operation for each t, and adding up the results.

At first glance, one might think that this approximation discards all correlations between the various variables or variable clusters of $\mathbf{Y}$. In general, however, this is not

[2]We note that we could have used a more refined computation that would have taken advantage of the co-occurrence of some subsets of $\mathbf{Y}$ within a single clique in order to avoid approximating them as independent. However, this extension would require that we marginalize the cliques in a potentially different way for every statistic that we need to compute. Our experiments (see below) suggest that the error introduced by this approximation is probably not large enough to be worth the significant computational overhead.

Table 9.1: Comparison of parametric EM based on exact and approximate ESS: negative log-likelihood on test data for parametric EM, for different starting points (results expressed in bits/time slice).

BAT network	Seed #1	Seed #2	Seed #3
Gold standard (reference model)	22.1860	22.1860	22.1860
Model learned without ESS approximation	22.4026	22.2801	22.3269
Model learned with ESS approximation	22.2633	22.2676	22.2782

the case, since the ESS for $\mathbf{Y}$ are constructed by accumulating posterior distributions of $\mathbf{Y}^{(t)}$ given different configurations of evidence. Consider, for example, the situation where the event of interest $\mathbf{Y}$ is composed of two binary variables $A, B \in \mathbf{Y}$, which belong to different cliques. If, over the course of time, it appears from the data that A and B are either both probably true or both probably false, depending on the evidence, then at each step the product distribution $\mathbb{P}[A^{(t)}] \otimes \mathbb{P}[B^{(t)}]$ will display a larger probability mass at either $\langle 0, 0\rangle$ or $\langle 1, 1\rangle$, revealing the correlation between A and B in the ESS of $\mathbf{Y}$. In other words, this approximation is able to learn the Exclusive-OR function. Of course, there are cases where this approximation would lose correlations. For example, if $A^{(t)}$ and $B^{(t)}$ are both marginally uniformly distributed, yet are correlated via higher order moments, the approximate sufficient statistics would not reveal such a correlation.

In general, if the evidence does not exhibit any information about skews in the marginal distribution of a variable, but only about the correlations between the different variables, our approximate ESS will fail to reveal the correlation. In response, we argue that such models are hard to learn in general, with or without our approximation. Indeed, if the evidence does not provide information about the value of a hidden variable, the prospects of learning something meaningful about its distribution are very limited.

9.1.3 Preliminary results

Even though the ultimate goal of ESS approximation is to support structural learning in large systems, we first tested the approximation on the better understood problem of parametric estimation. The validation test involved learning the model parameters given the correct model structure, from synthetic data sampled from the reference model. We used the BAT network, as first described in Section 4.3. The learning set was composed of a single long sequence of 20000 synthetic data points. We then attempted to estimate the parameters back from the data, given the correct structure, and measured the fitness of the resulting model on an independent synthetic test sequence, as before. We ran two versions of this experiment, the first one using the approximate forward-backward message-passing algorithm previously described but without the ESS approximation, the other using both approximations.

As can be seen in Table 9.1, the approximation does not degrade the learning accuracy. On the contrary, the approximation even seems to be slightly beneficial, which could speculatively be explained as a regularization effect, caused by the reduced effective dimensionality of the approximated ESS. The ensuing smoothing effect helped produce better models by reducing numerical overfitting.

9.2 Structure discovery

In the previous section, we suggested a method for efficiently computing expected sufficient statistics, to be used, for instance, during the E-step of SEM. Since SEM search in the M-step requires ESS for each family that may potentially be changed, this raises the question of what strategy should be used to decide on which statistics to compute.

Moreover, SEM and other local search methods are inapplicable unless some basic knowledge of the system is at hand; at the very least, knowledge of all hidden state variables is a requirement. Since we often face situations where even that much is unknown before techniques like SEM can be applied, this begs the question of how hidden state variables can be uncovered in the first place.

9.2.1 Efficient structure search

Although the SEM algorithm is unable to discover the existence of unknown variables, it is rather capable at fitting dependency structures over specified sets of (observed and hidden variables). Since our method uses SEM as a subroutine, we now focus on the problem of supplying SEM with an adequate collection of sufficient statistics.

A first approach is to compute in advance all the expected sufficient statistics the search might need. However, since the number of statistics grows exponentially with the size of the network, this solution is impractical. When the indegree of each variable is restricted, the number of statistics is polynomial, but still unrealistically large.

A second approach, which was used by Friedman *et al.* [FMR98], is to lazily compute sufficient statistics on demand, so that the statistics for a family $\mathbf{Y}$ are computed only when the search needs to evaluate a structure with this family. To avoid unnecessary computations, computed statistics are stored in a cache, relegating the actual ESS computation for statistics that have not yet been required during this particular search phase. Unfortunately, this approach does not scale appropriately for long data sequences, since each ESS computation requires a traversal over the entire sequence, using the forward-backward algorithm. This could be partially avoided by storing all the junction trees $\Upsilon^{(t)}$ computed during the forward-backward propagation, or, at least, the posterior of each variable at each point in time, but this would only place the burden on the storage needed for all that information.

These two solutions are at the extreme ends of a spectrum. Friedman *et al.* [FPN99] present an intermediate search method, which is more appropriate for our purpose. The search procedure works in stages. At the beginning of each stage the search procedure posts the statistics it will require for that stage. These are selected in an informed way, based on the current state of the search; for example, the search method may choose to include all families that result from at most a fixed number of local structure modifications. The requested statistics are then computed in one batch in the E-step, using a single inference phase for all of them at once.

More specifically, at the beginning of the E-step, the algorithm finds for each variable Y_i a set $\hat{P}\mathrm{a}[Y_i]$ of *candidate parents*, based on the current network structure.

During the M-step, the search is restricted to consider only operations that involve adding edges $X \to Y_i$ for $X \in \hat{P}\mathrm{a}[Y_i]$, or removing current arcs. The number of ESS required for these operations is fairly small, and can be collected and accumulated at once, in a single execution of forward-backward. The algorithm uses heuristics to focus the attention of the search procedure on plausible candidate parents, as described by Friedman *et al.* [FPN99]; the algorithm therefore requires relatively few statistics in each stage. The restricted search phase is deemed complete when no further progress can be made using the available statistics. This can happen either because further progress would require additional statistics, or because the modified structure is deemed optimal according to the last set of statistics. In either case, the M-step terminates, and a new EM iteration is initiated, based on the new structure. The process is iterated until convergence of the scoring function.

9.2.2 Discovering hidden variables

Beyond learning structural dependencies between known random variables, a fundamental problem remains when learning dynamic systems from real data: the discovery of hidden variables. In financial domains, for example, the stock price of companies in the same industry are typically correlated. This correlation is most often caused by hidden variables, such as news bulletins concerning the industry and the subjective perception thereof by investors. Those variables are hidden, and sometimes even difficult to quantify, but nonetheless have a real impact on the observable variables. Unless the learner realizes that observed correlations are caused by hidden variables, the induced model is likely to poorly reflect the reality.

The task of discovering new hidden variables is notoriously difficult. In temporal sequences, however, there are some cues that can indicate the presence of such variables. In particular, if a hidden variable is missing from a learned model, this often will be revealed as a non-Markovian correlation in the data with respect to the model, *i.e.*, a correlation that ties variables at times $t-\delta_1$ and $t+\delta_2$ given all variables in the model at time t. The non-Markovian correlations are induced by the loss of information as the hidden variable is forgotten from step to step; in other words, the model is unable to statistically explain the observed values of the variables, because it does not track the hidden dependence that causes those values. This phenomenon can be used to identify missing hidden variables in the current model. Practically, the learner can search for observed non-Markovian correlations, and use them as an indication that the model needs additional memory about the past to explain the present. The goal of this operation is to determine the existence of hidden variables, and to seed their initial location.

More concretely, suppose that we discover that we can predict $X^{(t)}$ using $X^{(t-1)}$ and $Y^{(t-2)}$, for example. Then we might consider creating a new hidden variable H such that $H^{\triangleleft}$ is a parent of $X^{\triangleright}$ and $Y^{\triangleleft}$ is a parent of $H^{\triangleright}$. Thus, we will have that $Y^{(t-2)}$ influences $H^{(t-1)}$ via the $Y^{\triangleleft} \to H^{\triangleright}$ edge, and that $H^{(t-1)}$ in turn influences $X^{(t)}$ via the $H^{\triangleleft} \to X^{\triangleright}$ edge. In other words, the variable H behaves as the memory of Y with a one-time-slice delay.

In general, we propose the following algorithm. We start by learning the edges among variables in k time slices, arranged as a k-TBN, for some fixed time window k. When some of the variables are unobserved, we use structural EM and our approximation methods to estimate the ESS of the variables in these k consecutive time slices. The sufficient statistics are computed once, and then used for an extended search phase over structures. This strategy, combined with our approach to computing ESS for variables that are far apart in the network, allows us to estimate the ESS for the k-TBN without ever doing inference on such a large structure.

After having learned such a network, we eliminate the non-Markovian arcs by creating new hidden variables to carry forward the value of those variables that participate as parents in non-Markovian correlations. Any variable X in time slice $t-d$ which directly influences a variable Y at time t requires $d-1$ new hidden variables, denoted $X^{-1}, ..., X^{-d+1}$: at time $t-1$, the i-th introduced variable X^{-i} has the same value that X had at time slice $t-i$. This operation transforms the learned k-TBN into a more conventional Markovian 2-TBN.

Before optimizing the parameters of the new network structure, a few more modifications are in order. We endow each newly created hidden variable with *persistence arcs*, or arcs from the anterior to the ulterior instantiation of the same variable; this allows the hidden variables to depend on longer term past. We also initialize the parameters of the newly introduced hidden variables to be noisy CPDs biased toward a combination of acquiring the value of the variables they close, and retaining their previous value through the persistence arc. The intent is to encourage the structural search to construct variables that remember global phenomena.

Having transformed the learned k-TBN into a 2-TBN using the above heuristics, the iteration is completed by running parametric EM in order to optimize the parameters. If convergence has not been reached, the process resumes with the three steps of the next iteration: structure learning, introduction of hidden variables, and parametric optimization.

9.2.3 Preliminary results

We tried our hidden variable discovery algorithm on four different domains: three of them involve real-world data, while the fourth is based on synthetic data sampled from a reference DBN. The experimental procedure involved running our structure learning and variable discovery algorithm for several iterations, assessing the fitness of the learned network at key milestones along the way. The first iteration starts with the collection of approximate ESS in the E-step, followed by a structure search phase in the M-step, allowing non-Markovian edges as described above. The result is a non-Markovian learned network involving only the initially known variables; its fitness is measured on test data. Next in order are the transformation of non-Markovian edges into hidden variables and the subsequent parameter tuning, which complete the first iteration; the fitness of the network on test data is measured at this point. The subsequent iterations are similar, except that all currently known variables are used in the inference and search phases, and the network fitness on test data is measured at the end of each iteration only. As previously described, the inference phase involves the collection of a reasonably large number of sets of approximate ESS, and is followed by a matching extensive search phase. For structure search, we experimented with both the Bayesian (BDe) score and the BIC score, and with both flat and tree-structured conditional probability distributions for the nodes of the learned model. We report the results for the BDe score with trees; the results for the other cases are somewhat different, but exhibit the same general trends. Since a single training set was used, the results of those experiments are to be regarded as merely suggestive.

As points of comparison, we also learned two other types of structure: standard Markovian DBN structures over the observable variables, as learned by a standard BN structure learning algorithm, and *Factorial HMM (FHMM)* structures with various numbers of hidden variables [GJ96], as tuned by parametric EM, using our approximate message passing algorithm of Section 8.1.1 to keep the computational effort to reasonable levels. We recall from Section 3.6 that an FHMM is best viewed as a DBN with some number h of hidden variables, each of which evolves independently of the others; each observable variable depends on all of the hidden variables within its time slice. FHMMs have been shown to be a good candidate for modeling several interacting processes evolving in parallel, such as the multiple articulatory mechanisms involved in the formation of speech. We attempted to include FHMMs with 2, 4, and 6 binary hidden variables in our experiments, except in the case of data sets with only a small number of observables, in which we tried FHMMs with fewer hidden variables.

The various data sets and experiments conducted on them are detailed in the following sections; all the results are summarized in Table 9.2.

Bach Chorales data set

The Bach Chorales data set was proposed as part of the 1991 Santa Fe competition for learning time series [WG90]; it encodes the melodic line of 100 chorales attributed to J.S. Bach. The model has five discrete attributes, labeled: *Key signature*, *Pitch*, *Duration*, *Fermata*, and *Time signature*; the first two attributes are melodic, while the last three capture temporal aspects of the piece. On average, each chorale is about 50 notes long. For our experiments, we partitioned the set into a training set and a test set. The training set consisted of 71 randomly chosen chorales, amounting to a total of 3212 training transitions; the test set consisted of the remaining 29 chorales, for a total of 1379 test transitions. The experiments consisted of learning a transition

Table 9.2: Fitness of learned models on test data for various learning algorithms and application domains (expressed as the negative log-likelihood in bits/time slice).

		Bach	Apnea	Stock	BAT
Gold standard		n/a	n/a	n/a	22.147
Parametric EM		n/a	n/a	n/a	22.873
FHMM	2 hid vars	8.486	3.635	24.268	23.957
	4 hid vars	5.623	—	24.302	23.562
	6 hid vars	—	—	23.213	23.773
	fully observable only	4.538	1.892	20.834	22.693
SEM	1st iteration	4.503	1.704	20.759	22.418
	2nd iteration	4.513	1.713	20.819	22.434
	3rd iteration	4.537	1.710	20.710	22.388

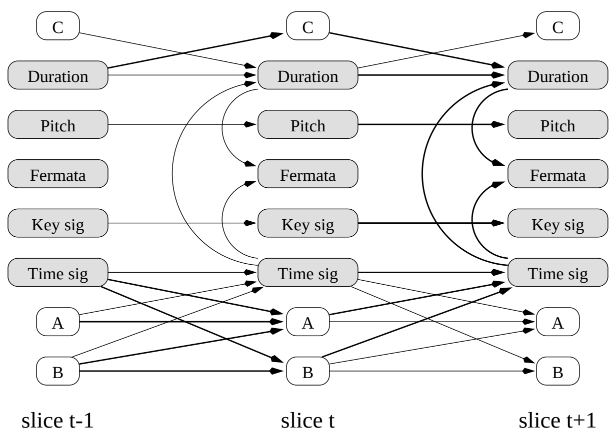


Figure 9.1: Learned 2-TBN model for the Bach Chorales data set. Observable variables are denoted by their names and represented by shaded nodes; hidden variables are labeled alphabetically. The 2-TBN model is unrolled for a few time slices to highlight long-range dependences, as represented by the thicker arcs.

model from the training chorales using the various contending methods listed above, and measuring their accuracy on the test set.

The first column of Table 9.2 shows the results for this data set. It can be seen that all instances of the DBN learning algorithm perform significantly better than any instance of the FHMM algorithm. In addition, the introduction of non-Markovian edges and hidden variables helps significantly with respect to standard structure search over the observables. The final network with hidden variables is shown on Figure 9.1. Somewhat disappointingly, the introduction of hidden variables *per se* does not improve the log-likelihood over the initial structure search for this data set, even though the learned network with hidden variables seems reasonable; this could be attributed to numerical overfitting.

Inspection of the learned structure of Figure 9.1 reveals that our algorithm detected the correlations between the three tempo attributes, but chose to keep the two melodic attributes decoupled from them. All variables have persistence arcs, except the *Fermata* variable, which represents a momentary event corresponding to the end of a segment. The algorithm introduced several hidden variables that capture the non-Markovian nature of the observable variables. Most interestingly, one of the hidden variables models a short non-Markovian dependence of two time slices on the *Duration* variable; on the other hand, the hidden variables introduced for the *Time signature* variable model longer-range dependences, which is quite natural since *Time signature* typically represents a longer-term aspect of a musical piece.

Sleep Apnea data set

The Sleep Apnea data set was also proposed as part of the Santa Fe competition [WG90]. It was obtained by monitoring the evolution of three medical parameters on a patient suffering from sleep apnea. The data set contains 34000 points, collected in a single run of several hours. The sequence is highly non-stationary, due to the various sleep phases and the condition of the patient. Following the suggestion of Dagum and Galper [DG93], each continuously-valued variable was discretized into seven buckets. For the experiments, the sequence was partitioned into four training subsequences, for a total of 19994 transitions, and one test sequence comprising the remaining 13999 transitions.

The relative measured fitness of the various learned models bears strong similarities to the previous data set, with our algorithm performing significantly better

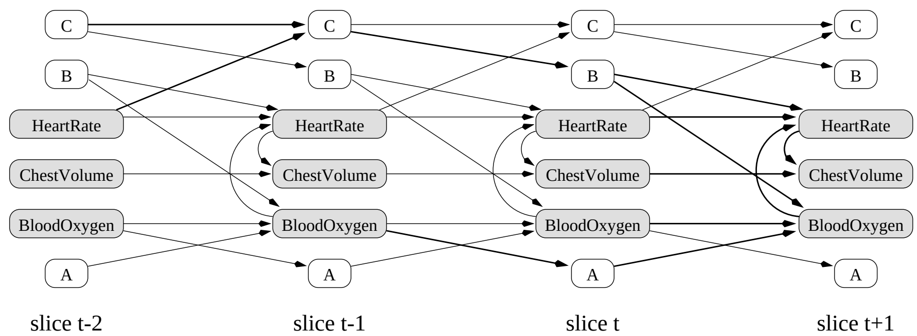


Figure 9.2: Learned 2-TBN model for the Sleep Apnea data set. Observable variables are denoted by their names and represented by shaded nodes; hidden variables are labeled alphabetically. The 2-TBN model is unrolled for a few time slices to highlight long-range dependences, as represented by the thicker arcs.

than all instances of FHMMs. Notably, for this data set, the introduction of hidden variables improves the measured fitness. The learned model is shown on Figure 9.2. Arguably, the network offers intuitive clues as to the significance of the data. For instance, the structure reveals a strong correlation between *BloodOxygen* and *HeartRate*, as well as between *HeartRate* and *ChestVolume*, but not directly between *BloodOxygen* and *ChestVolume*, as would be expected. As in the previous case, the algorithm discovered a number of non-Markovian correlations.

Stock Market data set

The Stock Market data set is a construction of ours, and was assembled from the daily closing prices of securities of 20 companies in a handful of industries: computer hardware, computer software, semiconductors, internet, U.S. car manufacturers, and Japanese car manufacturers. Since trends are typically more revealing of correlations than absolute prices, only the daily trend of each stock is recorded, as being "up" or "down" from the previous trading day, or "n/a" if the stock was not traded for some reason. The period covered extends from February 1992 to February 1999, for a total of 1768 trading days. It is noted that several of the tracked companies were not publicly traded during the earlier part of the period. For the experiments, we

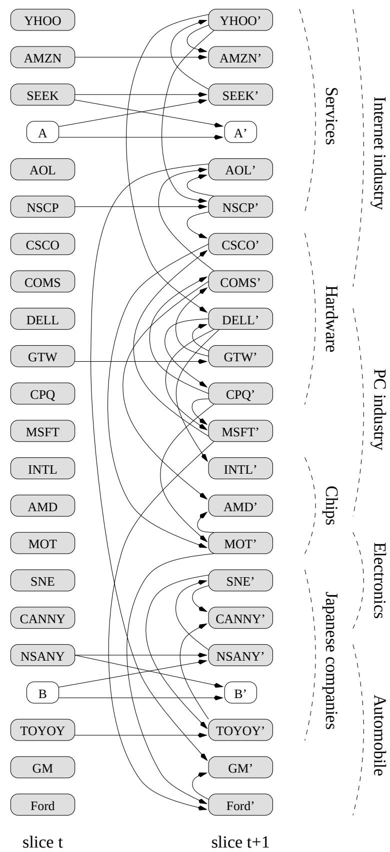


Figure 9.3: Learned 2-TBN model for the Stock Market data set. Observable variables are denoted by their names and represented by shaded nodes; hidden variables are labeled alphabetically.

partitioned the data into a training and a test subsequence, as usual, giving 1195 training transitions and 567 test transitions.

As before, the log-likelihood results are summarized in Table 9.2. For this data set, the introduction of the hidden variables shows a small improvement over the basic structural learning. The shape of the learned network, shown in Figure 9.3, is somewhat different from the case of the previous two domains, mainly due to the fact that most of the correlations appear within a single time slice. This phenomenon is quite natural, as the trading climate can differ significantly from one trading day to the next, although related stocks tend to move in concert within a given day.

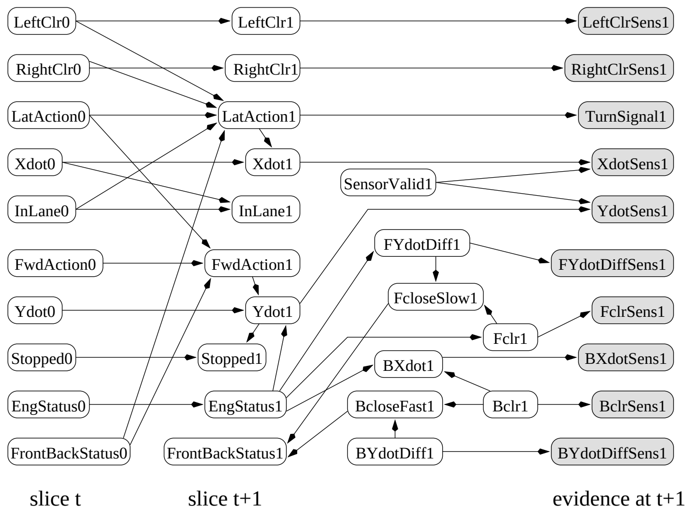


Figure 9.4: Reference BAT network used for comparison and synthetic data generation.

Accordingly, as our algorithm is geared to detecting correlations that induce temporal dependencies, it did not introduce many hidden variables. However, the correlations discovered by the algorithm appear to make sense, in that the induced arcs tend to accumulate between companies in the same industry, or between companies sharing similar characteristics.[3]

BAT synthetic data set

The BAT data set is a synthetic data set generated from the BAT network already encountered in previous chapters, which was originally devised for tracking the motion of a car on a freeway [FHKR95]. The synthetic data set was generated by sampling a single long trajectory from the reference network with its original parameters. The network is reproduced in Figure 9.4; it has 10 state variables and 10 observable variables; the former are expunged from the final sampled sequence, so that only the

[3]The behavior of our algorithm on such data begs the question of introducing hidden variables, not as a model of the memory of a process, but as a summary of a large group of similar correlations. Discovering hidden variables in this setting would require a complementary approach, such as one based on the common intuition of looking for "almost cliques" in the learned network [SGS93]. See [ELFK00] for recent work in this direction.

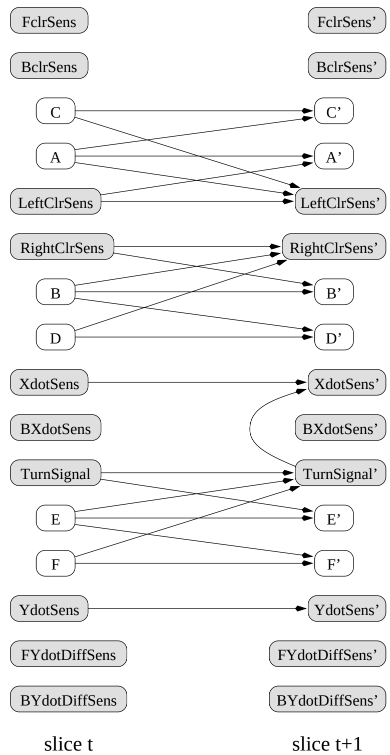


Figure 9.5: Learned 2-TBN model for the synthetic BAT data set. Observable variables are denoted by their names and represented by shaded nodes; hidden variables are labeled alphabetically.

latter are known to the learner. For the experiments, the sampled trajectory was segmented into four training and five test subsequences, for a total of 4992 training transitions, and 5029 test transitions. Only the observable variables were recorded from the trajectory, and the algorithm received no prior knowledge whatsoever about the correct structure.

For this domain, since the data are synthetic and a reference model is available, additional experiments are possible. One of them is to compare the fitness of the learned models to that of the reference network, giving us a "gold standard". Another experiment involves learning the parameters of the model only, given the correct structure of the reference network. All the results are reported in Table 9.2, as usual.

The performance on test data of the model learned by our algorithm is fairly close to the gold standard; it is also much better than parametric EM applied to the correct structure, probably because the large number of parameters in the correct structure cause numerical overfitting. The performance is also significantly better than the FHMM results, or the structure learning restricted to the observable variables. Thus, for this domain, the introduction of hidden variables actually helps to a nontrivial extent. However, the learned structures are rather complex and not very compelling, particularly when compared to the true network. For clarity, we have chosen to represent in Figure 9.5 the somewhat simpler structure learned after only a single iteration of SEM, due to the fairly high complexity of the later structures. We can see that the algorithm does discover a few interesting correlations, such as one between *TurnSignal* and sensed lateral movement *XdotSens*, as one would expect looking at the original model.

Part IV

Epilogue

Chapter 10

Related work

A number of interesting approaches to learning and reasoning about probabilistic systems, and stochastic processes in particular, can be found in the literature. Among those, we now briefly introduce the ones most germane to the focus of this dissertation, and discuss how they relate to our own contribution.

10.1 Inference

10.1.1 Variational inference

In recent years, the variational approach for Bayesian inference has quickly emerged as one of the most prevalent approximation techniques for inference in large Bayesian networks [JGJS99]. It has been applied in a large number of contexts, including that of dynamic Bayesian networks for learning purposes [Gha98].

The basic idea behind variational inference is to successively simplify the network at hand by removing dependences between variables, up to a point where the inference task at hand becomes manageable [SJ95]. The key to this approach is that dependences are removed, their average effects on the network are captured as a set of variational parameters. After the (regular) inference step has been carried out on the simplified network, we are left with a result parameterized by the variational parameters. The last step consists in optimizing the inference result for some quality criterion, eliminating the variational parameters as one would Lagrange coefficients.

This technique was originally inspired by the statistical physics method of *mean field* approximation, whereby a large number of concurrent interactions are approximated as a single mean interaction, in an application of the central limit theorem [Dur95, chap. 2].

One of the consequences of the connection to the central limit theorem is that the variational approximation tends to work best in circumstances where many small influences are simultaneously exerted, as opposed to a few strong ones. In this case, the central limit theorem suggests that the many influences will average out nicely, enabling their removal and approximation as a variational parameter. Thus, the whole class of variational methods for approximate inference shines best in heavily connected graphs, *i.e.*, graph with little structure in the sense used throughout this thesis. This is another difference with the inference approach we propose in Chapter 4, whose operation benefits from, and, indeed, relies on the structural features of the model.

It also turns out, from the mathematics of the variational optimization, that the criterion being minimized when eliminating the Lagrange coefficients, in essentially all flavors of variational inference, is the "reversed" KL divergence between the true distribution of interest P and its variational approximation Q (*i.e.*, to be clear, the KL divergence between the two, where Q occupies the place of the true distribution in the expression of the KL divergence):

$$Q^* \quad = \quad argmin_Q \mathbb{D}_{KL}[Q \,\|\, P] \ .$$

In view of this distinction, and given the central role played by the decomposability assumption on the second argument of the KL divergence (see, *e.g.*, Theorem 5.11), it is not unsurprising that variational methods should tend to benefit from different structural features than our methods do.

The reversal in the KL divergence coefficients and the ensuing disparity in the semantics of approximation errors highlight one of the most fundamental differences between the analysis of variational methods and the approximation paradigm suggested in the present work.

10.1.2 Particle filtering

An important class of inference methods for Bayesian networks is based on sampling or stochastic simulation, and is commonly referred to as particle filtering in this field. The idea is as simple as it is powerful: samples are generated and transmitted from one node to the next, in topological order, in such a way that the parent values are used to condition the sampling of the child, according to the conditional probability distributions in the model. A number of trajectories are obtained this way, and used to estimate the distribution of any variable of interest. Various methods exist to deal with evidence data, from the basic approach of discarding of any trajectory found incompatible with the conditioning data, to more sophisticated ones where the observed variables are not sampled, but used to weight the trajectories according to their posterior probability given the data [GSS93, DGA00]. The latter approach has been suggested for DBNs in [KKR95].

Improvements over basic particle filtering have been proposed in recent years, including a sampling-learning hybrid method for DBN inference due to Koller and Fratkina [KF98]. The difference with plain particle filtering is that the samples are not propagated as such, but, rather, are used to learn conditional distributions over the individual variables. The learned distributions are then used for further sampling.

The Rao-Blackwellization method provides another interesting compromize between sampling-based and message-based propagation. In essence, the Rao-Blackwell theorem is used to construct a lower-variance estimator of a variable of interest by selectively marginalizing out some of the variables. We refer the reader to [CR96, DGA00] for details.

In general, sequential Monte Carlo methods such as these have simplicity as one of their major advantages, the other one being that they may often be used in situations when all other approaches fail. Some drawbacks include the lack of absolute guarantees on the final result, and the often-made observation that analytical methods, when applicable, also tend to be more efficient. Due to statistical significance considerations, sampling based methods work best when the amount of information to be estimated is small. In other words, sampling will give much better results at estimating the marginal distribution of one variable of interest in a BN than a marginal over a fairly large number of them.

10.1.3 Accelerated exact inference

An approach to speeding up inference in DBNs without sacrificing to any approximation has been proposed by Kjærulff [Kjæ92]. It is based on the observation that, even though a DBN may contain many variables, in some cases the information is funneled from one state to the next through a "bottleneck" consisting of only a few state variables. Kjærulff's algorithm is a clever optimization of the junction tree inference algorithm for models featuring the bottleneck property.

The main appeal of this algorithm is, of course, that it provides exact results. However, its applicability remains confined to small networks with that particular topology.

10.1.4 Opportunistic approximation and mini-buckets

An approximate inference method bearing some resemblance to ours is the mini-bucket approach of Dechter and Rish [DR97].

This approach is a variant of the junction tree algorithm of [LS88], in which inference is conducted normally provided that all cliques in the junction tree are small. Should some clique exceed some threshold size, the computation of the clique potential will be approximated as a collection of projections over subsets of the clique, called mini-buckets. This decomposition is performed opportunistically as needed, in a greedy fashion.

Except for the fact that mini-buckets are defined for static BNs, the method bears some superficial similarity to our approximation, in that exceedingly large clusters are broken down into smaller ones. A first additional distinction is that, in our approach, inference is in fact carried out exactly; it is the result that is projected down to a specified set of clusters, in preparation for the next time slice. The inference process is accelerated *ipso facto*, for the reason that smaller output clusters typically imply smaller cliques in the junction tree. By contrast, the mini-bucket approach will approximate every large clique it encounters during the inference process itself. A second distinction is that, being exact until the final projection, our approximation algorithm affords many reassuring properties, such as the optimality of the result accuracy for a given representation structure (see Theorem 4.9), the contraction properties seen in Chapter 5, and the projection error results of Chapter 6. Mini-buckets offer different guarantees. A third distinction has to do with the very nature of the projection. In the mini-bucket algorithm, projections need not be confined within individual cliques of the junction tree; rather, the factors that intervene in the conputation of a large clique potential are partitioned prior to the operation, and each partition is multiplied out separately to produce a factor.

These differences emphasize that, ultimately, the two approaches were designed with rather different purposes in mind.

10.1.5 The factored frontier algorithm

The factored frontier algorithm recently due to Murphy and Weiss [MW01] is very closely related to our approximation approach.

In essence, the factored frontier defines a similar approximation as the one that places each variable in its individual cluster, referring to our algorithms of Chapter 4. However, instead of carrying out the approximate update as a conventional junction tree calibration followed by a projection, the inference step is implemented as "loopy belief propagation" [MWJ99], using factored distributions at all stages of the computation.

One benefit of this approach is that it remains applicable in models where even the fully factored version of our approximation would lead to impractically large cliques in the junction tree.

10.2 Learning

10.2.1 Variational learning

Variational learning [JJ00] is a class of algorithms for learning under conditions of partial observability, *i.e.*, when some of the parameters of the model to be learned pertain to hidden variables. Variational learning is closely related to the EM algorithm, which it generalizes. Both techniques approach the learning problem as an iterative two-phase optimization problem, in which the model parameters θ and a set of auxilliary parameters λ are iteratively improved by optimizing one while the other is kept constant; the auxilliary parameters λ are computed in the E-step, while the model parameters θ are optimized in the M-step.

The main differences with EM lie in the constraints placed on the auxilliary parameters, and on the optimization criterion. On the one hand, whereas the auxilliary parameters λ in EM are always the expected values of the missing data (or, equivalently, the expected sufficient statistics derived from them), in variational learning the λ are variational parameters indexing a probability distribution Q_λ from an unspecified family of distributions over the unobserved data. On the other hand, while the optimization criterion in EM is a measure of fitness of the model to the (completed) data, in variational learning the optimization is always connected to the minimization of the KL divergence between Q_λ and the true distribution P over the unobserved data, $\mathbb{D}_{KL}[Q_\lambda \,\|\, P]$. (Note that the arguments are reversed from their natural order, as in the case of variational inference; see Section 10.1.1.)

Variational learning is thus very similar to EM in spirit, but presents the advantage of the added flexibility in the choice of the Q_λ family. It also offers guarantees of optimality in terms of KL divergence to the true distribution.

The principles of variational learning have been used in a number of applications, including HMMs [Mac98], and DBNs [Att99].

10.2.2 Structure-based learning

In Chapter 9, we suggested one possible approach for discovering hidden variables while learning a dynamic model. Our approach was based on the detection of violations of the Markov property as hints of the existence of a hidden variable not captured by the current model.

Complementary approaches to discovering hidden variables from data have been proposed recently, such as those found in the work of Elidan *et al.* [ELFK00, EF01]. Similarly to our proposal of Section 9.2, these techniques attempt to infer the existence of hidden variables from structural patterns in models learned over oversable data. Specifically, these approaches are based on the observation that the dependencies induced by a hidden variable on the other variables of its family can no longer easily be represented, in the usual language of graphical models, if the hidden variable is removed. In other words, for most probability distributions $\mathbb{P}[A, B, C, D, H]$ that have a sparse BN representation, it is not the case that, summing out H, the resulting marginal $\mathbb{P}[A, B, C, D]$ will itself admit a BN representation over the variables that remain, other than a full clique. This suggests that any heavily connected subset of variables that appears in a learned model might in fact be subject to a common hidden influence.

The general idea behind structure-based discovery of hidden variables, thus, is to learn the structure over the known variables (whether observed or hidden) using a standard algorithm such as SEM [Fri97], then identify the full or almost full cliques in the learned network, replace their edges, and introduce new hidden variables. This idea is developed in [ELFK00], which also presents a greedy heuristic for finding almost full cliques of maximal size.

10.3 Decision-theoretic planning

Even though this dissertation is not directly concerned with the issue of controlling stochastic processes, the problems one faces in this task are not unrelated to the ones we have studied. Indeed, the field of decision-theoretic planning shares many of the challenges encountered with probabilistic inference and learning, with the added difficulty of accounting for the ramifications of the agent's own actions. We refer the reader to [BDH99] for a survey of the issues that arise when reasoning with partially observable Markov decision processes (POMDPs).

Among these challenges, the problem of compactly representating large structured state spaces in a form amenable to efficient decision making has recently received renewed attention. We mention the work of Poupart and Boutilier [PB00], in which they present a structured approximation scheme for tracking the state of POMDPs.

Chapter 11

Conclusion

The recent focus in AI on real-life systems has prompted the need for robust and efficient reasoning mechanisms. These reasoning mechanisms must accomodate our ignorance as well as the inherent uncertainty of the world's dynamics, and must provide a way to learn appropriate models from observed data. In addition, for an artificial agent to interact with the system, it must be able to use these models to decide on a proper course of action.

The probabilistic reasoning method is well suited to this endeavor. Besides a plethora of theoretical arguments for the soundness and robustness of the approach, the recent development of Bayesian Networks as efficient representations that afford an effective inference theory [Pea88, LS88] has generated a considerable amount of interest in the probabilistic method for reasoning about complex systems. Unfortunately, the time dimension still presents a difficult challenge for most of these techniques, which has typically restricted BNs to static or steady state applications. Clearly, the temporal aspect cannot be neglected in many applications, particularly those involving real-life systems that interact with each other and their environment.

11.1 Inference in complex stochastic systems

Temporal probabilistic inference deals with several important tasks, such as tracking the state of the system as it evolves, predicting its future behavior, explaining a sequence of observations, or deciding upon an appropriate course of action. These tasks and many others have at their core the problem of reasoning with a belief state (*i.e.*, a distribution over all possible states of the system at a given point in time). This reasoning paradigm has been widely successful for reasoning with flat representations of processes, such as hidden Markov models, as long as the state spaces are not too large [EAM95]. For more complex and structured processes, such as those represented by Dynamic Bayesian Networks, the need to maintain a belief state makes the reasoning task inescapably intractable.

In this context, a natural idea is to adopt a compact, factored representation of the belief state, and perform the temporal inference using such an approximate representation. We propose the first efficient such approach, in which a compactly represented belief state is maintained by first applying Bayesian updating using the model dynamics and observations, then projecting the result back into the class of compactly representable beliefs. Since these operations can be performed without ever expanding out the full state distribution explicitly, we only need to maintain compact belief states in memory, and therefore greatly reduce the computational complexity.

One immediate difficulty is that errors might accumulate beyond control due to the repetitive projections into the approximation class at each step of the reasoning process. We first show that this does not happen: Based on a novel contraction analysis for relative entropy, we show that the very stochasticity of the process causes such a rapid decay of all accumulated errors from the past, that the expected error at any time of the reasoning process remains bounded. We show that the rate at which these errors decay depends directly on the amount of interaction between the subprocesses according to which the belief state is factored. This analysis gives a principled way of dealing with large and structured stochastic processes: for example, as cars on a highway tend to interact only with their current neighbors, it is legitimate to (dynamically) reason about the entire highway in terms of small clusters of neighboring cars. Leveraging this theoretical analysis, we propose a very simple and practical inference algorithm for DBNs with well characterized accuracy and efficiency properties. Our inference algorithm is shown experimentally to track the state of a partially observed process with very high accuracy, at a fraction of the cost of the exact method. As the computational complexity remains polynomial for nicely structured processes (as many real-life systems are), our algorithm arguably offers the first viable approach to reasoning probabilistically about complex systems, by exploiting their structure.

We further refine our error propagation analysis to account for the richer structural features found in real-life systems. Realistic processes are composed of many sub-processes of similar structure, which may themselves be further decomposed in a hierarchical fashion. Interactions across processes are weaker and/or sparser than within a process, and usually take place through a limited number of channels, typically via the parent process in a hierarchical structure. We show how to exploit this wealth of structural information to guide the reasoning. Based on information-theoretic analyses, we provide a quantitative measure of the strength of interaction between subprocesses, and show how it captures intuitive notions such as weak and sparse interactions between processes. In turn, the theoretical framework gives a sound underpinning to one's intuition with regard to how such familiar notions can be exploited in a reasoning task.

11.2 Learning structured dynamic models

A frequent problem faced in the deployment of automated reasoning systems, is the acquisition of accurate system models. Complementary to the knowledge engineering approach, machine learning seeks to induce these models directly from observed data (which are usually inaccurate and incomplete). This task is a very difficult one, as it amounts to searching a very large space of possible models for those that adequately explain the observations. One of the best performing methods for parameter learning, known as the Expectation-Maximization algorithm, uses a bootstrapping approach: starting from some initial model, it uses a reasoning engine to estimate the state of the system, then uses this estimate to refine the model parameters; the process is repeated iteratively until convergence to a local optimum. Unfortunately, the cost of probabilistic inference, further compounded by its repeated iterative application, is an obstacle to the usability of EM in practical settings.

We show how to use our approximate inference algorithm as a substitute for the inference component in EM. The theoretical justification is based on a extension of our contraction results to the case of bidirectional messages used in EM. We offer ample experimental evidence that the resulting algorithm is able to learn structured DBN parameters with good accuracy. Compared to the standard EM algorithm, the cost of learning is reduced dramatically with almost imperceptible degradation of the model quality.

We also consider the issue of online learning, in which the model parameters are to be adapted in real time as the process evolves. This mode of operation poses an additional difficulty for regular EM, which requires the entire sequence of observations in order to re-estimate the model parameters at each iteration. Learning based on real-time filtering approaches that only take the past observations into account is another possibility, but discarding all data from the future typically leads to less accurate models. The median approach is to base learning on *smoothing*, in which a short window of future "lookahead" observations is considered in addition to the past ones. Smoothing is known to enhance the accuracy of state estimates in Kalman filters, and has been used heuristically in other settings [WH89, CC91]. We leverage our contraction analysis to give a theoretical justification to smoothing with a limited window: essentially, we show that all future observations beyond a window of reasonable size can be safely ignored. This also justifies an online modification of EM, in which parameter re-estimation is performed incrementally using small windows of observations around the present time. We demonstrate the effectiveness of this approach by learning the parameters of a medium-size DBN online using a short lookahead, without loss of accuracy with respect to the impractical full EM. By combining this partial smoothing with the approximate representation of belief states described above, we are able to learn structured DBNs orders of magnitude faster than was previously possible, at a comparable level of accuracy.

Finally, beyond merely learning DBN parameters in a fixed network structure, we present a preliminary investigation of the general problem of learning an entire dynamic model from minimally processed, partially observed data. We first propose a method to efficiently search and learn DBN network structures, which combines existing structural search techniques using EM [Fri97, FMR98] with our understanding of richly structured processes as we previously studied, in order to improve upon learning efficiency. We also suggest a way not only to learn a dependency graph between the variables that appear in the data set, but also to discover the existence of hidden state variables which are *a priori* unknown, based on observed violations of the Markovian property that such hidden variables would cause. For this problem, we offer a few preliminary experiments whereby entire DBNs are induced from raw time series data.

11.3 Future research directions

A particularly intriguing idea regarding the theoretical analysis lies in the unification of the contraction results of Chapter 5 and the projection error bounds of Chapter 6. As we mentioned earlier, the current fit between the two kinds of analyses is not as natural as one might have hoped, in spite of their profound mathematical similarity. It appears that the difficulty might reside in the virtual decomposition of the elementary update into a pure stochastic transition followed by a mere conditioning. If the imaginary meeting point could be dispensed with, the hope is that one might obtain stronger bounds for both aspects of the error at once. Such a unification is suggested to us from the study of sparsely interacting processes in Section 6.5, where ideas borrowed from the contraction analysis blend nicely into a decidedly projection-oriented error bound to produce a hybrid result over several time slices, in what could be a sneak preview of what a unified analysis might reveal.

One important aspect of reasoning under uncertainty that we have deliberately not covered in this work, is that of decision-theoretic planning and control. The same representation issues we are facing in inference and learning remain very much present indeed when the agent not only observes and learns, but is also allowed to act. It is our opinion that the study of factored approximations in such context would be of great interest.

Bibliography

[AL95] M. Artzrouni and X. Li, *A note on the coefficient of ergodicity of a column-allowable nonnegative matrix*, Linear algebra and applications **214** (1995), 93–101.

[Ast65] K.J. Aström, *Optimal control of Markov decision processes with incomplete state estimation*, J. Math. Anal. Applic. **10** (1965), 174–205.

[Att99] H. Attias, *Blind source separation by dynamic graphical models*, Proc. Int. Conf. Art. Neur. Net., 1999, pp. 126–131.

[Bau72] L.E. Baum, *An inequality and associated maximization technique in statistical estimation of probabilistic functions of a markov process*, Inequalities **3** (1972), 1–8.

[BDH99] C. Boutilier, T. Dean, and S. Hanks, *Decision-theoretic planning: Structural assumptions and computational leverage*, J. Art. Int. Res. **11** (1999), 1–94.

[BFK99] X. Boyen, N. Friedman, and D. Koller, *Discovering the hidden structure of complex dynamic systems*, Proc. UAI, vol. 15, 1999, pp. 91–100.

[BK98a] X. Boyen and D. Koller, *Approximate learning of dynamic models*, Proc. NIPS, vol. 11, 1998, pp. 396–402.

[BK98b] X. Boyen and D. Koller, *Tractable inference for complex stochastic processes*, Proc. UAI, vol. 14, 1998, pp. 33–42.

[BK99] X. Boyen and D. Koller, *Exploiting the architecture of dynamic systems*, Proc. AAAI, vol. 16, 1999, pp. 313–320.

[BMR97] J. Binder, K. Murphy, and S.J. Russell, *Space-efficient inference in dynamic probabilistic networks*, Proc. IJCAI, 1997.

[Bun91] W. Buntine, *Theory refinement on Bayesian networks*, Proc. UAI, 1991, pp. 52–60.

[CC91] C.K. Chui and G. Chen, *Kalman filtering with real-time applications*, Springer-Verlag, 1991.

[CGH95] D. Chickering, D. Geiger, and D. Heckerman, *Learning Bayesian networks: search methods and experimental results*, Proc. AI & Stats, 1995, pp. 112–128.

[CL68] C.K. Chow and C.N. Liu, *Approximating discrete probability distributions with dependence trees*, IEEE Transactions on Information Theory **14** (1968), 462–467.

[CR96] G. Casella and C.P. Robert, *Rao-blackwellisation of sampling schemes*, Biometrika **83** (1996), 81–94.

[CT91] T. Cover and J. Thomas, *Elements of information theory*, Wiley, 1991.

[DG93] P. Dagum and A. Galper, *Forecasting sleep apnea with dynamic network models*, Proc. UAI, 1993, pp. 64–71.

[DGA00] A. Doucet, S. Godsill, and C. Andrieu, *On sequential monte carlo sampling methods for bayesian filtering*, Statistics and Computing **10** (2000), no. 3, 197–208.

[DGH92] P. Dagum, A. Galper, and E. Horvitz, *Dynamic network models for forecasting*, Proc. UAI, 1992.

[DK89] T. Dean and K. Kanazawa, *A model for reasoning about persistence and causation*, Computational Intelligence **5** (1989), no. 3, 142–150.

[DLR77] A.P. Dempster, N.M. Laird, and D.B. Rubin, *Maximum-likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society **B39** (1977), 1–38.

[DR97] R. Dechter and I. Rish, *A scheme for approximating probabilistic inference*, Proc. UAI, 1997.

[Dur95] R. Durrett, *Probability: Theory and examples*, second edition ed., Duxbury Press, 1995.

[EAM95] Robert J. Elliott, Lakhdar Aggoun, and John B. Moore, *Hidden Markov models: Estimation and control*, Springer-Verlag, 1995.

[EF01] G. Elidan and N. Friedman, *Learning the dimensionality of hidden variables*, Proc. UAI, 2001.

[ELFK00] G. Elidan, N. Lotner, N. Friedman, and D. Koller, *Discovering hidden variables: a structure-based approach*, Proc. NIPS, 2000.

[FHKR95] J. Forbes, T. Huang, K. Kanazawa, and S.J. Russell, *The BATmobile: Towards a Bayesian automated taxi*, Proc. IJCAI, 1995, pp. 1878–1885.

[FMR98] N. Friedman, K. Murphy, and S.J. Russell, *Learning the structure of dynamic probabilistic networks*, Proc. UAI, 1998, pp. 139–147.

[FPN99] N. Friedman, D. Peér, and I. Nachman, *Learning Bayesian network structure from massive datasets: The "sparse candidate" algorithm*, Proc. UAI, 1999.

[Fri97] N. Friedman, *Learning belief networks in the presence of missing values and hidden variables*, Proc. ICML, 1997.

[Fri98] N. Friedman, *The Bayesian structural EM algorithm*, Proc. UAI, 1998, pp. 129–138.

[Gha98] Z. Ghahramani, *Learning dynamic bayesian networks*, Adaptive Processing of Sequences and Data Structures (C.L. Giles and M. Gori, eds.), Lecture Notes in Artificial Intelligence, Springer-Verlag, 1998, pp. 168–197.

[GJ96] Z. Ghahramani and M.I. Jordan, *Factorial hidden Markov models*, Proc. NIPS, vol. 8, 1996.

[GMMO00] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan, *Clustering data streams*, Proc. FOCS, 2000.

[Gol80] M.C. Golumbic, *Triangulated graphs*, Algorithmic Graph Theory and Perfect Graphs, Academic Press, 1980, pp. 98–100.

[GSS93] N.J. Gordon, D.J. Salmond, and A.F.M. Smith, *Novel approach to nonlinear/non-gaussian bayesian state estimation*, Proc. Inst. Elect. Eng. F **140** (1993), 107–113.

[GVP90] D. Geiger, T. Verma, and J. Pearl, *Identifying independence in Bayesian networks*, Networks **20** (1990), 507–534.

[HD94] C. Huang and A. Darwiche, *Inference in belief networks: a procedural guide*, Int. J. Approx. Reas. **11** (1994), 1–158.

[Hec99] D. Heckerman, *A tutorial on learning with Bayesian networks*, Learning in Graphical Models (M.I. Jordan, ed.), MIT Press, 1999.

[HGC95] D. Heckerman, D. Geiger, and D.M. Chickering, *Learning Bayesian networks: The combination of knowledge and statistical data*, Machine Learning **20** (1995), 197–243.

[HKM+94] T. Huang, D. Koller, J. Malik, G. Ogasawara, B. Rao, S.J. Russell, and J. Weber, *Automatic symbolic traffic scene analysis using belief networks*, Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI '94), 1994, pp. 966–972.

[HMC97] D. Heckerman, C. Meek, and G. Cooper, *A Bayesian approach to causal discovery*, Technical report MSR-TR-97-05, Microsoft Research, 1997.

[HS94] D. Heckerman and R. Shachter, *A decision-based view of causality*, Proc. UAI, vol. 10, 1994, pp. 302–310.

[JGJS99] M.I. Jordan, Z. Ghahramani, T. Jaakkola, and L.K. Saul, *An introduction to variational methods for graphical models*, Machine Learning **37** (1999), no. 2, 183–233.

[JJ94] F.V. Jensen and F. Jensen, *Optimal junction trees*, Proc. UAI, vol. 10, 1994, pp. 360–366.

[JJ00] T. Jaakkola and M. Jordan, *Bayesian parameter estimation via variational methods*, Statistics and Computing **10** (2000), 25–37.

[JKOP89] F.V. Jensen, U. Kjærulff, K.G. Olesen, and J. Pedersen, *An expert system for control of waste water treatment—a pilot project*, Tech. report, Judex Datasystemer A/S, Aalborg, Denmark, 1989, In Danish.

[JLO90] F.V. Jensen, S.L. Lauritzen, and K.G. Olesen, *Bayesian updating in recursive graphical models by local computations*, Computational Statistical Quarterly **4** (1990), 269–282.

[Kal60] R.E. Kalman, *A new approach to linear filtering and prediction problems*, J. of Basic Engineering **82** (1960), 34–45.

[KF98] D. Koller and R. Fratkina, *Using learning for approximation in stochastic processes*, Proc. ML, 1998.

[Kjæ90] U. Kjærulff, *Triangulation in graphs—algorithms giving small total state space*, Tech. Report R-90-09, Dept. of Math. and Comp. Sci., Aalborg University, Denmark, 1990.

[Kjæ92] U. Kjærulff, *A computational scheme for reasoning in dynamic probabilistic networks*, Proc. UAI, 1992, pp. 121–129.

[KKR95] K. Kanazawa, D. Koller, and S.J. Russell, *Stochastic simulation algorithms for dynamic probabilistic networks*, Proc. UAI, 1995, pp. 346–351.

[KL51] S. Kullback and R.A. Leibler, *On information and sufficiency*, Annals of Mathematical Statistics **22** (1951), no. 1, 79–86.

[Lau95] S.L. Lauritzen, *The EM algorithm for graphical association models with missing data*, Computational Statistics and Data Analysis **19** (1995), 191–201.

[LDLL90] S.L. Lauritzen, A.P. Dawid, B.N. Larsen, and H.-G. Leimer, *Independence properties of directed Markov fields*, Networks **20** (1990), 491–505.

[LP01] U. Lerner and R. Parr, *Inference in hybrid networks: Theoretical limits and practical algorithms*, Proc. UAI, 2001, pp. 310–318.

[LS88] S.L. Lauritzen and D.J. Spiegelhalter, *Local computations with probabilities on graphical structures and their application to expert systems*, J. Roy. Stat. Soc. **B 50** (1988), 157–224.

[Mac98] D.J.C. MacKay, *Ensemble learning for hidden Markov Models*, Tech. report, Cavendish Laboratory, University of Cambridge, 1998.

[Mur98] K.P. Murphy, *Inference and learning in hybrid bayesian networks*, Technical Report CSD-98-990, U.C. Berkeley, 1998.

[MW01] K. Murphy and Y. Weiss, *The factored frontier algorithm for approximate inference in dbns*, Proc. UAI, 2001.

[MWJ99] K.P. Murphy, Y. Weiss, and M.I. Jordan, *Loopy belief propagation for approximate inference: an empirical study*, Proceedings of Uncertainty in AI, 1999.

[NH98] R.M. Neal and G.E. Hinton, *A view of the EM algorithm that justifies incremental, sparse, and other variants*, Learning in Graphical Models (M.I. Jordan, ed.), Kluwer, 1998.

[PB00] P. Poupart and C. Boutilier, *Value-directed belief state approximation for POMDPs*, Proc. UAI, 2000, pp. 497–506.

[Pea88] J. Pearl, *Probabilistic reasoning in intelligent systems: Networks of plausible inference*, Morgan Kaufmann, 1988.

[Pro92] G. Provan, *Tradeoffs in constructing and evaluating temporal influence diagrams*, Proc. UAI, 1992, pp. 40–47.

[RJ86] L. Rabiner and B. Juang, *An introduction to hidden Markov models*, IEEE Acoustics, Speech & Signal Processing **3** (1986), no. 1, 4–16.

[Ros96] S. Ross, *Stochastic processes*, second ed., Wiley, 1996.

[Sch78] G. Schwarz, *Estimating the dimension of a model*, Annals of Statistics **6** (1978), 461–464.

[SGS93] H. Spirtes, C. Glymour, and R. Scheines, *Causation, prediction, and search*, Springer-Verlag, 1993.

[SHJ96] P. Smyth, D. Heckerman, and M.I. Jordan, *Probabilistic independence networks for hidden Markov probability models*, Neural Computation **9** (1996), no. 2, 227–269.

[Sim62] H. Simon, *The architecture of complexity*, Proc. Am. Phil. Soc. **106** (1962), 467–482.

[SJ95] L.K. Saul and M.I. Jordan, *Exploiting tractable substructure in intractable networks*, Proc. NIPS, vol. 7, 1995.

[SK89] R. Shachter and R. Kenley, *Gaussian influence diagrams*, Management Science **35** (1989), no. 5, 527–550.

[SS90] P.P. Shenoy and G.R. Shafer, *Axioms for probability and belief-function propagation*, Proc. UAI, 1990, pp. 169–198.

[TY84] R.E. Tarjan and M. Yannakakis, *Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs and selectively reduce hypergraphs*, SIAM J. Comp. **13** (1984), no. 3, 566–579.

[WG90] A. Weigend and N. Gershenfeld, *Time-series competition*, Proc. Nonlinear Modeling and Forecasting, vol. XII, Santa Fe institute, Addison-Wesley, 1990.

[WH89] M. West and J. Harrison, *Bayesian forecasting and dynamic models*, Springer-Verlag, 1989.

[ZR98] G. Zweig and S.J. Russell, *Speech recognition with dynamic bayesian networks*, Proc. AAAI, 1998, pp. 173–180.