UNIVERSITY OF CALIFORNIA,

IRVINE

Modeling of Multivariate Time Series Using Hidden Markov Models

DISSERTATION

submitted in partial satisfaction of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

Sergey Kirshner

Dissertation Committee:

Professor Padhraic Smyth, Chair

Professor Rina Dechter

Professor Richard Granger

Professor Max Welling

2005

The dissertation of Sergey Kirshner

is approved and is acceptable in quality

and form for publication on microfilm:

Committee Chair

University of California, Irvine

2005

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to thank my advisor Padhraic Smyth for his guidance and unconditional support during my years at UCI, and for teaching me so much. Padhraic, I do not know whether anyone else would have gone as far to accommodate me as you did – thank you! Good portion of the credit also belongs to my collaborator Andy Robertson for making my research feel so much more tangible. I am very grateful to my committee, Max Welling, Rina Dechter, and Rick Granger, for insightful suggestions and making sure I am not completely off-base. Special thanks to my collaborators from Lawrence Livermore National Laboratory, Chandrika Kamath and Erick Cantú-Paz, for involving me in their exciting project.

My graduate life would not have been the same without my officemate, Sridevi, and everyone else in the Datalab, Dima, Dasha, Igor, Xianping, Scott G., Joshua, Scott W., Seyoung, Naval, Chaitanya, Lucas, just to name a few. Thank you all for making it great and memorable! I also wish to thank my graduate counselors, Kris Bolcer and Mylena Wypchlak, for answering a million of repeating questions with a welcoming smile; David Eppstein, Sandra Irani, and others in the theory group for welcoming me when I first came to UCI; David Kay for pleasant conversations and tips on teaching; Mike Dillencourt for memorable pizza nights while grading 6A exams; staff at the grant and travel offices, especially Patty Jones, for making my conference travel painless (well, mostly painless).

Finally, I owe a lot to my family; my parents for enduring my constant traveling and just being there for me; my sister and her family for a friendly chat and support whenever I need it; and most importantly, to my wife Julia for constant encouragement and just the right amount of nagging.

CURRICULUM VITAE

Sergey Kirshner

Date of Birth: January 3, 1976 in Odessa, USSR

Education

Ph.D.	03/2005	UC Irvine	Information and Computer Science
M.S.	12/2001	UC Irvine	Information and Computer Science
B.A.	05/1998	UC Berkeley	Mathematics & Computer Science

Honors and Awards

RIACS Summer Student Research Program Grant, 2000

Honorable mention at the 56th William Lowell Putnam Mathematical Competition (top 2%), 1995

Professional Experience

09/1999–present Research Assistant, School of ICS, UC Irvine, CA

06-08/2002 Instructor, School of ICS, UC Irvine, CA

09/2000–03/2001 Teaching Assistant, School of ICS, UC Irvine, CA

06-09/2000 Student Research Scientist, RIACS/NASA ARC, Moffett Field, CA

06/1998–08/1999 Systems Analyst, NASA Ames Research Center, Moffett Field, CA

Peer-Reviewed Publications List

A.W. Robertson, **S. Kirshner,** and P. Smyth, 'Daily rainfall occurrence over Northeast Brazil and its downscalability using a hidden Markov model.' *Journal of Climate,* 17(22):4407–4424, 2004

S. Kirshner, P. Smyth, A.W. Robertson, 'Conditional Chow-Liu tree structures for modeling discrete-valued vector time series,' UAI-2004, pp. 317-324, 2004

S. Kirshner, S. Parise, and P. Smyth, 'Unsupervised learning from permuted data,' ICML-2003, pp. 345-352, 2003

S. Kirshner, I.V. Cadez, P. Smyth, and C. Kamath, 'Learning to classify galaxy shapes using the EM algorithm,' NIPS-2002, pp. 1497-1504, 2003

S. Kirshner, I.V. Cadez, P. Smyth, C. Kamath, and E. Cantú-Paz, 'Probabilistic model-based detection of bent-double radio galaxies,' ICPR-2002, vol.2, 2002, pp.499-502

D. Wolpert, **S. Kirshner,** C. Merz, and K. Tumer, 'Adaptivity in agent-based routing for data networks,' Agents-2000, pp.396-403

ABSTRACT OF THE DISSERTATION

Modeling of Multivariate Time Series
Using Hidden Markov Models

By

Sergey Kirshner

Doctor of Philosophy in Information and Computer Science

University of California, Irvine, 2005

Professor Padhraic Smyth, Chair

Vector-valued (or multivariate) time series data commonly occur in various sciences. While modeling univariate time series is well-studied, modeling of multivariate time series, especially finite-valued or categorical, has been relatively unexplored. In this dissertation, we employ hidden Markov models (HMMs) to capture temporal and multivariate dependencies in the multivariate time series data. We modularize the process of building such models by separating the modeling of temporal dependence, multivariate dependence, and non-stationary behavior. We also propose new methods of modeling multivariate dependence for categorical and real-valued data while drawing parallels between these two seemingly different types of data. Since this work is in part motivated by the problem of prediction precipitation over geographic regions from the multiple weather stations, we present in detail models pertinent to this hydrological application and perform a thorough analysis of the models on data collected from a number of different geographic regions.

Chapter 1

Introduction

High-dimensional multivariate data sets occur in a large number of problem domains. In many cases, these data sets have either sequential or temporal structure. For example, economics provides us with time series of stock or index prices; speech and music patterns can be thought of as sequences of signals at different frequencies; atmospheric scientists has been collecting measurements for locations all over the planet for the last century. Modeling of and forecasting from such data is an important problem in all of the respective areas. Thus, it is important to develop efficient, accurate, and flexible data analysis techniques for multivariate time series data.

While there is a significant volume of literature on time series modeling, the majority of it concerns with the univariate case. Extensions of real-valued univariate models to vector observations often involve high-dimensional auto-covariance matrices and are linear in structure (e.g., Brockwell and Davis, 2002; Shumway and Stoffer, 2000; Hamilton, 1994). Direct modeling of multivariate sequences for categorical data typically requires an even larger number of parameters than the real-valued case since we cannot use smoothness, linearity, or normality property of the real-valued data. Modeling of vector series for finite domains is a relatively new research area, and techniques such as dynamic Bayesian networks (Murphy, 2002) and dynamic graphical models (e.g., West and Harrison, 1997) are being explored for this problem. Mixed (both continuous- and discrete-valued) data modeling is even less explored due to the difficulty of modeling interactions between finite- and real-valued variables.

The approach in this thesis is to model discrete-time multivariate time series data using hidden Markov models (HMMs), a special class of dynamic Bayesian networks. HMMs assume that the observed data is generated via finite-valued latent (unobserved) process. This unobserved process is assumed to be in one of a finite number of discrete states at each discrete time point and to transition stochastically in a Markov fashion given the previous state or states. The observed data at each time point depends only on the value of the corresponding hidden state and is independent of the rest of the data. The HMM can be viewed as separating the temporal and multivariate aspects of the data, with the Markov dependency capturing the temporal property, and the static probability distribution of a vector at a given time given the hidden state capturing multivariate dependencies.

1.1 Application: Multi-Site Precipitation Modeling

One of the main applications for the research in this thesis is modeling and prediction of rainfall occurrences and amounts, across networks of locations. Precipitation prediction is an important problem in hydrology and agriculture and has a long history of research. Standard atmospheric modeling techniques are not particularly effective for this type of problem since precipitation is a local process and can have a large variance over time, i.e., precipitation can be unpredictable and have high entropy. General circulation models (GCMs) (e.g., Randall, 2000), the most commonly used tool for seasonal climate prediction, do not perform well in modeling local precipitation (e.g. Semenov and Porter, 1995) as they operate on a much coarser grid (usually, $2.5^\circ \times 2.5^\circ$) than the spatial scale of precipitation. Thus scientists often employ statistical models based on historical precipitation data (possibly with other atmospheric measurements and/or GCM predictions) for the task local precipitation modeling and prediction. A short review of existing statistical method for precipitation modeling can be found in Section 8.1.

One of the goals of this work is to develop tools for building statistical models for multi-site daily precipitation. Due to high variance, we obviously cannot make accurate predictions for a specific day and a specific location based just on the historical data. We can, however, build generative models that can produce realistic simulated daily rainfall sequences preserving observed occurrence (and amounts) correlations between the stations, precipitation wet/dry spell durations, and can reasonably predict how wet the predicted season would be given some additional atmospheric information. Simulated runs of this model can be used to improve water resource management or to influence decisions in crop planning. Of particular interest are seasonal predictions and simulations, where the statistical characteristics of a winter season (for example) are forecast by the model conditioned on the "state" of the climate system at the start of the season.

To give a flavor of the problem, we preview a precipitation occurrence data set collected from 10 rain stations in the Ceará region of Northeastern Brazil over wet seasons of Feb–April (90 days beginning February 1) from 1975–2002, provided by Fundaco Cearense de Meteorologia (FUNCEME; Figure 1.1). Four of the seasons (1976, 78, 84, 86) contained significant number of missing observations and were not considered. Figure 1.2 shows the locations of individual rain stations and their daily rainfall occurrence probabilities, one of the data statistics we would like our model to reproduce. As Figure 1.1 suggests, the data has very high variance; some of the variability can be visualized by looking at seasonal occurrence probabilities for each station (Figure 1.3). The problem of modeling this *interannual variability* recurs throughout atmospheric sciences and is quite important. Another problem of importance in this general context is modeling of spatial dependence — we employ correlation[1] as one of the measures of co-occurrence of rainfall for pairs of stations. Figure 1.4 summarizes pairwise correlations collectively for all years in the data set. Note that it is not sufficient to forecast the seasonal statistics of rainfall at a single station as the spatial correlation structure should also be retained. Finally, atmospheric scientists are interested in models matching the distribution of dry (and sometimes, wet) spell lengths.[2] These spell lengths for individual stations often exhibit geometric behavior (Figure 1.5) as is well known in precipitation modeling literature (e.g., Wilks and Wilby, 1999). Thus, the challenge is to develop methods that can take as input data such as that shown in Figure 1.1 and produce predictions for future seasons that are accurate across multiple statistical criteria (e.g., mean occurrence, variability, spatial correlations, run lengths).

[1] Linear (Pearson's) correlation is defined as $\rho_{xy} = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}}$.

[2] A dry (wet) spell of length n is an event of n consecutive days without (with) precipitation.

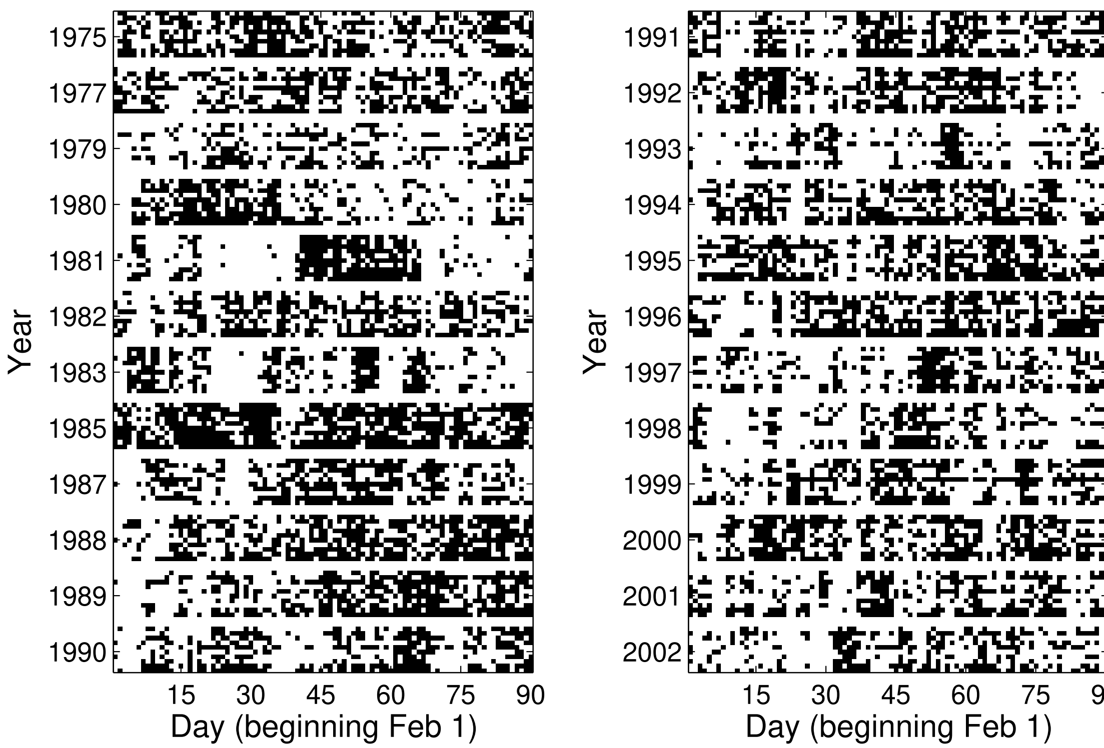

Figure 1.1: Visualization of the Ceará data set. Each of 24 seasons consists of 90 observations for each of 10 stations. Stations are displayed top to bottom with black squares indicating rain and white squares indicating no rain.

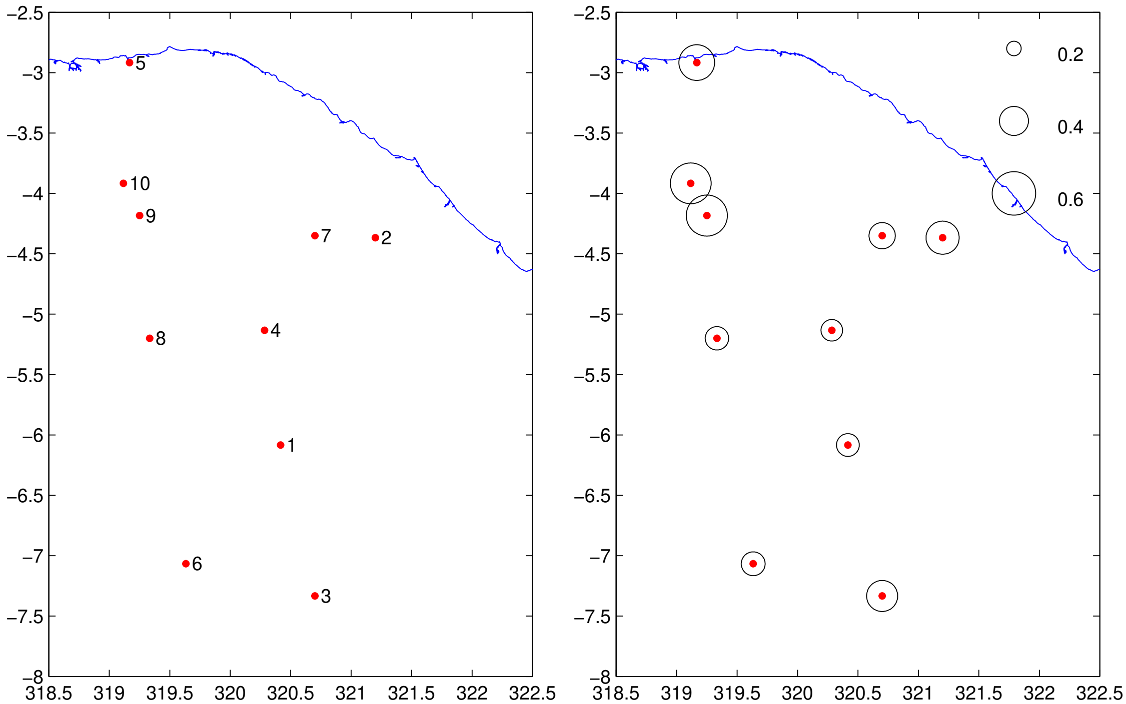

Figure 1.2: A map of Ceará region with rainfall station locations (left) and probability of precipitation (right). The stations are (with elevation): (1) Acopiara (317 m), (2) Aracoiaba (107 m), (3) Barbalha (405 m), (4) Boa Viagem (276 m), (5) Camocim (5 m), (6) Campos Sales (551 m), (7) Caninde (15 m), (8) Crateus (275 m), (9) Guaraciaba Do Norte (902 m), and (10) Ibiapina (878 m). Circle radius denotes the February–April climatological daily rainfall probability for 1975–2002.

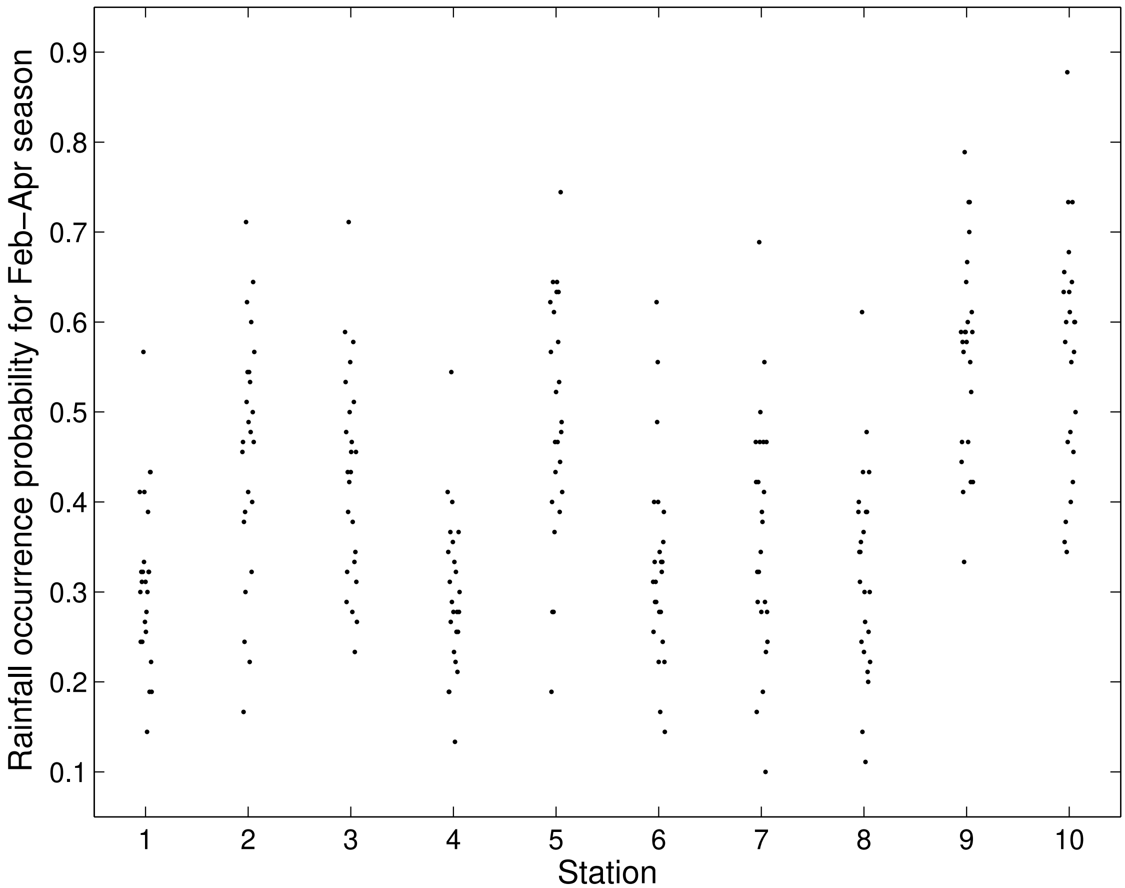

Figure 1.3: Annual occurrence probabilities for each station in the Ceará network.

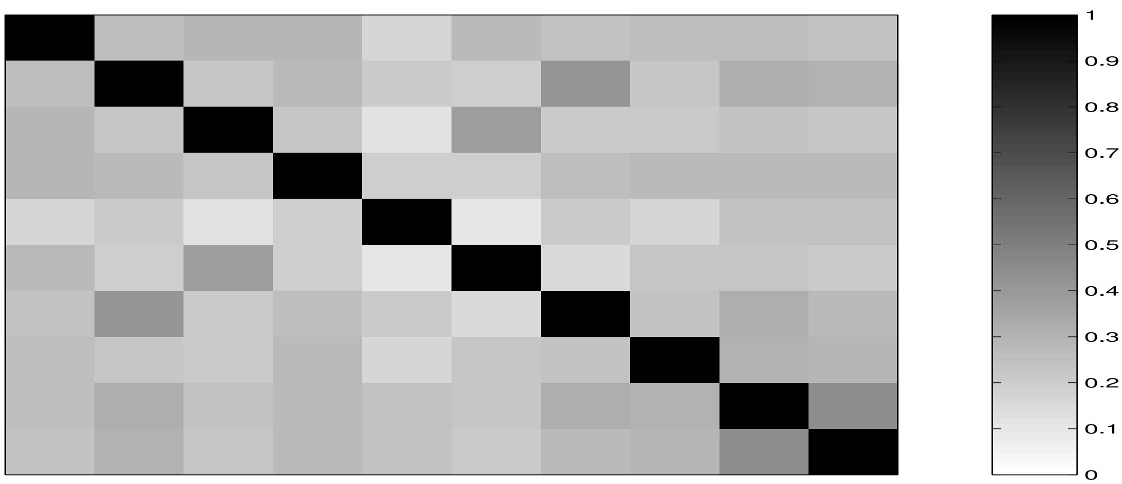

Figure 1.4: Pairwise correlations of rainfall daily occurrence for the Ceará network.

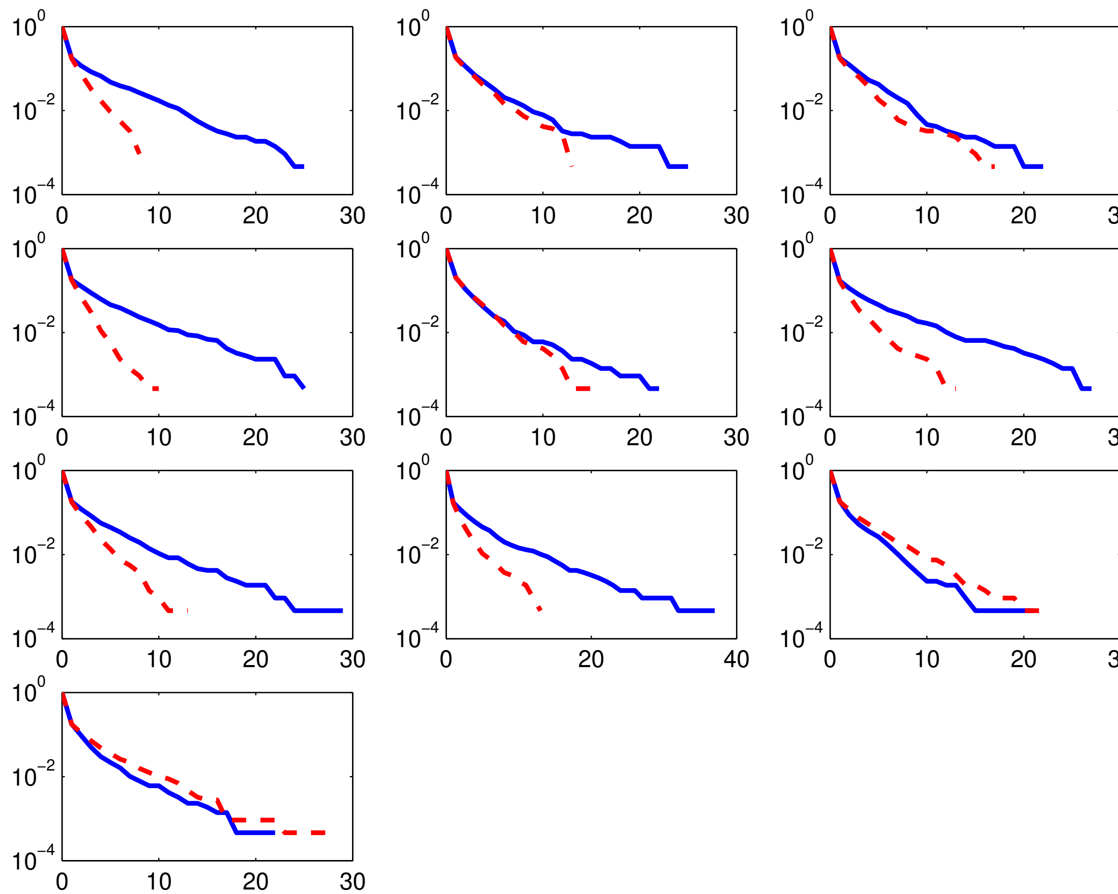

Figure 1.5: Spell length distribution per station for the Ceará network. Stations are displayed left-to-right and top-to-bottom (e.g., first row contains plots for stations 1,2,3). x-axes indicate spell length; y-axes indicate probability of spell duration of greater or equal length. Dry spells are displayed in solid (blue); wet spells are displayed in dashed (red).

1.2 High Level Overview of Related Work

Table 1.1 provides a crude and somewhat incomplete overview of commonly used models for multivariate categorical data with latent states. The x-axis in the table indicates the type of model being used for the hidden states, and the y-axis indicates the type of multivariate model being used on observations. By decomposing the problem into (a) temporal latent structure and (b) multivariate observation models, we can "cover" the space of different modeling possibilities within the general class of HMM-type models.

Most of the previously studied models fall along the lines of learning of multivariate structure without temporal component (left side of Table 1.1) or temporal structure without multivariate component (top part of Table 1.1). The work in this thesis will focuses on modeling both aspects, as reflected in the table.

1.3 Contributions

The primary novel contributions of this thesis are:

- The conditional Chow-Liu forest model and associated learning algorithm for conditional probability distributions on multivariate categorical data (Section 4.2.1, Kirshner et al. (2004));
- Hidden Markov models with Chow-Liu trees and conditional Chow-Liu forests and their learning algorithms for multivariate time series modeling (Section 6.2, Kirshner et al. (2004));
- Derivation of analytic expressions for covariance matrices for multivariate tree-structured Gaussian models (Section 5.2.2);

- Product of univariate conditional maximum entropy model and associated learning algorithm for conditional probability distributions on multivariate categorical data; HMM with the above model for data distribution in each hidden state (Sections 4.3.2, 4.3.5, 6.3.3);

- A new type of HMM for multi-site precipitation amount modeling (Section 6.1.2).

Other contributions are:

- Derivation of the update rules for the transition component of non-homogeneous HMMs (Appendix A.1, Robertson et al. (2003));

- Empirical study of the sensitivity of the parameter estimation to the amount of training data (Section 7);

- Comprehensive empirical study of HMM performance on the precipitation data (Chapter 8, Robertson et al. (2004); Kirshner et al. (2004));

1.4 Thesis Outline

Chapter 2 provides the basic notation and concepts of graphical models used throughout the thesis. Chapter 3 introduces and provides an overview of hidden Markov models. Chapter 4 describes models for the multivariate categorical data. Chapter 5 deals with models on multivariate real-valued data concentrating on Gaussian models and touching upon the exponential distributions. Chapter 6 ties up the results from Chapters 4 and 5 to describe the main result of this thesis, HMM-based multivariate time series models. Chapter 7 demonstrate the effectiveness of the models on simulated data while Chapter 8 applies the models to precipitation occurrence data for several geographical regions. Finally, Chapter 9 summarizes the main results and outlines remaining questions and possible future approaches.

Table 1.1: Models for multivariate categorical data.

Multivariate Structure	Latent Structure				
	No Explicit Temporal Component			Explicit Temporal Component	
	No Latent Variables	Mixture	Non-homogeneous Mixture	HMM	Non-homogeneous HMM
Univariate (Bernoulli)	e.g., Duda et al. (2000)	e.g., McLachlan and Peel (2000)	Jordan and Jacobs (1994)	Baum et al. (1970); Rabiner (1989)	Bengio and Frasconi (1995); Bengio and Bengio (1996); Meilă and Jordan (1996)
Conditional Independence	e.g., Duda et al. (2000)	e.g., McLachlan and Peel (2000)	Jordan and Jacobs (1994)	e.g., Zucchini and Guttorp (1991)	Hughes and Guttorp (1994)
Multivariate Probit (binary only)	e.g., Ashford and Sowden (1970); Chib and Greenberg (1998)				
Chow-Liu Trees	Chow and Liu (1968)	Grim (1984); Meilă and Jordan (2000)		Kirshner et al. (2004), **thesis**	**thesis**
MaxEnt (exponential)	Brown (1959); Darroch and Ratcliff (1972)	Pavlov et al. (2003)		Hughes et al. (1999), **thesis**	Hughes et al. (1999), **thesis**

Chapter 2

Preliminaries

This chapter defines the notation and briefly describes some of the important concepts used in the remainder of this thesis.

2.1 Notation

In the interest of consistency and for easier readability, we list here the general notation adopted in the rest of the thesis. Deviations from this notation will be mentioned prior to use.

Sets are denoted by calligraphic capital letters (e.g., $\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \dots$) except for when the sets already have established symbols (e.g., $\mathbb{R}$). Elements or members of sets are denoted by lowercase roman letters (e.g., $a \in \mathcal{A}$), and subsets are denoted by uppercase roman letters (e.g., $A \subseteq \mathcal{A}$).

All random variables are denoted in capital Roman letters (e.g., $X, Y, Z, \dots$); in addition, multivariate or vector random variables are denoted in boldface (e.g., $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \dots$). Values of random variables are denoted by lowercase letters with vectors in boldface (e.g., $X = x$ or $\boldsymbol{X} = \boldsymbol{x}$). Probability distributions defined on the discrete-valued domains are denoted by P (e.g., $P(X)$) while probability density functions (PDFs) defined over continuous domains are denoted by $p(X = x)$. A probability distribution $P(X)$ (or $p(X)$) value at a given point x $P(X = x)$ will be denoted by $P(x)$ (or, respectively, $p(x)$) for short. For a vector random variable $\boldsymbol{X} = (X_1, \ldots, X_M)$, individual components or variables will be indexed by one of two ways: (1) by a number $1, \ldots, M$, e.g., X_1 or X_i for $i = 1, \ldots, M$, or (2) by an element of a set $\mathcal{V}$ consisting of M elements where each $v \in \mathcal{V}$ corresponds to exactly one component of $\boldsymbol{X}$, e.g., $u \in \mathcal{V}$ corresponds to X_u. The latter notation also allows us to denote multiple variables from a vector random variable, e.g., if $A = \{a, b, c\}$, $\boldsymbol{X}_A = \{X_a, X_b, X_c\}$. We will also use this index set notation for the domain $\mathcal{X}$ of $\boldsymbol{X}$, e.g., $\boldsymbol{x}_A \in \mathcal{X}_A$.

All vectors are assumed to be *column* vectors. A transpose of a vector $\boldsymbol{x}$ is denoted by $\boldsymbol{x}'$. A transpose of a matrix $\boldsymbol{A}$ is denoted by $\boldsymbol{A}'$.

Individual model parameters are denoted by Greek letters (e.g., λ) while vector model parameters and sets of model parameters are denoted by boldfaced Greek letters (e.g., $\boldsymbol{\theta}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\upsilon}$).

2.2 Brief Introduction to Graphical Models

Since in this thesis we are dealing with multivariate probability distributions, it is useful to describe a general framework for specifying the structure of conditional independencies (and dependencies) between variables. We will use graphs with nodes corresponding to variables, and edges corresponding to multivariate dependencies, to represent the structure of the probability distributions. Such graphical models are often called belief networks. We will briefly define a few types of commonly used graphical models.

Let P be a distribution defined on an M-variate random variable $\boldsymbol{X} = (X_1, \ldots, X_M)$ where each X_i takes values on the domain $\mathcal{X}_i$, and $\boldsymbol{X}$ takes values on $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_M$. Let $\mathcal{V} = \{v_1, \ldots, v_M\}$ be a set of vertices or nodes with each v_i corresponding to random variable X_i. We will represent dependencies between the variables in $\boldsymbol{X}$ by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with edges $\mathcal{E}$ selected to capture *conditional independence* of the variables of $\mathcal{X}$. For disjoint $A, B, C \subseteq \mathcal{V}$, we will say that A and B are conditionally independent given C, denoted by $A \perp B \mid C$, if

$$P(\boldsymbol{x}_A, \boldsymbol{x}_B | \boldsymbol{x}_C) = P(\boldsymbol{x}_A | \boldsymbol{x}_C) P(\boldsymbol{x}_B | \boldsymbol{x}_C), \text{ whenever } P(\boldsymbol{x}_C) > 0.$$

Alternatively,

$$P(\boldsymbol{x}_A | \boldsymbol{x}_B, \boldsymbol{x}_C) = P(\boldsymbol{x}_A | \boldsymbol{x}_C), \text{ when } P(\boldsymbol{x}_B, \boldsymbol{x}_C) > 0.$$

2.2.1 Markov Networks

First, we briefly describe how to use undirected graphs to describe conditional independence relations. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with nodes $\mathcal{V}$ corresponding to variables of $\boldsymbol{X}$ and a set of undirected edges $\mathcal{E}$ defined on $\{\{u, v\} \in \mathcal{V} \times \mathcal{V} : u \neq v\}$. The conditional independence relations of P can be encoded using a separation property. For three disjoint subsets $A, B, C \subseteq \mathcal{V}$, C separates A from B if any path connecting any $a \in A$ and any $b \in B$ passes through a node from C. As in Pearl (1988), if $\forall A, B, C \subseteq \mathcal{V}$

$$A \perp B \mid C \Longrightarrow C \text{ separates } A \text{ from } B,$$

then $\mathcal{G}$ is a *dependency map* or D-map of P. Conversely, if separation in $\mathcal{G}$ implies the conditional independence,

$$A \perp B \mid C \Longleftarrow C \text{ separates } A \text{ from } B,$$

then $\mathcal{G}$ is an *independency map* or I-map of P. If both conditions hold, then $\mathcal{G}$ is a *perfect map* of P. Since we are interested in capturing the conditional independency relations for a distribution without introducing additional dependency assumptions, we require that undirected graphical models be I-maps of P. Such graph always exists as a graph with edges connecting all pairs of vertices, otherwise called a *complete graph*, is an I-map. We further restrict our graph to be a *minimal* I-map of P by requiring that a deletion of any $e \in \mathcal{E}$ would make G cease to be an I-map. Such $\mathcal{G}$ is called a *Markov network* of P (Pearl, 1988). Markov networks can be interpreted as the sparsest undirected graphs not introducing additional conditional independency assumptions.

We define the set of *neighbors* for a node $v \in \mathcal{V}$, denoted by $ne\,(v)$ to be the set of all vertices sharing an edge with v, i.e., $ne\,(v) = \{u \in \mathcal{V} : \{u, v\} \in \mathcal{E}\}$. Let the *boundary* of a subset $A \subseteq \mathcal{V}$, denoted by $bd\,(A)$ be the union of neighbors not in A of all vertices in A, i.e., $bd\,(A) = \{u \in \mathcal{V} \setminus A : \exists v \in A,\; u \in ne\,(v)\}$. Under a set of conditions, the probability of a subset of variables is conditionally independent from the rest of the variables given their boundary:

$$p\left(\boldsymbol{x}_A|\boldsymbol{x}_{\mathcal{V}\backslash A}\right) = P\left(\boldsymbol{x}_A|\boldsymbol{x}_{bd(A)}\right).$$

If $\mathcal{G}$ is an I-map of P, and if P is positive on $\mathcal{X}$, then P factorizes into a product of distributions on its cliques, complete subsets of $\mathcal{G}$. Let $\mathcal{C}(\mathcal{G}) = \{C \subseteq \mathcal{V} : C \text{ is a clique in } \mathcal{G}\}$ be the set of cliques. Then by the Hammersley-Clifford theorem (Hammersley and Clifford, 1971) $P(\boldsymbol{x})$ factorizes as

$$P(\boldsymbol{x}) \propto \prod_{C \in \mathcal{C}(\mathcal{G})} \psi(\boldsymbol{x}_C)$$

where $\psi(\boldsymbol{x}_C) > 0$ are functions, often called *potential functions* or just potentials, defined on the cliques.

2.2.2 Bayesian Networks

Alternatively, we can employ directed acyclic graphs (DAGs) to represent conditional independence relations. These DAGs are also often referred to as *Bayesian networks* in computer science. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a DAG with nodes $\mathcal{V}$ corresponding to variables of $\boldsymbol{X}$ and a set of directed edges $\mathcal{E}$ as a subset of $\{(u, v) \in \mathcal{V} \times \mathcal{V} : u \neq v\}$. As in the case for undirected graphs, we will define a separation criterion for subsets of $\mathcal{V}$; however, we need to take the directionality in account. If $A, B, C \subseteq \mathcal{V}$, C *d-separates* ("d" stands for directionally) A from B if along any path from every $a \in A$ to every $b \in B$ there is a node $u \in \mathcal{V}$ such that at least one of two conditions hold: (1) u has converging arrows and none of w's descendants are in C, or (2) $w \in C$ and w does not have converging arrows (Pearl, 1988). While the rules may appear a little confusing at first, in practice, d-separation on a graph can be checked by a few simple rules (Shachter, 1998). As with undirected graphs, a DAG $\mathcal{G}$ is an *I-map* of P, if $\forall A, B, C \subseteq \mathcal{V}$

$$A \perp B \mid C \Longleftarrow C \text{ d-separates } A \text{ from } B.$$

$\mathcal{G}$ is a minimal I-map if removal of any edge removes the I-map property (Pearl, 1988). Similar to Markov networks, minimal I-maps are the sparsest DAGs not introducing additional conditional independence assumptions.

Perhaps the most important property of Bayesian networks is the factorization of the joint probability distribution under $\mathcal{G}$. For a node $v \in \mathcal{V}$, let a set of *parents*, denoted by $pa\left(v\right)$, be the set of all nodes in $\mathcal{V}$ pointing to v, i.e., $pa\left(v\right) = \{u \in \mathcal{V} : (u, v) \in \mathcal{E}\}$. The joint probability $P\left(\boldsymbol{x}\right)$ factorizes as

$$P\left(\boldsymbol{x}\right) = \prod_{v \in \mathcal{V}} P\left(x_v | \boldsymbol{x}_{pa(v)}\right).$$

A Bayesian network can be converted into a Markov network by a process of *moralization*: first, for each node, all of its parents are connected by undirected edges; second, all directed edges are made undirected. Conversely, in order to convert a Markov network into a belief network, one needs to select an ordering on the nodes of $\mathcal{V}$, and then process the nodes according to the ordering, choosing as parents for each node a minimal set of nodes preceding it in the ordering, separating it from the rest of the preceding nodes.

2.2.3 Decomposable Models

In general, a distribution might not have a perfect representation as a Markov network or as a Bayesian network, i.e., I-maps do not capture all of conditional independence relations present under the distribution. However, the set of models with conditional independency that can be perfectly represented by both Markov and Bayesian networks have a special structure with a number of desirable properties. This class of models is called *decomposable*. It can be shown that a model is decomposable if and only if its Markov network is *chordal*, having no loops of length 4 or more without a chord. Such graph is also called decomposable. Perhaps the most important property of the decomposable models is that their properties can be captured by the set of maximal cliques. In a slight abuse of the notation, let $\mathcal{C}$ be the set of maximal cliques of decomposable graph $\mathcal{G}$. A *junction tree*[1] $\mathcal{J} = (\mathcal{C}, \mathcal{E}_J)$ is an undirected tree with maximal cliques of $\mathcal{G}$ as nodes and their pairs as edges selected to support the *running intersection property*, that every node of the junction tree on the path between $C_1 \in \mathcal{C}$ and $C_2 \in \mathcal{C}$ contains all nodes of $\mathcal{V}$ in the intersection $C_1 \cap C_2$. Then the joint probability distribution $P(\boldsymbol{x})$ can be decomposed as a product of the joint probabilities on the maximal cliques and *separators*, pairwise clique intersections corresponding to edges $\mathcal{E}_J$ of the junction tree:

$$P(\boldsymbol{x}) = \frac{\prod_{C \in \mathcal{C}} P(\boldsymbol{x}_C)}{\prod_{\{C_1, C_2\} \in \mathcal{E}_J} P(\boldsymbol{x}_{C_1 \cap C_2})} \propto \prod_{C \in \mathcal{C}} P(\boldsymbol{x}_C).$$

This decomposition can be used to speed up inference and learning algorithms for Markov networks, and we will exploit it whenever possible. However, such decomposition exists only for decomposable models as the junction tree exists if and only if the graph $\mathcal{G}$ is decomposable (or chordal) (Cowell et al., 1999), making decomposable

[1] Junction trees are sometimes called join trees or clique trees.

models a very important special class of models. As we will see later, because of the efficient inference and learning that exists for tree-structured models, junction trees are often constructed and used on models with non-decomposable graphs by first *triangulating* (adding chords to rid of chordless loops) the corresponding network.

2.2.4 Notes on Graphical Models

In this section, we only briefly covered the concepts essential to the rest of the thesis. A thorough treatment of graphical models can be found in a number of texts (e.g., Pearl, 1988; Cowell et al., 1999; Lauritzen, 1996). Markov and Bayesian networks are not the only ways to represent conditional independencies in probability distributions. Both Markov and belief networks can be generalized by *chain graphs*, the graphs allowing both directed and undirected edges, and the conditional independence relations can be extended to chain graphs as well (Cowell et al., 1999). Another alternative, *factor graphs* represent the probability distributions as a product of functions on subsets of variables (Kschischang et al., 2001).

Chapter 3

A Review of Homogeneous and Non-Homogeneous Hidden Markov Models

In this chapter, we describe properties of hidden Markov models (HMMs) and explain how we are going to use HMMs to model multivariate time series. Most of the material in this chapter is well-known in the literature (e.g., Rabiner, 1989; MacDonald and Zucchini, 1997; Bengio, 1999; Ghahramani, 2001) and is included here for completeness and to provide context for later chapters.

3.1 Model Description

Let $\boldsymbol{R}_t$ be an M-dimensional vector of measurements at time t. Let $\boldsymbol{R}_{1:T} = (\boldsymbol{R}_1, \ldots, \boldsymbol{R}_T)$ be a vector sequence of length T. Let $S_{1:T} = (S_1, \ldots, S_T)$ be the corresponding sequence of hidden (latent) states with each of the hidden states $S_t,\ t = 1, \ldots, T$ taking on one of K values. A hidden Markov model defines a joint distribution on $\boldsymbol{R}_{1:T}$ and $S_{1:T}$ using two conditional independence assumptions:

1. Individual vector observations $\boldsymbol{R}_t$ are conditionally independent of all other variables in the model given S_t, i.e.,

$$P\left(\boldsymbol{r}_t|\boldsymbol{r}_{1:t-1}, s_{1:t}\right) = P\left(\boldsymbol{r}_t|s_t\right). \tag{3.1}$$

2. The sequence of latent states $S_{1:T}$ has the k-th order stationary Markov property, i.e., the probability distribution for the current hidden state depends only on the values of the previous k hidden states:

$$P\left(s_t|s_{1:t-1}\right) = \begin{cases} P\left(s_t|s_{t-k:t-1}\right) & t > k, \\ P\left(s_t|s_{1:t-1}\right) & t \leq k. \end{cases}$$

   Unless otherwise stated, we will assume the first-order model ($k = 1$):

$$P\left(s_t|s_{1:t-1}\right) = \begin{cases} P\left(s_t|s_{t-1}\right) & t \geq 2, \\ P\left(s_1\right) & t = 1. \end{cases} \tag{3.2}$$

Let

$$\gamma_{ji} = P\left(S_t = i|S_{t-1} = j\right) \text{ and } \pi_i = P\left(S_1 = i\right).$$

We further assume stationarity (independence of time) of the probability distribution of vector $\boldsymbol{R}_t$ given the corresponding hidden state S_t:

$$P\left(\boldsymbol{R}_t = \boldsymbol{r}|S_t = i\right) = F_i\left(\boldsymbol{r}\right).$$

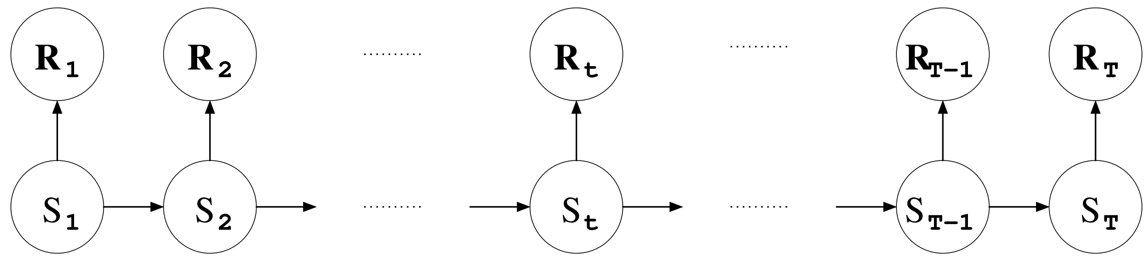

Figure 3.1: Bayesian network representation of a homogeneous HMM satisfying conditional independence assumptions (3.1) and (3.2). (Smyth et al., 1997)

The stationarity assumption for the probability $P(s_t|s_{t-1})$ defines a *homogeneous* HMM; lack of stationarity for $P(S_t|S_{t-1})$ defines a *non-homogeneous* HMM.

For a homogeneous HMM, let $\mathbf{\Pi} = (\pi_1, \ldots, \pi_K)$ be the *first-state probability vector*, and let $\mathbf{\Gamma} = (\gamma_{11}, \ldots, \gamma_{KK})$ be the *transition matrix*. Let $\mathbf{\Upsilon}$ defining all of $F_i(\boldsymbol{r}) = P(\boldsymbol{r}_t|S_t = i)$ be the set of *emission* parameters. The joint probability of the data and the hidden states can then be written as

$$P(\boldsymbol{r}_{1:T}, S_{1:T} = s_{1:T}|\mathbf{\Pi}, \mathbf{\Gamma}, \mathbf{\Upsilon}) = \left[\pi_{s_1} \prod_{t=2}^{T} \gamma_{s_{t-1}s_t}\right] \left[\prod_{t=1}^{T} F_{s_t}(\boldsymbol{r}_t)\right].$$

A Bayesian network representation of a homogeneous HMM is shown in Figure 3.1.

3.1.1 Auto-regressive Models

It is sometimes beneficial to represent temporal dependence in time series data not only via the transition probability (3.2), but also by using the emission distribution. This can be accomplished by relaxing the conditional independence assumption (3.1) to allow direct dependence of $\boldsymbol{X}_t$ on a short history of previous $\boldsymbol{X}_\tau$ for $\tau < t$. This *auto-regressive* model (sometimes called a k-th order auto-regressive model) (Poritz, 1982; Rabiner, 1989) replaces the first conditional independence assumption (3.1)

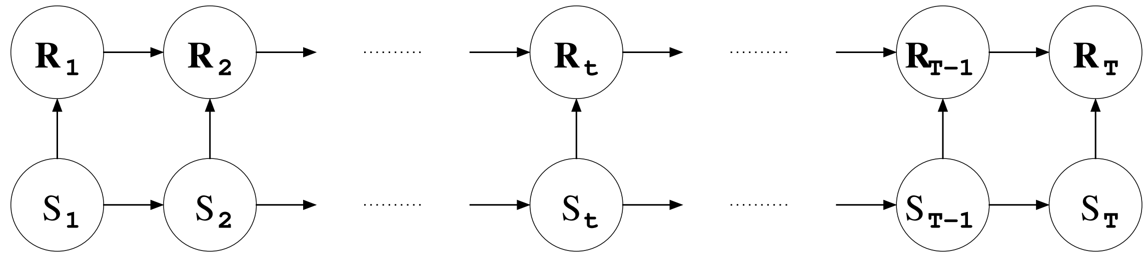

Figure 3.2: Bayesian network representation of an AR-HMM satisfying conditional independence assumptions (3.3) and (3.2).

replaced with:

1. Individual vector observations $\boldsymbol{R}_t$ are conditionally independent of all other variables in the model given S_t and the previous k observations, i.e.,

$$P\left(\boldsymbol{r}_t|\boldsymbol{r}_{1:t-1}, s_{1:t}\right) = \begin{cases} P\left(\boldsymbol{r}_t|s_t, \boldsymbol{r}_{t-k:t-1}\right) & t > k, \\ P\left(\boldsymbol{r}_t|s_t, \boldsymbol{r}_{1:t-1}\right) & t \leq k. \end{cases} \tag{3.3}$$

Unless otherwise specified, we consider only the first order ($k = 1$) auto-regressive models.

It is trivial to observe that the original conditional independence assumption for the emission distribution is a special case of the conditional independence assumption for an auto-regressive model. To consider both models at the same time in the future, we will denote $P\left(\boldsymbol{r}_t|S_t = i, \boldsymbol{r}_{t-1}\right)$ by $F_i\left(\boldsymbol{r}_t|\boldsymbol{r}_{t-1}\right)$ where for non-auto-regressive model, $F_i\left(\boldsymbol{r}_t|\boldsymbol{r}_{t-1}\right) = F_i\left(\boldsymbol{r}_t\right) = P\left(\boldsymbol{r}_t|S_t = i\right)$ for all values of $\boldsymbol{r}_{t-1}$. For auto-regressive models, $\boldsymbol{\Upsilon}$ contains parameters for $P\left(\boldsymbol{r}_t|s_t, \boldsymbol{r}_{t-1}\right)$, and the joint probability is expressed as

$$P\left(\boldsymbol{r}_{1:T}, S_{1:T} = s_{1:T}|\boldsymbol{\Pi}, \boldsymbol{\Gamma}, \boldsymbol{\Upsilon}\right) = \left[\pi_{s_1} \prod_{t=2}^{T} \gamma_{s_{t-1}s_t}\right] \left[\prod_{t=1}^{T} F_{s_t}\left(\boldsymbol{r}_t|\boldsymbol{r}_{t-1}\right)\right].$$

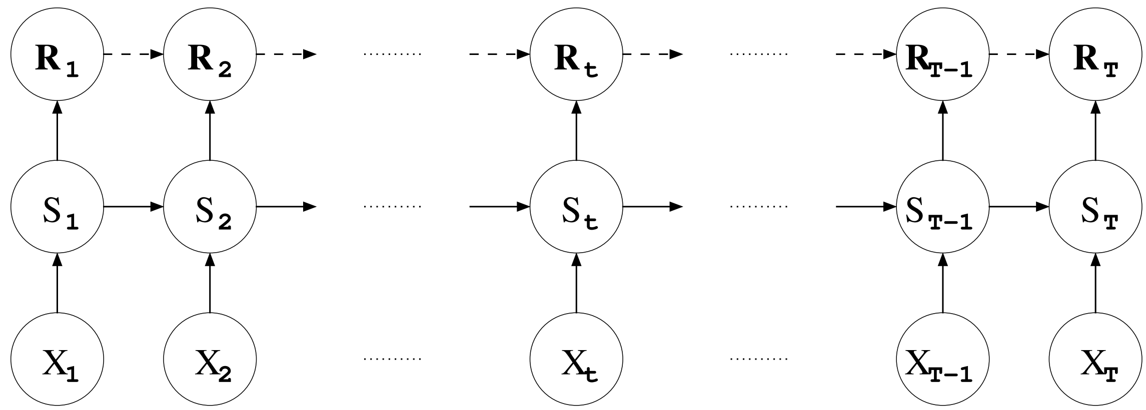

Figure 3.3: Bayesian network representation of a non-homogeneous HMM satisfying the conditional independence assumptions (3.1) (or, seen as dashed lines, (3.3) for AR model) and (3.4).

A graphical model for a first order auto-regressive HMM (AR-HMM) is presented in Figure 3.2. For discrete-valued $\boldsymbol{R}$, this model is sometimes called a *double-chain* Markov model (Berchtold, 1999).

3.1.2 Non-homogeneous HMMs

Sometimes the data $\boldsymbol{R}$ is known (or assumed) to be generated dependent on a set of observed time series of variables $\boldsymbol{X}$. For example, precipitation data can be influenced by the values of atmospheric variables like sea surface pressure, wind vector, and temperature. Homogeneous HMMs can be extended to allow the probability distribution of the output variables to be dependent on observed input variables. Here, we adopt the non-homogeneous HMM (NHMM) framework of Hughes and Guttorp (Hughes and Guttorp, 1994; Hughes et al., 1999). Assume that we have a sequence of D-dimensional input column vectors $\boldsymbol{X}_{1:T} = (\boldsymbol{X}_1, \ldots, \boldsymbol{X}_T)$. The presence of the input vectors replaces the (homogeneous) Markov assumption of HMMs with the assumption that the probability for a hidden state S_t also depends on the value of the corresponding input vector $\boldsymbol{X}_t$:

$$P\left(s_t|s_{1:t-1},\boldsymbol{x}_{1:t}\right)=\begin{cases}P\left(s_t|s_{t-1},\boldsymbol{x}_t\right) & t\geq 2,\\ P\left(s_1|\boldsymbol{x}_1\right) & t=1.\end{cases}\tag{3.4}$$

The corresponding graphical model is shown in Figure 3.3. The values of the first-state probability vector and the transition matrix are now functions of the input vector value:

$$\pi_i\left(\boldsymbol{x}\right)=P\left(S_1=i|\boldsymbol{X}_1=\boldsymbol{x}\right)\text{ and }\gamma_{ji}\left(\boldsymbol{x}\right)=P\left(S_t=i|S_{t-1}=j,\boldsymbol{X}_t=\boldsymbol{x}\right).$$

We employ polytomous (or multinomial) logistic regression to parametrize the hidden state transition:

$$\begin{aligned}P\left(S_t=i|S_{t-1}=j,\boldsymbol{X}_t=\boldsymbol{x}\right) &= \frac{\exp\left(\sigma_{ji}+\boldsymbol{\rho}_i^t\boldsymbol{x}\right)}{\sum_{k=1}^K\exp\left(\sigma_{jk}+\boldsymbol{\rho}_k^t\boldsymbol{x}\right)}\text{ for }t\geq 2,\\ P\left(S_1=i|\boldsymbol{X}_1=\boldsymbol{x}\right) &= \frac{\exp\left(\lambda_i+\boldsymbol{\rho}_i^t\boldsymbol{x}\right)}{\sum_{k=1}^K\exp\left(\lambda_k+\boldsymbol{\rho}_k^t\boldsymbol{x}\right)}\text{ for }t=1,\end{aligned}$$

where $\lambda_i,\sigma_{ji}\in\mathbb{R}$ and $\boldsymbol{\rho}_i\in\mathbb{R}^D$. If the parameters are not further restricted, the parameters are not uniquely defined. Let $\boldsymbol{\omega}_i=\left(\lambda_i,\sigma_{1i},\ldots,\sigma_{Ki},\left(\rho_i\right)_1,\ldots,\left(\rho_i\right)_D\right)\in\mathbb{R}^{K+D+1}$. Pick any $\boldsymbol{\omega}\in\mathbb{R}^{K+D+1}$ and replace each of $\boldsymbol{\omega}_i$ with $\boldsymbol{\omega}_i+\boldsymbol{\omega}$. The resulting set of parameters would yield the same probability distribution for all $\gamma_{ij}\left(\mathbf{x}\right)$ and $\pi_i\left(\boldsymbol{x}\right)$. To guarantee the uniqueness of the parameters, we set $\boldsymbol{\omega}_1=\mathbf{0}$.

Let $\boldsymbol{\Omega}=\left(\boldsymbol{\omega}_1,\ldots,\boldsymbol{\omega}_K\right)$ be the set of transition parameters, and let $\boldsymbol{\Theta}=\left(\boldsymbol{\Omega},\boldsymbol{\Upsilon}\right)$ be the set of all parameters of an NHMM. The joint probability of the data and the hidden states can then be expressed as

$$P\left(\boldsymbol{r}_{1:T}, s_{1:T} | \boldsymbol{x}_{1:T}, \boldsymbol{\Theta}\right) = \left[\pi_{s_1}\left(\boldsymbol{x}_1\right) \prod_{t=2}^{T} \gamma_{s_{t-1}s_t}\left(\boldsymbol{x}_t\right)\right] \left[\prod_{t=1}^{T} F_{s_t}\left(\boldsymbol{r}_t | \boldsymbol{r}_{t-1}\right)\right]. \tag{3.5}$$

Note that homogeneous HMMs are just a special case of NHMMs with

$$\pi_i = \frac{\exp\left(\lambda_i\right)}{\sum_{k=1}^{K} \exp\left(\lambda_k\right)} \text{ and } \gamma_{ji} = \frac{\exp\left(\sigma_{ji}\right)}{\sum_{k=1}^{K} \exp\left(\sigma_{jk}\right)} \quad i, j = 1, \ldots, K,$$

or, equivalently, $\boldsymbol{\rho}_i = \mathbf{0}, \ i = 1, \ldots, K$.

3.2 Hidden State Distributions

In this section, we assume that the set of parameters $\boldsymbol{\Theta}$ is known, and we omit dependence on $\boldsymbol{\Theta}$ in all equations in this section. Given these parameters, we derive the well-known equations and methods related to estimating the probabilities of the unobserved states $S_{1:T}$.

3.2.1 Inference

It is often desirable to calculate a probability distribution over the unobserved variables given the observed data. In other words, we want to estimate

$$P\left(s_{1:T} | \boldsymbol{r}_{1:T}, \boldsymbol{x}_{1:T}\right) = \frac{P\left(s_{1:T}, \boldsymbol{r}_{1:T} | \boldsymbol{x}_{1:T}\right)}{P\left(\boldsymbol{r}_{1:T} | \boldsymbol{x}_{1:T}\right)} = \frac{P\left(s_{1:T}, \boldsymbol{r}_{1:T} | \boldsymbol{x}_{1:T}\right)}{\sum_{S_{1:T}} P\left(s_{1:T}, \boldsymbol{r}_{1:T} | \boldsymbol{x}_{1:T}\right)}.$$

The numerator can be computed from Equation 3.5. The sum in the denominator, however, has K^T terms and cannot be evaluated directly. To compute the likelihood of the data $P\left(\boldsymbol{r}_{1:T} | \boldsymbol{x}_{1:T}\right)$, we can employ a recursive procedure called the *Forward-Backward* procedure (e.g., Rabiner, 1989). For each value of each hidden state S_t, we recursively calculate a summary of information preceding the state (α_t) and following the state (β_t) as follows:

$$\alpha_t(i) = P(S_t = i, \boldsymbol{r}_{1:t}|\boldsymbol{x}_{1:t}) \text{ and } \beta_t(i) = P(\boldsymbol{r}_{t+1:T}|S_t = i, \boldsymbol{r}_t, \boldsymbol{x}_{t+1:T}). \tag{3.6}$$

Then

$$\begin{aligned}
\alpha_1(i) &= P(S_1 = i|\boldsymbol{x}_1) P(\boldsymbol{r}_1|S_1 = i); \\
\alpha_{t+1}(i) &= P(\boldsymbol{r}_{t+1}|S_{t+1} = j, \boldsymbol{r}_t) \sum_{i=1}^{K} P(S_{t+1} = j|S_t = i, \boldsymbol{x}_{t+1}) \alpha_t(i); \\
\beta_T(i) &= 1; \\
\beta_t(i) &= \sum_{j=1}^{K} P(S_{t+1} = j|S_t = i, \boldsymbol{x}_{t+1}) P(\boldsymbol{r}_{t+1}|S_{t+1} = j, \boldsymbol{r}_t) \beta_{t+1}(j).
\end{aligned}$$

If the computation of $F_{s_t}(\boldsymbol{r}_t|\boldsymbol{r}_{t-1})$ has time complexity $O(R_{time})$ and storage requirements $O(R_{space})$, then the time complexity of computing α's and β's is $O(TK(K + R_{time}))$ which is linear in T and quadratic in K. Space complexity $O(TK + R_{space})$. Using the values of α and β we can then easily evaluate the expressions needed for inference, sampling, and, later, for learning. The likelihood of the data sequence $P(\boldsymbol{R}_{1:T}|\boldsymbol{X}_{1:T})$ can be computed as

$$P(\boldsymbol{r}_{1:T}|\boldsymbol{x}_{1:T}) = \sum_{i=1}^{K} P(S_T = i, \boldsymbol{r}_{1:T}|\boldsymbol{x}_{1:T}) = \sum_{i=1}^{K} \alpha_T(i).$$

3.2.2 Sampling

It is sometimes necessary to sample sequences of hidden states $S_{1:T}$ from the posterior distribution $P(s_{1:T}|\boldsymbol{r}_{1:T}, \boldsymbol{x}_{1:T})$ (e.g., for Bayesian learning of the HMM parameters). Instead of using a direct Gibbs sampler, we can use the Forward-Backward recursion to sample from $P\left(s_{1:T}|\boldsymbol{r}_{1:T},\boldsymbol{x}_{1:T}\right)$ more efficiently (Chib, 1996; Scott, 2002). First note that

$$P\left(s_{1:T}|\boldsymbol{r}_{1:T},\boldsymbol{x}_{1:T}\right)=\prod_{t=1}^{T}P\left(s_t|s_{t+1:T},\boldsymbol{r}_{1:T},\boldsymbol{x}_{1:T}\right).$$

This expansion suggests the following sampling strategy: for $t=T,T-1,\ldots,1$ select a sample i_t according to $P\left(s_t|S_{t+1:T}=i_{t+1:T},\boldsymbol{r}_{1:T},\boldsymbol{x}_{1:T}\right)$. The resulting sequence $(i_1,\ldots,i_T)$ would then be a sample from $P\left(s_{1:T}|\boldsymbol{r}_{1:T},\boldsymbol{x}_{1:T}\right)$. To compute $P\left(s_t|s_{t+1:T},\boldsymbol{r}_{1:T},\boldsymbol{x}_{1:T}\right)$, we first notice that

$$\begin{aligned}
&P\left(S_t=i,s_{t+1:T},\boldsymbol{r}_{1:T}|\boldsymbol{x}_{1:T}\right)= \\
&\quad= P\left(S_t=i,\boldsymbol{r}_{1:t}|\boldsymbol{x}_{1:t}\right)P\left(s_{t+1:T},\boldsymbol{r}_{t+1:T}|S_t=i,\boldsymbol{r}_t,\boldsymbol{x}_{t+1:T}\right) \\
&\quad= \alpha_t\left(i\right)\gamma_{is_{t+1}}\left(\boldsymbol{x}_{t+1}\right)F_{s_{t+1}}\left(\boldsymbol{r}_{t+1}|\boldsymbol{r}_t\right)P\left(s_{t+2:T},\boldsymbol{r}_{t+2:T}|s_{t+1},\boldsymbol{r}_{t+1},\boldsymbol{x}_{t+2:T}\right).
\end{aligned}\tag{3.7}$$

It follows from (3.7) that

$$\begin{aligned}
P\left(S_t=i|s_{t+1:T},\boldsymbol{r}_{1:T},\boldsymbol{x}_{1:T}\right) &= \frac{P\left(S_t=i,s_{t+1:T},\boldsymbol{r}_{1:T}|\boldsymbol{x}_{1:T}\right)}{\sum_{k=1}^{K}P\left(S_t=k,s_{t+1:T},\boldsymbol{r}_{1:T}|\boldsymbol{x}_{1:T}\right)} \\
&= \frac{\alpha_t\left(i\right)\gamma_{is_{t+1}}\left(\boldsymbol{x}_{t+1}\right)}{\sum_{k=1}^{K}\alpha_t\left(k\right)\gamma_{ks_{t+1}}\left(\boldsymbol{x}_{t+1}\right)}.
\end{aligned}$$

The time complexity of sampling is $O\left(TK\left(K+R_{time}\right)\right)$ to compute the values of α, plus an additional $O\left(TK\right)$ per each simulated sequence. The space complexity is $O\left(TK+R_{space}\right)$ to compute α's plus an additional $O\left(T+K\right)$ per each simulated sequence.

Algorithm VITERBI($\boldsymbol{r}_{1:T}, \boldsymbol{\Theta}$)
Inputs: Sequence $\boldsymbol{r}_{1:T}$ of T M-variate vectors and parameters $\boldsymbol{\Theta}$ of an (N)HMM

1. Compute $P(\boldsymbol{r}_t|S_t = i, \boldsymbol{r}_{t-1})$ for all $t = 1, \ldots, T$ and $i = 1, \ldots, K$.
2. For $t = 1, \ldots, T$
   - For $i = 1, \ldots, K$
     - Compute $m(t, i)$ as defined in (3.8, 3.9) and a value j of S_{t-1} such that
       $$j = \arg\max_{S_{t-1}} P(S_t = i|s_{t-1}, \boldsymbol{x}_t)\, m(t-1, s_{t-1}),$$
       i.e.,
       $$m(t, i) = P(S_t = i, S_{t-1} = j, s_{1:t-2}, \boldsymbol{r}_{1:t}|\boldsymbol{x}_{1:t}).$$
3. Find $i_T = \arg\max_i m(T, i)$.
4. Reconstruct the sequence of states $S_{1:T} = i_{1:T}$ such that for all $t = 1, \ldots, T-1$, $i_t = \arg\max_{S_t} P(S_{t+1} = i_{t+1}|S_t = i_t, \boldsymbol{x}_{t+1})\, m(t, i_t)$.

Output: Sequence $i_{1:T}$ of hidden states

Figure 3.4: Viterbi algorithm for finding most likely sequence of hidden states.

3.2.3 Most Likely Sequences

Now, we briefly address the problem of finding the most likely sequence of hidden states, i.e.,

$$(i_1, \ldots, i_T) = \arg\max_{S_{1:T}} P(s_{1:T}|\boldsymbol{r}_{1:T}, \boldsymbol{x}_{1:T}) = \arg\max_{S_{1:T}} P(s_{1:T}, \boldsymbol{r}_{1:T}|\boldsymbol{x}_{1:T}).$$

This problem can be solved using the Viterbi (1967) algorithm, a dynamic programming algorithm which finds the best subsequences $S_{1:t}$ for $t = 1, \ldots, T$. Let $m(t, i)$ be the maximum probability of a state sequence of length t ending in state i:

$$m(t, i) = \max_{S_{1:t-1}} P(S_t = i, s_{1:t-1}, \boldsymbol{r}_{1:t}|\boldsymbol{x}_{1:t}).$$

$m(t,i)$ for $t = 1, \ldots, T$ and $i = 1, \ldots, K$ can be computed efficiently capitalizing on the following recursive property:

$$m(1,i) = P(\boldsymbol{r}_1|S_1 = i) P(S_1 = i|\boldsymbol{x}_1), \quad (3.8)$$

$$m(t,i) = P(\boldsymbol{r}_t|S_t = i, \boldsymbol{r}_{t-1}) \max_j P(S_t = i|S_{t-1} = j, \boldsymbol{x}_t) m(t-1,j). \quad (3.9)$$

This leads to the algorithm described in Figure 3.4. The algorithm has time complexity of $O(TK(K + R_{time}))$ and space complexity of $O(TK + R_{space})$.

3.3 Learning

In this section, we review approaches for finding a set of NHMM parameters that best fit a data set.

3.3.1 EM Framework for NHMMs

As we have seen in Section 3.1, the set of NHMM parameters $\boldsymbol{\Theta}$ consists of transition parameters $\boldsymbol{\Omega}$ specifying $P(s_{1:T}|\boldsymbol{x}_{1:T})$ and emission parameters $\boldsymbol{\Upsilon}$ specifying $P(\boldsymbol{r}_{1:T}|s_{1:T})$.

We assume the data set consists of N sequences each of length T. Let $\boldsymbol{R}_{nt}$ denote the t-th data vector of sequence n, and let $\boldsymbol{X}_{nt}$ and S_{nt} be the corresponding vector of inputs and the hidden state (respectively) for the same index and sequence. $\boldsymbol{R}_{n,1:T}$, $S_{n,1:T}$, and $\boldsymbol{X}_{n,1:T}$ denote n-th sequences of data, hidden states, and input vectors, $1 \leq n \leq N$. We will also use a single symbol $\mathbf{R}$, $\mathbf{S}$, and $\mathbf{X}$ to denote all N sequences of data, hidden states, and input vectors. We assume that the observed sequences are conditionally independent given the model. We further assume that the data set is complete, i.e., all none of the entries of $\boldsymbol{R}$ and $\boldsymbol{X}$ are missing.

In the Bayesian framework, we are interested in estimating the posterior distribution of the model parameters given the data. To do so, we first assume a fixed model structure $\mathcal{M}$ (for example, for NHMMs $\mathcal{M}$ consists of the number of hidden states and the functional and structural form of $P(\boldsymbol{r}|s)$), and then we define a prior distribution $P(\mathbf{\Theta}|\mathcal{M})$ on the parameters given the model. By Bayes rule

$$P(\mathbf{\Theta}|\mathcal{M}, \mathbf{R}, \mathbf{X}) = \frac{P(\mathbf{\Theta}|\mathbf{X}, \mathcal{M})\, P(\mathbf{R}|\mathbf{X}, \mathbf{\Theta}, \mathcal{M})}{P(\mathbf{R}|\mathbf{X}, \mathcal{M})}.$$

We assume that the prior is independent of the values of the input variables.[1] In this section, we assume $\mathcal{M}$ is given[2], and we will drop it from further equations. The posterior distribution can then be written as

$$P(\mathbf{\Theta}|\mathbf{R}, \mathbf{X}) = \frac{P(\mathbf{\Theta})\, P(\mathbf{R}|\mathbf{X}, \mathbf{\Theta})}{P(\mathbf{R}|\mathbf{X})}.$$

Instead of looking at the full posterior, we will concentrate on finding its mode. This maximum *a posteriori* approach requires finding $\arg\max_{\mathbf{\Theta}} P(\mathbf{\Theta})\, P(\mathbf{R}|\mathbf{X}, \mathbf{\Theta})$. Unless indicated otherwise, we will assume uniform priors thus utilizing the maximum likelihood approach of finding $\arg\max_{\mathbf{\Theta}} P(\mathbf{R}|\mathbf{X}, \mathbf{\Theta})$.

[1] The prior certainly depends on the *type* of the input variables. For example, in precipitation modeling, having $\mathbf{X}$ as observed sea-surface temperature and having $\boldsymbol{X}$ as observed wind vectors would yield different priors. We assume $\mathcal{M}$ includes the type of $\mathbf{X}$.

[2] The Bayesian framework allows us to define a prior on $\mathcal{M}$ and to study $P(\mathcal{M}|\mathbf{R}, \mathbf{X})$.

Under the NHMM model, the log-likelihood $l(\mathbf{\Theta})$ of the observed data, given the inputs, is defined as:

$$\begin{aligned} l(\mathbf{\Theta}) &= \ln P(\mathbf{R}|\mathbf{X},\mathbf{\Theta}) = \ln P(\boldsymbol{r}_{1,1:T},\ldots,\boldsymbol{r}_{N,1:T}|\boldsymbol{x}_{1,1:T},\ldots,\boldsymbol{x}_{N,1:T},\mathbf{\Theta}) \\ &= \sum_{n=1}^{N}\ln\sum_{S_{n,1:T}}\pi_{s_{n1}}(\boldsymbol{x}_{n1}|\mathbf{\Theta})\prod_{t=2}^{T}\gamma_{s_{n,t-1}s_{nt}}(\boldsymbol{x}_{nt}|\mathbf{\Theta})\prod_{t=1}^{T}F_{s_{nt}}(\boldsymbol{r}_{nt}|\boldsymbol{r}_{n,t-1},\mathbf{\Theta}). \end{aligned} \tag{3.10}$$

We seek the value of the parameter vector $\mathbf{\Theta}$ that maximizes (3.10). Maximization of this likelihood cannot be performed analytically. However, Baum-Welch algorithm (Baum et al., 1970), a variant of the Expectation-Maximization (EM) procedure (Dempster et al., 1977), provides an iterative method of climbing the $l(\mathbf{\Theta})$ surface in space of values for $\mathbf{\Theta}$. Starting with an initial set of parameters $\mathbf{\Theta}^0$, we iteratively calculate new sets of parameters improving the log-likelihood of the data at each iteration. Once a convergence criterion is reached, the last set of parameters $\hat{\mathbf{\Theta}}$ is chosen as the solution. This process of initialization followed by iterative "uphill" movement until convergence is repeated for several random initializations of $\mathbf{\Theta}^0$ and the $\hat{\mathbf{\Theta}}$ that corresponds to the largest value of $l(\hat{\mathbf{\Theta}})$ is chosen as the maximum likelihood estimate.

Let $\mathbf{\Theta},\mathbf{\Theta}'$ be sets of parameters. The difference in log-likelihoods can be broken down as

$$\begin{aligned} l(\mathbf{\Theta}') - l(\mathbf{\Theta}) &= \ln\frac{P(\mathbf{R}|\mathbf{X},\mathbf{\Theta}')}{P(\mathbf{R}|\mathbf{X},\mathbf{\Theta})} = \sum_{\mathbf{S}} P(\mathbf{S}|\mathbf{R},\mathbf{X},\mathbf{\Theta})\ln\frac{P(\mathbf{R}|\mathbf{X},\mathbf{\Theta}')}{P(\mathbf{R}|\mathbf{X},\mathbf{\Theta})} \\ &= \sum_{\mathbf{S}} P(\mathbf{S}|\mathbf{R},\mathbf{X},\mathbf{\Theta})\ln\frac{P(\mathbf{R},\mathbf{S}|\mathbf{X},\mathbf{\Theta}')}{P(\mathbf{R},\mathbf{S}|\mathbf{X},\mathbf{\Theta})} && (3.11) \\ &\quad + \sum_{\mathbf{S}} P(\mathbf{S}|\mathbf{R},\mathbf{X},\mathbf{\Theta})\ln\frac{P(\mathbf{S}|\mathbf{R},\mathbf{X},\mathbf{\Theta})}{P(\mathbf{S}|\mathbf{R},\mathbf{X},\mathbf{\Theta}')}. && (3.12) \end{aligned}$$

Expression 3.12 is a relative entropy (or Kullback-Leibler divergence) between $P(\mathbf{S}|\mathbf{R}, \mathbf{X}, \mathbf{\Theta})$ and $P(\mathbf{S}|\mathbf{R}, \mathbf{X}, \mathbf{\Theta}')$. Given two distributions $p(x)$ and $q(x)$ on $x \in \mathcal{X}$, the KL-divergence is defined as

$$KL(p \parallel q) = \sum_x p(x) \ln \frac{p(x)}{q(x)}.$$

KL can be thought of as an asymmetric distance between the distributions since $KL(p \parallel q) \geq 0$ with $KL(p \parallel q) = 0$ if and only if $p \equiv q$, and in general, $KL(p \parallel q) \neq KL(q \parallel p)$. (For more information on the KL-divergence, refer to Cover and Thomas (1991)). Since (3.12) is non-negative,

$$\begin{aligned} l(\mathbf{\Theta}') - l(\mathbf{\Theta}) &\geq \sum_{\mathbf{S}} P(\mathbf{S}|\mathbf{R}, \mathbf{X}, \mathbf{\Theta}) \ln \frac{P(\mathbf{R}, \mathbf{S}|\mathbf{X}, \mathbf{\Theta}')}{P(\mathbf{R}, \mathbf{S}|\mathbf{X}, \mathbf{\Theta})} \\ &= \sum_{\mathbf{S}} P(\mathbf{S}|\mathbf{R}, \mathbf{X}, \mathbf{\Theta}) \ln P(\mathbf{R}, \mathbf{S}|\mathbf{X}, \mathbf{\Theta}') - \sum_{\mathbf{S}} P(\mathbf{S}|\mathbf{R}, \mathbf{X}, \mathbf{\Theta}) \ln P(\mathbf{R}, \mathbf{S}|\mathbf{X}, \mathbf{\Theta}). \end{aligned}$$

To update the parameters $\mathbf{\Theta}^{r+1}$ at iteration r, we maximize

$$\begin{aligned} Q\left(\mathbf{\Theta}^r, \mathbf{\Theta}^{r+1}\right) &= E_{P(\mathbf{S}|\mathbf{R}, \mathbf{X}, \mathbf{\Theta}^r)}\left[\ln P\left(\mathbf{S}, \mathbf{R}|\mathbf{X}, \mathbf{\Theta}^{r+1}\right)\right] = \\ &= \sum_{n=1}^{N} \sum_{S_{n,1:T}} P\left(s_{n,1:T}|\boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \mathbf{\Theta}^r\right) \ln P\left(\boldsymbol{s}_{n,1:T}, \boldsymbol{r}_{n,1:T}|\boldsymbol{x}_{n,1:T}, \mathbf{\Theta}^{r+1}\right). \end{aligned} \tag{3.13}$$

By (3.13), $l(\mathbf{\Theta}^{r+1}) - l(\mathbf{\Theta}^r) \geq Q(\mathbf{\Theta}^r, \mathbf{\Theta}^{r+1}) - Q(\mathbf{\Theta}^r, \mathbf{\Theta}^r)$, so by maximizing $Q(\mathbf{\Theta}^r, \mathbf{\Theta}^{r+1})$, we guarantee that the log-likelihood will not decrease.

$Q(\mathbf{\Theta}^r, \mathbf{\Theta}^{r+1})$ is maximized in two steps. In the first, the E-step, we calculate $P(s_{n,1:T}|\boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \mathbf{\Theta}^r)$. In the second, the M-step, we maximize $Q(\mathbf{\Theta}^r, \mathbf{\Theta}^{r+1})$ with respect to the parameters in $\mathbf{\Theta}^{r+1}$. It is clearly infeasible to compute and store probabilities of $N \times K^T$ possible sequences of hidden states $P(s_{n,1:T}|\boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \mathbf{\Theta}^r)$

as suggested in the E-step. It turns out that only a manageable set of $N \times T \times K$ probabilities $A_{nt}(i) = P(S_{nt} = i | \boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \boldsymbol{\Theta}^r)$ and $N \times (T-1) \times K^2$ probabilities $B_{nt}(i,j) = P(S_{nt} = i, S_{n,t-1} = j | \boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \boldsymbol{\Theta}^r)$ are needed to perform optimization in the M-step:

$$
\begin{aligned}
&Q\left(\boldsymbol{\Theta}^r, \boldsymbol{\Theta}^{r+1}\right) \\
&= \sum_{n=1}^{N} \sum_{S_{n,1:T}} P\left(s_{n,1:T} | \boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \boldsymbol{\Theta}^r\right) \ln P\left(s_{n,1:T}, \boldsymbol{r}_{n,1:T} | \boldsymbol{x}_{n,1:T}, \boldsymbol{\Theta}^{r+1}\right) \\
&= \sum_{n=1}^{N} \sum_{S_{n,1:T}} P\left(s_{n,1:T} | \boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \boldsymbol{\Theta}^r\right) \times \left(\sum_{t=1}^{T} \ln P\left(\boldsymbol{r}_{nt} | s_{nt}, \boldsymbol{r}_{n,t-1}, \boldsymbol{\Theta}^{r+1}\right) \right. \\
&\qquad \left. + \ln P\left(s_{n1} | \boldsymbol{x}_{n1}, \boldsymbol{\Theta}^{r+1}\right) + \sum_{t=2}^{T} \ln P\left(s_{nt} | s_{n,t-1}, \boldsymbol{x}_{nt}, \boldsymbol{\Theta}^{r+1}\right) \right) \\
&= \sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{i=1}^{K} A_{nt}(i) \ln P\left(\boldsymbol{r}_{nt} | S_{nt} = i, \boldsymbol{r}_{n,t-1}, \boldsymbol{\Theta}^{r+1}\right) \qquad (3.14) \\
&\quad + \sum_{n=1}^{N} \sum_{i=1}^{K} A_{n1}(i) \ln P\left(S_{n1} = i | \boldsymbol{x}_{n1}, \boldsymbol{\Theta}^{r+1}\right) \qquad (3.15) \\
&\quad + \sum_{n=1}^{N} \sum_{t=2}^{T} \sum_{i=1}^{K} \sum_{j=1}^{K} B_{nt}(i,j) \ln P\left(S_{nt} = i | S_{n,t-1} = j, \boldsymbol{x}_{nt}, \boldsymbol{\Theta}^{r+1}\right). \qquad (3.16)
\end{aligned}
$$

The quantities A_{nt} and B_{nt} can be calculated using values of α and β (3.6) for sequence n as computed by the recursive Forward-Backward procedure:

$$
A_{nt}(i) = \frac{\alpha_{nt}(i)\, \beta_{nt}(i)}{\sum_{k=1}^{K} \alpha_{nT}(k)} \text{ and } B_{nt}(i,j) = \frac{F_i(\boldsymbol{r}_{nt} | \boldsymbol{r}_{n,t-1})\, \gamma_{ji}(\boldsymbol{x}_{nt})\, \alpha_{n,t-1}(j)\, \beta_{nt}(i)}{\sum_{k=1}^{K} \alpha_{nT}(k)}.
$$

For the M-step, we find a set of parameters $\boldsymbol{\Theta}^{r+1}$ maximizing $Q(\boldsymbol{\Theta}^r, \boldsymbol{\Theta}^{r+1})$ (with added Lagrangians to adjust for constraints). It helps to notice first that the Expression 3.14 (we will denote it $Q_R(\boldsymbol{\Upsilon}^{r+1})$) and expressions 3.15 and 3.16 (we will denote their sum $Q_S(\boldsymbol{\Omega}^{r+1})$) can be optimized independently of each other. We discuss the optimization of Q_R with respect to various emission distributions with parameters $\boldsymbol{\Upsilon}^{r+1}$ in Section 3.4. For the homogeneous case, the updated parameters can be computed in closed form by solving a system of equations obtained by setting to zero partial derivatives of $Q_S\left(\boldsymbol{\Omega}^{r+1}\right)$ (with added Lagrangians to adjust for constraints) with respect to parameters of $\boldsymbol{\Gamma}$ and $\boldsymbol{\Pi}$:

$$\pi_i^{(r+1)} = \frac{\sum_{n=1}^{N} A_{n1}(i)}{N} \text{ and } \gamma_{ji}^{(r+1)} = \frac{\sum_{n=1}^{N}\sum_{t=2}^{T} B_{nt}(i,j)}{\sum_{n=1}^{N}\sum_{t=1}^{T-1} A_{nt}(j)}.$$

Unfortunately, in the non-homogeneous case, the parameters of the transition $P\left(S_t|S_{t-1},\boldsymbol{X}_t\right)$ have non-linear partial derivatives, and the system of equations resulting from equating their partial derivatives to zero cannot be solved analytically. We use a conjugate gradient algorithm to find a set of parameters $\boldsymbol{\Omega}^{r+1}$ making $Q_S\left(\boldsymbol{\Omega}^{r+1}\right) \geq 0$. The details are provided in Appendix A.1.

3.4 Modeling Emission Distributions

As was seen earlier in the chapter, the M-step for finding the parameters in an HMM can be split into maximization with respect to the transition parameters and maximization with respect to emission parameters, i.e., the optimization of

$$Q_R\left(\boldsymbol{\Upsilon}^{r+1}\right) = \sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{i=1}^{K} A_{nt}(i)\log P\left(\boldsymbol{r}_{nt}|S_{nt}=i,\boldsymbol{r}_{n,t-1},\boldsymbol{\Upsilon}^{r+1}\right) \tag{3.17}$$

where $A_{nt}(i) = P\left(S_{nt}=i|\boldsymbol{r}_{n,1:T},\boldsymbol{x}_{n,1:T},\boldsymbol{\Theta}^r\right)$. An intuitive way to interpret this optimization problem is to view it as approximating K probability distributions with weighted or fractional data samples where for a distribution i a data point $\mathbf{R}_{nt}$ has weight $A_{nt}(i)$. The specifics of the optimization depend on the forms of distribution for each of the hidden states $P\left(\boldsymbol{r}|S=i,\boldsymbol{r}_{previous},\boldsymbol{\Upsilon}\right)$, $i=1,\ldots,K$.

This begs the question: what should the structural form of each distribution $P(\boldsymbol{r}|S=i,\boldsymbol{r}_{previous},\boldsymbol{\Upsilon})$ be? How should we determine the conditional independence structure of each distribution? The application domain should obviously factor in the determination. However, the number of free parameters should also play a major role, especially considering that each of the mixture components is built only on a fraction of the data (based on the fractional counts corresponding to each component). This is especially important for categorical data since we cannot make a smoothness assumption about the functional form of the probability distribution (of the sort that is usually made about the probability density function in a real-valued data case). Therefore, we are interested in learning parsimonious probability distributions with sparse dependency structure. Since the structure of the probability distribution may vary for different domains, we can either (a) have it determined by the experts, (b) infer it from the data, or (c) use both the data and the expert knowledge. We will discuss several models with predetermined structure; however, the emphasis of this thesis is on the automatic discovery of underlying dependency structure.

Let $\boldsymbol{\phi}_i$ denote the set of parameters specifying the conditional independence *structure* for the emission distribution for state i, and let $\boldsymbol{\upsilon}_i$ denote the parameters needed to define the emission distribution in state i given the structure $\boldsymbol{\phi}_i$. The probability of a vector observation $\boldsymbol{R}$ given the state is then

$$P(\boldsymbol{r}|S=i,\boldsymbol{\Upsilon})=P(\boldsymbol{r}|\boldsymbol{\phi}_i,\boldsymbol{\upsilon}_i)\,.$$

The algebraic expression for the objective function Q_R in Equation 3.17 can be simplified further depending on the independence of structure and parameters for different states. Commonly, the emission parameters ν_i for different states i are assumed to be independent of each other, and in this thesis we make this assumption as well. (Kirshner et al. (2003) describe an example of a mixture model with shared parameters.) We can further assume that the structures $\boldsymbol{\phi}_i$ of emission distributions for different states i are also independent. Then the objective function Q_R can be rewritten as

$$
\begin{aligned}
Q_R\left(\boldsymbol{\Upsilon}^{r+1}\right) = \sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{i=1}^{K} A_{nt}(i)\log P\left(\boldsymbol{r}_{nt}|\boldsymbol{\phi}_i^{r+1},\boldsymbol{\upsilon}_i^{r+1}\right) & \\
= \sum_{i=1}^{K}\left(\sum_{\nu=1}^{N}\sum_{\tau=1}^{T} A_{\nu\tau}(i)\right)\sum_{n=1}^{N}\sum_{t=1}^{T}\frac{A_{nt}(i)}{\sum_{\nu=1}^{N}\sum_{\tau=1}^{T} A_{\nu\tau}(i)}\log P\left(\boldsymbol{r}_{nt}|\boldsymbol{\phi}_i^{r+1},\boldsymbol{\upsilon}_i^{r+1}\right) & \\
= \sum_{i=1}^{K} C_i \sum_{n=1}^{N}\sum_{t=1}^{T} W_{nt}(i)\log P\left(\boldsymbol{r}_{nt}|\boldsymbol{\phi}_i^{r+1},\boldsymbol{\upsilon}_i^{r+1}\right) & \qquad (3.18)
\end{aligned}
$$

where $C_i = \sum_{n=1}^{N}\sum_{t=1}^{T} A_{nt}(i)$ and $W_{nt}(i) = \frac{A_{nt}(i)}{C_i}$. Note that

$$
\sum_{n=1}^{N}\sum_{t=1}^{T} W_{nt}(i) = \sum_{n=1}^{N}\sum_{t=1}^{T}\frac{A_{nt}(i)}{C_i} = \sum_{n=1}^{N}\sum_{t=1}^{T}\frac{A_{nt}(i)}{\sum_{\nu=1}^{N}\sum_{\tau=1}^{T} A_{\nu\tau}(i)} = 1.
$$

By setting

$$
P_i(\boldsymbol{r}) = \sum_{\substack{n=1,t=1 \\ \boldsymbol{R}_{nt}=\boldsymbol{r}}}^{N,T} W_{nt}(i), \qquad (3.19)
$$

we define a probability distribution $P_i(\boldsymbol{r})$ on $\mathcal{R}$. The objective function can then be rewritten as

$$
\begin{aligned}
Q_R\left(\boldsymbol{\Upsilon}^{r+1}\right) &= \sum_{i=1}^{K} C_i \sum_{n=1}^{N}\sum_{t=1}^{T} W_{nt}(i)\log P\left(\boldsymbol{r}_{nt}|\boldsymbol{\phi}_i^{r+1},\boldsymbol{\upsilon}_i^{r+1}\right) \\
&= \sum_{i=1}^{K} C_i \sum_{\boldsymbol{r}\in\mathcal{R}} P_i(\boldsymbol{r})\log P\left(\boldsymbol{r}|\boldsymbol{\phi}_i^{r+1},\boldsymbol{\upsilon}_i^{r+1}\right),
\end{aligned}
$$

and can be optimized by independently finding $\boldsymbol{\phi}_i^{r+1}$ and $\boldsymbol{v}_i^{r+1}$ to maximize $\sum_{\boldsymbol{r}\in\mathcal{R}} P_i(\boldsymbol{r}) \log P\left(\boldsymbol{r}|\boldsymbol{\phi}_i^{r+1}, \boldsymbol{v}_i^{r+1}\right)$. This is equivalent to minimizing $KL\left(P_i \parallel P\left(\cdot|\boldsymbol{\phi}_i^{r+1}, \boldsymbol{v}_i^{r+1}\right)\right)$ independently for each $i = 1, \ldots, K$. Both of these optimization problems are usually solved by finding zeros of partial derivatives with respect to the parameters (while checking that the resulting value of the objective function is a maximum). Chapters 4 and 5 deal with various categorical and real-valued distributions which can be used to model $P(\boldsymbol{r}|S = i, \boldsymbol{v}_i)$.

If the distributions in each hidden state share an unknown conditional independence structure $\boldsymbol{\phi}$, Q_R in the form (3.18) cannot be optimized independently for different i. (Examples of such models can be found in Meilă and Jordan (2000) and Bilmes (1999).) However, note that

$$\sum_{i=1}^{K} C_i = \sum_{i=1}^{K}\sum_{n=1}^{N}\sum_{t=1}^{T} A_{nt}(i) = \sum_{n=1}^{N}\sum_{t=1}^{T}\left(\sum_{i=1}^{K} P\left(S_{nt} = i|\boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \boldsymbol{\Theta}^r\right)\right) = NT.$$

We can recast Q_R as

$$\begin{aligned} Q_R\left(\boldsymbol{\Upsilon}^{r+1}\right) &= \sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{i=1}^{K} A_{nt}(i) \log P\left(\boldsymbol{r}_{nt}|\boldsymbol{\phi}^{r+1}, \boldsymbol{v}_i^{r+1}\right) \\ &= \sum_{i=1}^{K} C_i \sum_{n=1}^{N}\sum_{t=1}^{T} W_{nt}(i) \log P\left(\boldsymbol{r}_{nt}|\boldsymbol{\phi}^{r+1}, \boldsymbol{v}_i^{r+1}\right) \\ &= NT \sum_{i=1}^{K} P_S(i) \sum_{\boldsymbol{r}\in\mathcal{R}} P_i(\boldsymbol{r}) \log P\left(\boldsymbol{r}|\boldsymbol{\phi}^{r+1}, \boldsymbol{v}_i^{r+1}\right) \end{aligned}$$

with $P_S(i) = \frac{C_i}{NT}$ defining a distribution over hidden states $i = 1, \ldots, K$. Thus maximizing $Q_R\left(\boldsymbol{\Upsilon}^{r+1}\right)$ is equivalent to minimizing the conditional relative entropy

$KL_{\boldsymbol{\Upsilon}^{r+1}|S}(P_i|P)$:

$$KL_{\boldsymbol{\Upsilon}^{r+1}|S}\left(P_i \parallel P\left(\cdot|\boldsymbol{\Upsilon}^{r+1}\right)\right) = \sum_{i=1}^{K} P_S(i) \sum_{\boldsymbol{r}\in\mathcal{R}} P_i(\boldsymbol{r}) \log \frac{P_i(\boldsymbol{r})}{P\left(\boldsymbol{r}|\boldsymbol{\phi}^{r+1}, \boldsymbol{\upsilon}_i^{r+1}\right)}.$$

The conditional relative entropy has the same property as the relative entropy (also called Kullback-Leibler divergence):

$$KL\left(p(x|y) \parallel q(x|y)\right) = \sum_y p(y) \sum_x p(x|y) \log \frac{p(x|y)}{q(x|y)} \geq 0$$

with $KL\left(p(x|y) \parallel q(x|y)\right) = 0$ iff $p(x|y) \equiv q(x|y)$.

3.5 Historical Remarks and Connections to Other Models

Hidden Markov models were known and used for speech processing as early as 1960's (Baum and Petrie, 1966). While initially mostly used in speech community, HMMs and their variations were also found to be useful in other applications, e.g., biology and bioinformatics (e.g., Krogh et al., 1994; Baldi and Brunak, 2001), economics (e.g., Bhar and Hamori, 2004), vision task recognition (e.g., Brand et al., 1997), music segmentation (e.g., Raphael, 1999), freeway traffic modelling (e.g., Kwon and Murphy, 2000), and the application considered in this thesis, multi-site precipitation occurrence modeling (e.g., Hughes and Guttorp, 1994; Hughes et al., 1999). Non-homogeneous HMMs were first introduced by Hughes and Guttorp (1994) to model multi-station rainfall occurrences for Washington state. Independently, and at roughly the same time, Bengio and Frasconi (1995) proposed an Input/Output HMM (IOHMM) framework allowing not only non-stationary transitions between hidden states, but also modeling output vectors dependent on both hidden states and input variables. Meilă and Jordan (1996) also proposed a type of NHMM as an extension of the mixture of experts.

There are several well-studied models related to NHMMs. If we make the transition matrix of a non-AR model to have all rows identical, we arrive at a (non-homogeneous) mixture model (or a mixture of experts (Jordan and Jacobs, 1994)) without the dependence on time t:

$$P\left(S_t = i|\boldsymbol{x}_t\right) = \pi_i\left(\boldsymbol{x}_t\right) \text{ and } P\left(\boldsymbol{r}_t|\boldsymbol{x}_t\right) = \sum_{i=1}^{K} \pi_i\left(\boldsymbol{x}_t\right) F_i\left(\boldsymbol{r}_t\right) \quad \forall t = 1, \ldots, T.$$

By considering a degenerate case of $K = 1$ for an AR-HMM, we obtain a first-order auto-regressive model of the vector series $\boldsymbol{R}_{1:T}$:

$$P\left(\boldsymbol{r}_{1:T}\right) = P\left(\boldsymbol{r}_{1:T}\right) \prod_{t=2}^{T} P\left(\boldsymbol{r}_t|\boldsymbol{r}_{t-1}\right).$$

NHMMs can also be viewed as both a special case of a more general class of temporal models called *dynamic Bayesian networks* (DBNs) (e.g., Murphy, 2002). DBNs define a probability distribution over a time-dependent set of variables. Just as with NHMMs, the variables or vectors can be broken into three types: $\boldsymbol{R}$, observed data (or output) vectors or sets of variables, $\boldsymbol{S}$, vector or a set of latent or hidden variables, and $\boldsymbol{X}$, a set of input data vectors. The joint distribution is based on one conditional independence assumption, Markov dependency of the consecutive time slices:

$$P\left(\boldsymbol{r}_t, \boldsymbol{s}_t|\boldsymbol{r}_{1:t-1}, \boldsymbol{s}_{1:t-1}, \boldsymbol{x}_{1:T}\right) = \begin{cases} P\left(\boldsymbol{r}_t, \boldsymbol{s}_t|\boldsymbol{r}_{t-1}, \boldsymbol{s}_{t-1}, \boldsymbol{x}_t\right) & t > 1 \\ P\left(\boldsymbol{r}_1, \boldsymbol{s}_1|\boldsymbol{x}_1\right) & t = 1. \end{cases} \tag{3.20}$$

with conditional independencies of probability distributions in both cases of (3.20) represented by Bayesian networks. Our formulation of NHMMs also allows $P(\boldsymbol{r}_t, s_t|\boldsymbol{x}_t)$ to have a graphical model representation dependent on the value of S_t, something standard DBNs do not allow. This can be corrected by representing $P(\boldsymbol{r}_t, s_t|\boldsymbol{x}_t)$ by a *Bayesian multinet* (Geiger and Heckerman, 1996), a set of Bayesian nets for different values of some of the variables. (When the structure of the Bayesian nets differs according to the value of a latent variable (the scenario we consider in the previous section), the problem is usually treated as a mixture of Bayesian networks or DAGs (Thiesson et al., 1999).) By combining temporal Bayesian structure of DBNs with flexibility of multinets, we obtain *dynamic Bayesian multinets* (Bilmes, 2000).

3.6 Summary

We have defined the NHMM in this chapter and showed how to perform inference and to learn the parameters of such a model. All of the results in this chapter are known in the literature with the exception of conjugate gradient method for learning NHMMs. In the next two chapters we will focus on modeling multivariate distributions for the emission distribution component of HMMs and NHMMs.

Chapter 4

Parsimonious Models for Multivariate Distributions for Categorical Data

In this chapter, we discuss modeling joint distributions for vectors of categorical (finite-valued) data. These models vary in the complexity of their dependence structures, and we are particularly interested in using them as emission distributions with HMMs. In the remainder of this chapter, however, we will focus on multivariate structure and not explicitly refer to the HMM context (models discussed here will be integrated with HMMs in Chapter 6). Our contributions in this chapter relate to learning multivariate joint and conditional distributions without a predefined dependence structure. For these models, we will learn both the parameters and the dependence structure.

As in Chapter 2.2, suppose that a multivariate distribution $P(\boldsymbol{x})$ is defined on domain $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_M$.[1] By finite-valued, nominal-valued, or categorical we mean a distribution defined on a finite-valued domain, i.e., $|\mathcal{X}| < \infty$. We denote the number of domain values for each of the variables by $B_m = |\mathcal{X}_m| < \infty$, $m = 1, \ldots, M$. We further assume that $\mathcal{X}_m = \{0, 1, \ldots, B_m - 1\}$. For simplicity, we assume that all of the variables have the same cardinality $B = B_1 = \cdots = B_M$. All of the results in this chapter can be trivially extended to domains of unequal cardinality.

We will also consider modeling *conditional* probability distributions $P(\boldsymbol{x}|\boldsymbol{y})$ where $\boldsymbol{Y}$ is a M_y-dimensional finite valued vector random variable with domain $\mathcal{Y} = \mathcal{Y}_1 \times \ldots \mathcal{Y}_{M_y}$. Let $\mathcal{V}_y$ be a set of vertices corresponding to variables of $\boldsymbol{Y}$ with $|\mathcal{V}_y| = M_y$. We further assume that all of variables of $\boldsymbol{Y}$ have the same domain cardinality B as variables of $\boldsymbol{X}$. While some of the conditional distributions discussed in this section can be applied to general finite valued $\boldsymbol{Y}$, we will pay special attention to distributions where the conditioning variable $\boldsymbol{Y}$ is defined on the same domain as $\boldsymbol{X}$, i.e., $\mathcal{X} = \mathcal{Y}$. We intend to use these distributions to model auto-regressive emission distributions with HMMs.

Unless otherwise constrained, an M-variate joint probability distribution with each variable taking on B values requires $B^M - 1$ free parameters to specify it. This number of parameters is intractable to work with even for relatively small values of B and M due to, among other reasons, insufficient data to learn the parameters of the distribution accurately. One way to limit the number of free parameters is to restrict the dependency structure of the distribution. As a result, we may not be able to capture the multivariate dependencies involving large number of variables. However, we are often interested primarily in capturing local or low-order (e.g., second-order or

[1]The notation is overloaded as random variable $\boldsymbol{X}$ is not the same as input vector in Chapter 3

pairwise) dependencies. By considering distributions expressible in the form of low-order distributions, we can drastically reduce the dimensionality of the parameter space. A distribution modeling only l-variable blocks of the M-variate distribution will require $O\left(\binom{M}{l} B^l\right)$ free parameters, manageable for small values of l, as compared to $O\left(B^M\right)$ free parameters required by an unconstrained distribution. Moreover, we are often interested in even sparser structures, i.e., where only a small number of l-variable interactions would be modeled directly, with the rest of the interactions either modeled indirectly (through interactions with other variables) or by lower order blocks of variables. For example, a model encoding only univariate and bivariate variable interactions for M-variate binary data will require only $M\left(M+1\right)/2$ free parameters[2] as compared to $2^M - 1$ free parameters needed to define an arbitrary distribution on an M-variate binary vector. For the case of conditional distributions, the number of free parameters is even larger since interactions of variables of $\boldsymbol{X}$ are also dependent on variables $\boldsymbol{Y}$. Sometimes the number of free parameters can be reduced by adding unobserved variables (e.g., mixture models); however, this significantly complicates learning of the structure, and sometimes, of the parameters.

Abstractly, in a maximum likelihood context, we can consider the problem of approximating a target distribution $P\left(\boldsymbol{x}\right)$, where $P\left(\boldsymbol{x}\right)$ is usually not defined in a parametric form but rather empirically, as a set $\mathcal{D} = \left\{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\right\}$ of M-dimensional vectors, possibly with a positive weight assigned to each vector. For example, in the context of HMMs, the parameter update for emission distribution for state i has target distribution $P_i\left(\boldsymbol{r}\right)$ composed of data points $\mathbf{R}$ with corresponding weights $\boldsymbol{A}\left(i\right) = \left\{A_{nt}\left(i\right)\right\}_{n=1,\ldots,N,\ t=1,\ldots,T}$. For simplicity, we assume everywhere in this chapter that $\mathcal{D}$ is complete, meaning that all M components are observed for all N vectors

[2]We will see an example of such distribution in Section 4.3.

in the set. While we discuss some of more general models, we concentrate on the models directly capturing only the first- and second-order (univariate and bivariate) interactions ($l = 2$). First we consider the simplest model assuming independence of variables ($l = 1$) and a simple extension of it to a conditional model (Section 4.1). The independence model has low time complexity for all operations, and thus is very practical. After that, we consider second order models of intermediate time complexities. Among them, we consider tree-structured models in Section 4.2 (including a novel model for conditional distributions, Section 4.2.1) which can be learned with high efficiency. Then, in Section 4.3 we consider more general exponential models based on the maximum entropy principle allowing a broad class of conditional independencies. We also introduce a new factored probability model with factors expressed in the form of conditional maximum entropy models (Section 4.3.2) and describe algorithms for estimating both the parameters and the structure. Finally, a few alternative models are considered in Section 4.4.

4.1 Independence and Conditional Independence Models

The independence model[3] decomposes the joint probability distribution on variables $\boldsymbol{X} = (X_1, \ldots, X_M)$ into the product of marginal probabilities of individual variables:

$$P_I(\boldsymbol{x}) = \prod_{m=1}^{M} P_I(x_m).$$

[3]Note: implicit in our discussion is the fact that we are building a model for each state in the HMM. So what is called an "independence model" here is in reality a conditional independence model given states.

In order to specify the distribution under the independence model, we need to specify the parameters for each of $P(x_m)$. Since each of X_m takes on B values, let $p_{mb} = P_I(X_m = b)\,,\ b = 0, \ldots, B-1$. For each $m = 1, \ldots, M$, $\sum_{b=0}^{B-1} p_{mb} = 1$, so we need $B-1$ free parameters for each $P_I(x_m)$, and $M(B-1)$ free parameters total.

Next, we address the parameter estimation. Let

$$\boldsymbol{\upsilon} = \{p_{mb}\}_{m=1,\ldots,M,\ b=0,\ldots,B-1}\,.$$

Given a distribution $P(\boldsymbol{x})$ defined on $\mathcal{R}$, we want to find $\boldsymbol{\upsilon}$ to minimize $KL(P \parallel P_I(\cdot|\boldsymbol{\upsilon}))$:

$$\begin{aligned}\arg\min_{\boldsymbol{\upsilon}} KL(P \parallel P_I(\cdot|\boldsymbol{\upsilon})) &= \arg\min_{\boldsymbol{\upsilon}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}) \log \frac{P(\boldsymbol{x})}{\prod_{m=1}^{M} P_I(x_m|\boldsymbol{\upsilon})} = \\ &= \sum_{m=1}^{M} \arg \min_{p_{m0},\ldots,p_{m,B-1}} KL\left(P(x_m) \parallel P_I(x_m|p_{m0}, \ldots, p_{m,B-1})\right),\end{aligned}$$

so the parameters for $p_{m0}, \ldots, p_{m,B-1}$ can be estimated independently of other m. The minimum of KL is reached when the distributions are identical, i.e.,

$$p_{mb} = P(X_m = b) \quad \forall m = 1, \ldots, M,\ b = 0, \ldots, B-1. \tag{4.1}$$

4.1.1 Auto-regressive Conditional Independence

The independence model can be trivially extended to a conditional model given $\boldsymbol{Y}$ by assuming that individual variables X_m are conditionally independent given $\boldsymbol{Y}$ (Figure 4.1, left):

$$P_{CI}(\boldsymbol{x}|\boldsymbol{y}) = \prod_{m=1}^{M} P_{CI}(x_m|\boldsymbol{y})\,.$$

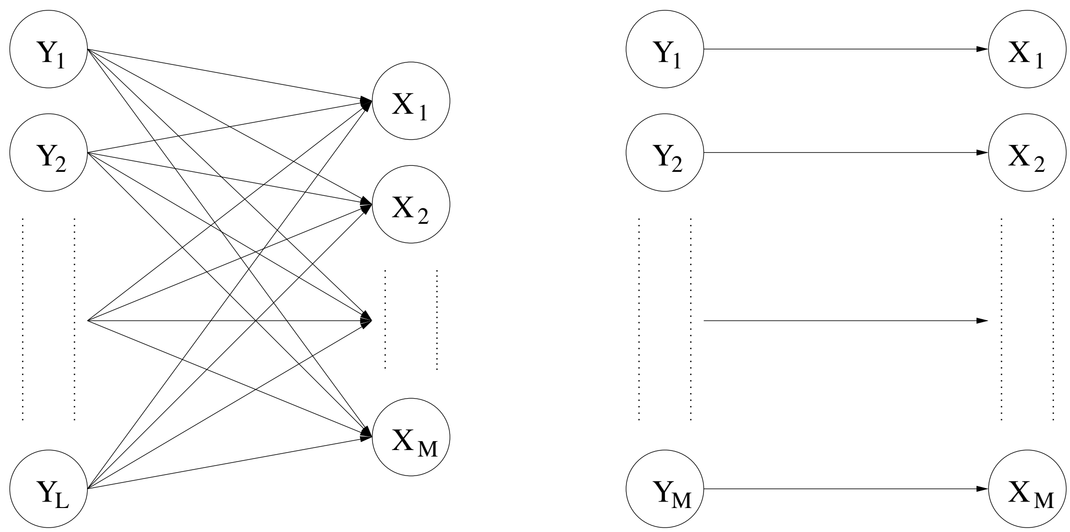

Figure 4.1: General conditional independence model (left) and correspondence conditional independence model (right).

If $\boldsymbol{\upsilon}_m$ is the set of parameters specifying $P(x_m|\boldsymbol{y})$, then $\boldsymbol{\upsilon}_1, \ldots, \boldsymbol{\upsilon}_M$ minimizing $KL(P \parallel P_{CI}(\boldsymbol{x}|\boldsymbol{y}))$ can be obtained by optimizing each of the $KL(P \parallel P_{CI}(x_m|\boldsymbol{y}))$ independently of others. Modeling $P(x_m|\boldsymbol{y})$ by specifying the probabilities for all combinations of values for X_m and $\boldsymbol{Y}$ is unrealistic due to the amount of data and computational demands (as it requires the estimation $O\left(B^{M+1}\right)$ parameters), so we need to make some simplifying assumption about either the structure or the functional form of $P_{CI}(x_m|\boldsymbol{y})$. For example, if $\mathcal{X} = \mathcal{Y}$, we can consider a conditional independence model where each x_m is dependent only on the corresponding y_m (Figure 4.1, right):

$$P_{CI}(\boldsymbol{x}|\boldsymbol{y}) = \prod_{m=1}^{M} P_{CI}(x_m|y_m). \tag{4.2}$$

4.2 Tree-based Approximations

Chow and Liu (1968) proposed a method for approximating the joint distribution of a set of discrete variables using products of distributions involving no more than pairs of variables. If $P(\boldsymbol{x})$ is an M-variate distribution on discrete variables and $\mathcal{V} = \{v_1, \ldots, v_M\}$ is a set of nodes corresponding to variables in $\mathcal{X}$, the Chow-Liu method constructs a distribution $T(\boldsymbol{x})$ for which the corresponding Bayesian and Markov network is tree-structured. If $\mathcal{G}_T = (\mathcal{V}, \mathcal{E}_T)$ is the Markov network associated with T, then

$$T(\boldsymbol{x}) = \frac{\prod_{\{u,v\}\in\mathcal{E}_T} T(x_u, x_v)}{\prod_{v\in\mathcal{V}} T(x_v)^{degree(v)}} = \prod_{\{u,v\}\in\mathcal{E}_T} \frac{T(x_u, x_v)}{T(x_v)\,T(x_u)} \prod_{v\in\mathcal{V}} T(x_v). \tag{4.3}$$

Chow and Liu showed that in order to minimize $KL(P \parallel T)$, the edges for the tree $\mathcal{E}_T$ have to be selected to maximize the total mutual information $\sum_{\{u,v\}\in\mathcal{E}_T} I(x_u, x_v)$ of the edges. Mutual information between variables x_u and x_v is defined as

$$I(x_u, x_v) = \sum_{X_u} \sum_{X_v} P(x_u, x_v) \ln \frac{P(x_u, x_v)}{P(x_u)\,P(x_v)}. \tag{4.4}$$

The optimal tree can be found by calculating mutual information $I(x_u, x_v)$ for all possible pairs of variables in $\mathcal{V}$, and then solving the maximum spanning tree problem, with pairwise mutual information from Equation 4.4 as edge weights (e.g., Cormen et al., 1990). Once the edges are selected, the probability distribution T on the pairs of vertices connected by edges is defined to be the same as in the target distribution P:

$$\forall \{u, v\} \in \mathcal{E}_T \quad T(x_u, x_v) = P(x_u, x_v), \tag{4.5}$$

and the resulting distribution T minimizes $KL(P \parallel T)$. Figure 4.2 outlines the algorithm for finding T.

The proof that T is indeed the tree-structured distribution closest to P consists of two parts. First, for any given tree structure for T, to minimize $KL(P \parallel T)$,

Algorithm ChowLiu(P)
Inputs: Distribution P over domain $\mathcal{X}$; procedure MWST(weights) that outputs a maximum weight spanning tree over $\mathcal{V}$

1. Compute marginal distributions $P(x_u, x_v)$ and $P(x_u)$ $\quad \forall u, v \in \mathcal{V}$
2. Compute mutual information values $I(x_u, x_v)$ $\quad \forall u, v \in \mathcal{V}$
3. $\mathcal{E}_T = \mathrm{MWST}(\{I(x_u, x_v)\})$
4. Set $T(x_u, x_v) = P(x_u, x_v) \quad \forall \{u, v\} \in \mathcal{E}_T$

Output: T

Figure 4.2: Chow-Liu algorithm (very similar to Meilă and Jordan, 2000).

$P \equiv T$ on all pairwise distributions corresponding to an edge in the graph and on all univariate marginals, so the optimal T must satisfy the property in (4.5). (This will be shown later, in Corollary 4.3.) For a tree-structured T satisfying (4.5),

$$\begin{aligned}
KL(P \parallel T) &= \sum_{\boldsymbol{X}} P(\boldsymbol{x}) \ln \frac{P(\boldsymbol{x})}{T(\boldsymbol{x})} \\
&= -H_P[\boldsymbol{X}] - \sum_{\boldsymbol{X}} P(\boldsymbol{x}) \left(\sum_{v \in \mathcal{V}} T(x_v) + \sum_{\{u,v\} \in \mathcal{E}_T} \frac{T(x_u, x_v)}{T(x_u) T(x_v)} \right) \\
&= -H_P[\boldsymbol{X}] - \sum_{v \in \mathcal{V}} \sum_{X_v} P(x_v) \ln P(x_v) - \sum_{\{u,v\} \in \mathcal{E}_T} \sum_{X_u, X_v} P(x_u, x_v) \ln \frac{P(x_u, x_v)}{P(x_u) P(x_v)} \\
&= -H_P[\boldsymbol{X}] + \sum_{v \in \mathcal{V}} H_P[X_v] - \sum_{\{u,v\} \in \mathcal{E}_T} I(x_u, x_v).
\end{aligned}$$

Since entropies H_P are independent of the structure of $\mathcal{G}_T$, the minimum of $KL(P \parallel T)$ corresponds to the structure maximizing $\sum_{\{u,v\} \in \mathcal{E}_T} I(x_u, x_v)$.

If each of the variables in $\mathcal{V}$ takes on B values, finding the optimal tree T has time complexity $O(M^2 B^2)$ for the mutual information calculations and $O(M^2)$ for finding the maximum spanning tree, totalling $O(M^2 B^2)$. (For the case of sparse high-dimensional data, Meilă (1999) showed that the Chow-Liu algorithm can be sped up.)

The advantages of Chow-Liu trees include (a) the existence of a simple algorithm for finding the optimal tree [4], (b) the parsimonious nature of the model (the number of parameters is linear in dimensionality of the space), and (c) the resulting tree structure T often has a simple intuitive interpretation. While there are other algorithms that retain the idea of a tree-structured distribution, while allowing for more complex dependencies (e.g., thin junction trees, Bach and Jordan, 2002), these algorithms have higher time complexity than the original Chow-Liu algorithm and do not guarantee optimality of the structure that is learned. Thus, in the results in this paper we focus on Chow-Liu trees under the assumption that they are a generally useful modeling technique in the context of multivariate data.

4.2.1 Conditional Chow-Liu Forests

We now propose an extension of the Chow-Liu method to model conditional distributions (Kirshner et al., 2004). The trivial extension of building a Chow-Liu tree for each value of the conditioning variable $\boldsymbol{Y}$ is computationally impractical due to the exponential (in M) number of free parameters. As with Chow-Liu trees, we want the corresponding probability distribution to be factored into a product of distributions involving no more than two variables. Pairs of variables are represented as an edge in a corresponding graph with nodes corresponding to variables in $\mathcal{V}_{xy} = \mathcal{V} \cup \mathcal{V}_y$. However, since all of the variables in $\mathcal{V}_y$ are observed, we are not interested in modeling $P(\boldsymbol{y})$, and do not wish to restrict $P(\boldsymbol{y})$ by making independence assumptions

[4] In fact, if we change the structure to allow cliques of size more than 2 in the graph G_T, the problem of finding optimal approximation distribution becomes NP-hard (Chickering, 1996; Srebro, 2003).

Algorithm CondChowLiu(P)
Inputs: Distribution P over domain $\mathcal{X} \times \mathcal{Y}$; procedure MWST($\mathcal{V}_{xy}$, weights) that outputs a maximum weight spanning tree over $\mathcal{V}_{xy}$

1. (a) Compute marginal distributions $P(x_u, x_v)$ and $P(x_u)$ $\forall u, v \in \mathcal{V}$
   (b) Compute marginal distributions $P(y_u)$ and $P(y_u, x_v)$ $\forall u \in \mathcal{V}_y$, $v \in \mathcal{V}$
2. (a) Compute mutual information values $I(x_u, x_v)$ $\forall u, v \in \mathcal{V}$
   (b) Compute mutual information values $I(y_u, x_v)$ $\forall u \in \mathcal{V}_y, v \in \mathcal{V}$
   (c) Find $u(v) = \arg\max_{u \in \mathcal{V}_y} I(y_u, x_v)$ $\forall v \in \mathcal{V}$
   (d) Let $\mathcal{V}' = \mathcal{V} \cup \{v'\}$, and set $I(x_{v'}, x_v) = I\left(y_{u(v)}, x_v\right)$ $\forall v \in \mathcal{V}$
3. (a) $\mathcal{E}_{T'} = \mathrm{MWST}(\mathcal{V}', \mathbf{I})$
   (b) $\mathcal{E}_x = \{\{u, v\} | u, v \in \mathcal{V}, \{u, v\} \in \mathcal{E}_{T'}\}$
   (c) $\mathcal{E}_y = \{\{u(v), v\} | v \in \mathcal{V}, \{v, v'\} \in \mathcal{E}_{T'}\}$.
4. (a) Set $T(x_u, x_v) = P(x_u, x_v)$ $\forall \{u, v\} \in \mathcal{E}_x$
   (b) Set $T(y_u, x_v) = P(y_u, x_v)$ $\forall \{u, v\} \in \mathcal{E}_y$

Output: T

Figure 4.3: Conditional Chow-Liu algorithm (Kirshner et al., 2004).

about the variables in $\mathcal{V}_y$.[5] The structure for an approximation distribution $T(\boldsymbol{x}|\boldsymbol{y})$ will be constructed by adding edges such as not to introduce paths involving multiple variables from $\mathcal{V}_y$. T can be trivially extended to a joint distribution by setting $T(\boldsymbol{y}) \equiv P(\boldsymbol{y})$.

Let $\mathcal{G}_F = (\mathcal{V}_{xy}, \mathcal{E}_F)$ be an undirected forest, a collection of disjoint trees, containing edges $\mathcal{E}_x$ between pairs of variables in $\mathcal{V}$ and edges $\mathcal{E}_y$ connecting variables in $\mathcal{V}$ and $\mathcal{V}_y$, $\mathcal{E}_F = \mathcal{E}_x \cup \mathcal{E}_y$. If the probability distribution $T(\boldsymbol{x}|\boldsymbol{y})$ has forest $\mathcal{G}_F$ for a Markov

[5]The graphical model structure not making any conditional independence assumptions is the complete graph; we may assume that the nodes in $\mathcal{V}_y$ form a clique.

network, then similar to Equation 4.3:

$$\begin{aligned} T(\boldsymbol{x}|\boldsymbol{y}) &= \prod_{\{u,v\}\in\mathcal{E}_x} \frac{T(x_u, x_v)}{T(x_u)\,T(x_v)} \prod_{v\in\mathcal{V}} T(x_v) \prod_{\{u,v\}\in\mathcal{E}_y} \frac{T(y_u, x_v)}{T(y_u)\,T(x_v)} \\ &= \prod_{\{u,v\}\in\mathcal{E}_x} \frac{T(x_u, x_v)}{T(x_u)\,T(x_v)} \prod_{v\in\mathcal{V}} T(x_v) \prod_{\{u,v\}\in\mathcal{E}_y} \frac{T(x_v|y_u)}{T(x_v)}. \end{aligned} \tag{4.6}$$

We will again use the KL-divergence, this time between conditional distributions $P(\boldsymbol{x}|\boldsymbol{y})$ and $T(\boldsymbol{x}|\boldsymbol{y})$, as an objective function:

$$KL(P \parallel T) = \sum_{\boldsymbol{Y}} P(\boldsymbol{y}) \sum_{\boldsymbol{X}} P(\boldsymbol{x}|\boldsymbol{y}) \ln \frac{P(\boldsymbol{x}|\boldsymbol{y})}{T(\boldsymbol{x}|\boldsymbol{y})}.$$

It can be shown that the optimal probability distribution T with corresponding Markov network $\mathcal{G}_F$ is

$$T(x_u, x_v) = P(x_u, x_v) \quad \forall \{u, v\} \in \mathcal{E}_x, \tag{4.7}$$

and

$$T(x_v|y_u) = P(x_v|y_u) \quad \forall \{u, v\} \in \mathcal{E}_y. \tag{4.8}$$

As with the unconditional distribution, we wish to find pairs of variables to minimize

$$KL(P \parallel T) = \sum_{v\in\mathcal{V}_x} H_P[X_v] - H_P[\boldsymbol{X}|\boldsymbol{Y}] - \sum_{\{u,v\}\in\mathcal{E}_x} I(x_u, x_v) - \sum_{\{u,v\}\in\mathcal{E}_y} I(y_u, x_v) \tag{4.9}$$

where $H_P[X_v]$ denotes the entropy of $P(x_v)$, and $H_P[\boldsymbol{X}|\boldsymbol{Y}]$ denotes the conditional entropy of $P(\boldsymbol{x}|\boldsymbol{y})$. Both $H_P[X_v]$ and $H_P[\boldsymbol{X}|\boldsymbol{Y}]$ are independent of $\mathcal{E}_F$, so as in the unconditional case, we need to solve a maximum spanning tree problem on the graph with nodes $\mathcal{V}_{xy}$ while not allowing paths between vertices in $\mathcal{V}_y$ (alternatively, assuming all nodes in $\mathcal{V}_y$ are connected).

The algorithm for learning the conditional Chow-Liu (CCL) distribution is shown in Figure 4.3. As with the case of joint distribution Chow-Liu trees, the proof of the algorithm's correctness consists of two parts. For the first part, we need to show that Equations 4.7 and 4.8 hold for all T with a forest Markov network $\mathcal{G}_F$ minimizing $KL\left(P\left(\boldsymbol{x}|\boldsymbol{y}\right) \parallel T\left(\boldsymbol{x}|\boldsymbol{y}\right)\right)$. The second part describing how to select the edges of the tree is a direct extension of the original Chow and Liu (1968) proof for the Expression 4.9.

Theorem 4.1. *Let* $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *be vectors of categorical random variables defined on* $\mathcal{X}$ *and* $\mathcal{Y}$*, respectively, and let* $P\left(\boldsymbol{x},\boldsymbol{y}\right)$ *be a probability distribution defined on* $\mathcal{X}\times\mathcal{Y}$*. Let* $T\left(\boldsymbol{x}|\boldsymbol{y}\right)$ *be a conditional distribution on* $\mathcal{X}$ *with conditional independencies represented by a Bayesian network* $\mathcal{G}_B = \left(\mathcal{V}_{xy}, \mathcal{E}_B\right)$ *such that there are no arrows from* $v \in \mathcal{V}$ *to* $u \in \mathcal{V}_y$*, and thus the probability distribution factorizes as*

$$T\left(\boldsymbol{x}|\boldsymbol{y}\right) = \prod_{v\in\mathcal{V}} T\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) \tag{4.10}$$

where $pa_x\left(v\right)$ *and* $pa_y\left(v\right)$ *are the parents of* v *from* $\mathcal{V}$ *and* $\mathcal{V}_y$*, respectively. Let* $P_B\left(\boldsymbol{x}|\boldsymbol{y}\right)$ *be a conditional distribution with the Bayesian network* $\mathcal{G}_B$ *matching* P *on all conditional factors:*

$$\begin{aligned} P_B\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) &= P\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) \quad \forall v \in \mathcal{V} \\ \textit{and } P_B\left(\boldsymbol{x}|\boldsymbol{y}\right) &= \prod_{v\in\mathcal{V}} P_B\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right). \end{aligned}$$

Then

$$\begin{aligned} KL\left(P\left(\boldsymbol{x}|\boldsymbol{y}\right)\right. &\parallel \left. T\left(\boldsymbol{x}|\boldsymbol{y}\right)\right) = KL\left(P\left(\boldsymbol{x}|\boldsymbol{y}\right) \parallel P_B\left(\boldsymbol{x}|\boldsymbol{y}\right)\right) \\ &+ \sum_{v\in\mathcal{V}} KL\left(P\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) \parallel T\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right)\right). \end{aligned}$$

See Appendix B for a proof. As a consequence, $T \equiv P_B$ minimizes $KL\left(P \parallel T\right)$ over all distributions with given dependency structure $\mathcal{G}_B$.

For the modeling of a joint distribution T (as opposed to conditional), we can obtaining the corresponding result by setting $\mathcal{Y} = \emptyset$ and $\mathcal{V}_y = \emptyset$. Directed tree-structured (or forest-structured) distributions have additional properties.

Theorem 4.2. *Let distributions P and T be as defined in Theorem 4.1. Further assume that $\mathcal{G}_B = (\mathcal{V}_{xy}, \mathcal{E}_B)$ is a forest (i.e., $|pa_x(v)| + |pa_y(v)| \leq 1$) with*

$$T\left(x_v | x_{pa_x(v)}, y_{pa_y(v)}\right) = P\left(x_v | x_{pa_x(v)}, y_{pa_y(v)}\right) \quad \forall v \in \mathcal{V}$$

with $\boldsymbol{Y}$ distributed according to $P(\boldsymbol{y})$. Then $\forall v \in \mathcal{V}$,

$$T(x_v) = P(x_v).$$

See Appendix B for a proof.

Corollary 4.3. *For P, T, and $\mathcal{G}_B = (\mathcal{V}_{xy}, \mathcal{E}_B)$ defined as in Theorem 4.2,*

$$\begin{aligned} T(x_u, x_v) &= P(x_u, x_v) \quad \forall (u, v) \in \mathcal{E}_D \cap \mathcal{V} \times \mathcal{V}, \text{ and} \\ T(y_u, x_v) &= P(y_u, x_v) \quad \forall (u, v) \in \mathcal{E}_D \cap \mathcal{V}_y \times \mathcal{V}. \end{aligned}$$

Proof. Follows directly from Theorems 4.1 and 4.2. $\square$

From Theorem 4.1 and Corollary 4.3, it also follows that if $\mathcal{G}_B$ is a forest, then

$$KL(P \parallel T) = KL(P \parallel P_B) + KL(P_B \parallel T).$$

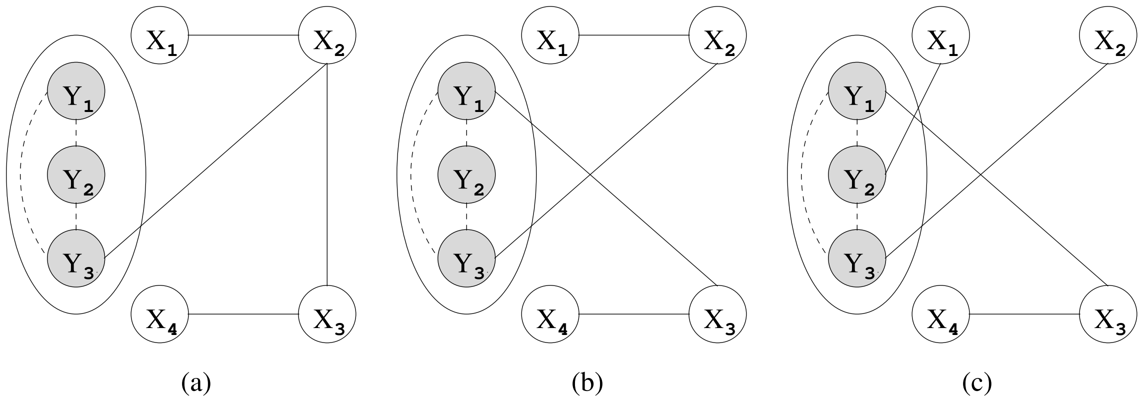

Figure 4.4: Conditional CL forest for a hypothetical distribution with (a) 1 component (b) 2 components (c) 3 components.

To show that Equations 4.7 and 4.8 hold, we need to convert an undirected graph $\mathcal{G}_F$ into a directed graph $\mathcal{G}_D$. This can be accomplished by having each of the trees in $\mathcal{G}_F$ have its root at a vertex from $\mathcal{V}_y$ if that tree has one, or in one of the leaves otherwise, and then applying the algorithm of creating a directed tree with a given root given an undirected tree (Pearl, 1988). Then each edge $(u, v) \in \mathcal{E}_B$ ($(u, v) \in \mathcal{V}_{xy} \times \mathcal{V}$) corresponds to an edge $\{u, v\} \in \mathcal{E}_F$. For $\{u, v\} \in \mathcal{E}_x$, $T(x_u, x_v) = P(x_u, x_v)$ by Corollary 4.3. For $\{u, v\} \in \mathcal{E}_y$, $T(x_v|y_u) = P(x_v|y_u)$ by Theorem 4.1.

This problem of finding the best forest is equivalent to finding a maximum weight spanning tree in a graph with vertices $\mathcal{V} = \mathcal{V} \cup \mathcal{V}_y$ where all nodes in $\mathcal{V}_y$ are already connected, and the weights of edges connecting pairs of nodes in $\mathcal{V}$ or pairs with one node in $\mathcal{V}$ and one in $\mathcal{V}_y$ are determined by appropriate mutual informations. We can view $\mathcal{V}_y$ as a supernode with weights from this supernode v' to a node $v \in \mathcal{V}$ determined as the maximum mutual information from any node in $\mathcal{V}_y$ to v, i.e.,

$$weight(v', v) = \max_{u \in \mathcal{V}_y} I(y_u, x_v).$$

Due to the restrictions on the edges, the CCL networks can contain disconnected tree components (referred to as forests). These CCL forests can consist of 1 to $\min\{|V_y|, |V_x|\}$ components. (See Figure 4.4 for an illustration.)

4.3 Beyond Trees

Trees provide flexible and parsimonious distributions for multivariate data; however, while they are richer than the conditional independence model, the trees are quite restrictive in structure. This is especially evident in conditional Chow-Liu forests where due to the restriction of having no more than one parent in a Bayesian network representation, the conditional distribution splits into multiple subgraphs and corresponding factors. In this section, we consider models with richer dependency structures, not having the limitations of the tree-structured networks. Potentially, they can be used to model any categorical-valued distribution; however, these structures lose some of the desirable properties of trees. By leaving the restricted domain of trees, the search for the optimal structure and the parameters given the structure are decoupled and cannot be easily performed simultaneously. We also lose the structure optimality guarantee (Chickering, 1996; Srebro, 2003). As a result, the algorithms for non-tree structures may have significantly higher computational complexity than algorithms for tree-structured distributions.

We will utilize *Maximum Entropy* models, MaxEnt for short, as building blocks for models when the tree structure for an approximating distribution is undesirable or insufficient. The MaxEnt model can be viewed as the "smoothest" probability distribution satisfying constraints on some of its low-dimensional marginals. Consider an example using the precipitation occurrence modeling as an application.

Example 4.1. Assume we have an empirical joint distribution $P(\boldsymbol{x})$ of occurrence for five stations with variable X_i corresponding to station $i = 1, \ldots, 5$.[6] We want to approximate our target P with a distribution $\hat{P}(\boldsymbol{x})$ which matches all univariate marginals and also matches the simultaneous rainfall occurrence for a pair of stations 1 and 2, and for a triple of stations 3, 4, and 5:

$$\begin{aligned}
\hat{P}(X_i = 1) &= P(X_i = 1), \quad i = 1, \ldots, 5, \\
\hat{P}(X_1 = 1, X_2 = 1) &= P(X_1 = 1, X_2 = 1), \\
\hat{P}(X_3 = 1, X_4 = 1, X_5 = 1) &= P(X_3 = 1, X_4 = 1, X_5 = 1).
\end{aligned}$$

Alternatively, the constraints of the marginals can be expressed as

$$\begin{aligned}
\sum_{\boldsymbol{X}} \hat{P}(\boldsymbol{x}) f_i(x_i) &= \sum_{\boldsymbol{X}} P(\boldsymbol{x}) f_i(x_i), \quad i = 1, \ldots, 5, \\
\sum_{\boldsymbol{X}} \hat{P}(\boldsymbol{x}) f_{12}(x_1, x_2) &= \sum_{\boldsymbol{X}} P(\boldsymbol{x}) f_{12}(x_1, x_2), \\
\sum_{\boldsymbol{X}} \hat{P}(\boldsymbol{x}) f_{345}(x_3, x_4, x_5) &= \sum_{\boldsymbol{X}} P(\boldsymbol{x}) f_{345}(x_3, x_4, x_5)
\end{aligned}$$

where functions f called *features* are defined as

$$f_i(x_i) = x_i, \quad f_{12}(x_1, x_2) = x_1 x_2, \quad \text{and } f_{345}(x_3, x_4, x_5) = x_3 x_4 x_5.$$

More generally, we can define features as functions defined on subspaces of $\mathcal{X} \times \mathcal{Y}$. Each feature f is defined as a function $f : \mathcal{X}_{D_x(f)} \times \mathcal{Y}_{D_y(f)} \to \mathbb{R}$ where $D_x(f) \subseteq \mathcal{V}$ and $D_y(f) \subseteq \mathcal{V}_y$ are sets of components of vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively, which are used in defining the feature f. Each feature can be trivially extended to the domain $\mathcal{X} \times \mathcal{Y}$ by setting $f(\boldsymbol{x}, \boldsymbol{y}) = f\left(\boldsymbol{x}_{D_x(f)}, \boldsymbol{y}_{D_y(f)}\right)$. Let $\mathcal{F}$ be the set of features. The

[6] We are deliberately ignoring the temporal dependence of precipitation in this example.

constraint for each feature $f \in \mathcal{F}$ can be written as

$$\sum_{\boldsymbol{X}} \sum_{\boldsymbol{Y}} P(\boldsymbol{y}) \hat{P}(\boldsymbol{x}|\boldsymbol{y}) f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\boldsymbol{X}} \sum_{\boldsymbol{Y}} P(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{x}, \boldsymbol{y}). \tag{4.11}$$

We will denote a constraint corresponding to a feature f by c_f, and the set of such constraints by $\mathcal{C}$.

Assume that $\mathcal{C}$ is consistent. We wish to find a distribution $\hat{P}$ satisfying all constraints in $\mathcal{C}$ and also such that $\hat{P}$ is most similar to some other exponential distribution $Q(\boldsymbol{x}|\boldsymbol{y})$, i.e., minimizing $KL\left(\hat{P} \parallel Q\right)$. If Q is uniform on $\mathcal{X}$, this is equivalent to satisfying $\mathcal{C}$ while maximizing the entropy of $\hat{P}$, lending the name of the model. Unless otherwise specified, we assume that Q is uniform. By differentiating $KL\left(\hat{P} \parallel Q\right)$ with Lagrangians added for constraints in $\mathcal{C}$, we obtain the following functional form of the solution P_{ME} (e.g., see Berger et al., 1996):

$$P_{ME}(\boldsymbol{x}|\boldsymbol{y}) = \frac{Q(\boldsymbol{x}|\boldsymbol{y})}{Z(\boldsymbol{y})} \exp\left(\sum_{f \in \mathcal{F}} \lambda_f f(\boldsymbol{x}, \boldsymbol{y})\right) \tag{4.12}$$

$$\text{where } Z(\boldsymbol{y}) = \sum_{\boldsymbol{X}} Q(\boldsymbol{x}|\boldsymbol{y}) \exp\left(\sum_{f \in \mathcal{F}} \lambda_f f(\boldsymbol{x}, \boldsymbol{y})\right)$$

is a normalization function to enforce $\sum_{\boldsymbol{X}} P_{ME}(\boldsymbol{x}|\boldsymbol{y}) = 1$ with $\lambda_f \in \mathbb{R}$ for all $f \in \mathcal{F}$. Let $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathcal{F}|}$ be a vector of coefficients λ_f for this exponential model with the components of $\boldsymbol{\lambda}$ indexed by $f \in \mathcal{F}$. To specify a MaxEnt model, one needs to specify the set of features $\mathcal{F}$ and then to estimate the set of parameters $\boldsymbol{\lambda}$ minimizing $KL(P \parallel P_{ME}(\cdot|\boldsymbol{\lambda}, \mathcal{F}))$.

It is worth pointing out some of the properties of the MaxEnt family of distributions. The set of parameters $\boldsymbol{\lambda}$ satisfying all of the consistent constraints $\mathcal{C}$ also minimizes the relative entropy $KL\left(P \parallel P_{ME}\right)$, a proxy for a negated log-likelihood $-\sum_{\boldsymbol{X},\boldsymbol{Y}} P\left(\boldsymbol{x},\boldsymbol{y}\right) \ln P_{ME}\left(\boldsymbol{x}|\boldsymbol{y}\right)$. This property makes MaxEnt models fit easily in the maximum likelihood framework of parameter estimation. While the minimum often cannot be found in closed form, $KL\left(P \parallel P_{ME}\right)$ is concave as a function of $\boldsymbol{\lambda}$, so the minimum is unique[7] (Della Pietra et al., 1997) and can be found by changing $\boldsymbol{\lambda}$ in an iterative manner decreasing $KL\left(P \parallel P_{E}\right)$ at each iteration.

The dependency network for a distribution from the MaxEnt family can be found from the domain of its features. The edges $\mathcal{E}$ of a Markov network of a distribution $P_{ME}\left(\boldsymbol{x}\right)$ or a *Conditional Random Field* or CRF (Lafferty et al., 2001) for a conditional distribution $P_{ME}\left(\boldsymbol{x}|\boldsymbol{y}\right)$ consist of pairs of vertices corresponding to variables of $\boldsymbol{X}$ appearing in at least one feature, i.e.,

$$\mathcal{E} = \left\{\{u,v\} \in \mathcal{V} \times \mathcal{V} : u \neq v \text{ and } \exists f \in \mathcal{F} \ \left(u \in D_x\left(f\right) \text{ and } v \in D_x\left(f\right)\right)\right\}.$$

Then features $f \in \mathcal{F}$ can be seen as specifying potentials on the cliques $D_x\left(f\right)$ with the rest of the cliques (unspecified by features) in the Hammersley-Clifford decomposition having the potential of 1.

Let us again consider the Example 4.1. The set of features $\mathcal{F} = \{f_1, f_2, f_3, f_4, f_5, f_{12}, f_{345}\}$ gives rise to the MaxEnt solution

$$\begin{aligned} P_{ME}\left(\boldsymbol{x}\right) = \frac{1}{Z} \exp\big(\lambda_1 f_1\left(x_1\right) &+ \lambda_2 f_2\left(x_2\right) + \lambda_3 f_3\left(x_3\right) + \lambda_4 f_4\left(x_4\right) + \\ &+ \lambda_5 f_5\left(x_5\right) + \lambda_{12} f_{12}\left(x_1, x_2\right) + \lambda_{345} f_{345}\left(x_3, x_4, x_5\right)\big). \end{aligned} \tag{4.13}$$

[7] The distribution satisfying the set of constraints is unique. Depending on the set of constraints, there could be multiple sets of parameters corresponding to the same MaxEnt distribution.

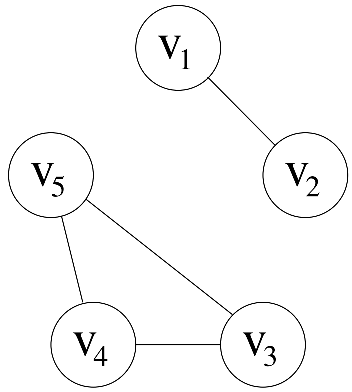

Figure 4.5: Markov Network for the MaxEnt model in (4.13).

Let $\mathcal{V} = \{v_1, \ldots, v_5\}$ with vertex v_i corresponding to variable $\boldsymbol{X}_i$, $i = 1, \ldots, 5$. There are only two constraints, f_{12} and f_{345} with domains consisting of more than two variables. The corresponding Markov network is shown in Figure 4.5.

Tree-structured distributions described in Section 4.2 can also be represented in the MaxEnt form. Suppose that each variable (either Y_u or X_v) can take on one of B possible values, $0, \ldots, B-1$. Let $\mathcal{E}$ be the set of edges defined on $\mathcal{V}_{xy} \times \mathcal{V}$. The set of features $\mathcal{F}$ would consist of univariate features

$$f_{vb}(x_v) = \begin{cases} 1 & X_v = b, \\ 0 & X_v \neq b, \end{cases}$$

for $v \in \mathcal{V}$ and $b = 1, \ldots, B-1$, and of bivariate features

$$f_{uvb_1b_2}(y_u, x_v) = \begin{cases} 1 & Y_u = b_1 \text{ and } X_v = b_2, \\ 0 & \text{otherwise}, \end{cases} \quad \forall (u, v) \in \mathcal{E} \cap \mathcal{V}_y \times \mathcal{V}$$

and

$$f_{uvb_1b_2}(x_u, x_v) = \begin{cases} 1 & X_u = b_1 \text{ and } X_v = b_2, \\ 0 & \text{otherwise,} \end{cases} \quad \forall (u, v) \in \mathcal{E} \cap \mathcal{V} \times \mathcal{V}$$

for $b_1, b_2 = 1, \ldots, B-1$.

Example 4.2. As another illustration of the models representable in the MaxEnt form, we consider a model for a joint probability distribution matching all of the bivariate marginals of the target distribution $P(\boldsymbol{x})$. We will refer to this model as a *full bivariate MaxEnt* model. For simplicity, we assume that $\boldsymbol{X}$ is M-variate with each variables taking on B values. The set of features $\mathcal{F}$ for a MaxEnt model will consist of $B-1$ univariate features

$$f_{vb} = (x_v) = \begin{cases} 1 & X_v = b, \\ 0 & \text{otherwise,} \end{cases}$$

for each of M variables and $(B-1)^2$ of bivariate features

$$f_{uvb_1b_2}(x_u, x_v) = \begin{cases} 1 & X_u = b_1 \text{ and } X_v = b_2, \\ 0 & \text{otherwise,} \end{cases}$$

for each of $M(M-1)/2$ pairs of variables. The Markov network for the model is an M-vertex clique.

Learning of the MaxEnt model involves the construction of the set of features $\mathcal{F}$ (and the corresponding constraints) and the estimation of the parameters $\boldsymbol{\lambda}$. In the next section (4.3.1), we discuss the estimation of the parameters $\boldsymbol{\lambda}$ given the set of features $\mathcal{F}$ and the target distribution $P(\boldsymbol{x}|\boldsymbol{y})$. In Section 4.3.4, we outline an algorithm for the construction of a set of bivariate features and the estimation of its parameters for a specialized class of distributions built on a MaxEnt model.

4.3.1 Learning Parameters of Maximum Entropy Models

There are a number of algorithms for learning $\boldsymbol{\lambda}$ by maximizing $KL\left(P \parallel P_E\right)$ for a specified set of features $\mathcal{F}$, given the data making up the empirical distribution P. Early methods optimized the joint probability model ($\mathcal{Y} = \emptyset$), e.g., iterative scaling (Brown, 1959; Csiszár, 1975), generalized iterative scaling (GIS) (Darroch and Ratcliff, 1972); later methods allow the optimization of conditional distributions, e.g., IS (Jelinek, 1998, page 235), GIS (Rosenfeld, 1994), improved iterative scaling (IIS) (Berger et al., 1996; Della Pietra et al., 1997), sequential conditional generalized IS (Goodman, 2002), and a number of methods based on general optimization, e.g., gradient descent, conjugate gradient methods (Polak-Ribiere or Fletcher-Reeves), or second-order methods (e.g., Malouf, 2002). All of the algorithms are computationally intensive as they require multiple iterations; each iteration requires summation over all values of $\mathcal{Y}$ (which is usually replaced by summing over the $\boldsymbol{Y}$ in the data) and a summation over all values of $\mathcal{X}$, a number exponential in M. A summation of $\boldsymbol{X}$ can often be decomposed into lower-dimensional sums (equivalently representable by message passing methods described later) or replaced by sampling.

We can utilize the structure of the Markov network (or Markov random field) $\mathcal{G}$ of P_E to solve the optimization problem by a message passing algorithm that updates parameters corresponding to the features defined on the cliques of the junction tree (Jiroušek and Přeučil, 1995; Bach and Jordan, 2002) and propagates messages with updates of the probability distributions defined on these cliques to other cliques in the junction tree. Let $\mathcal{J} = (\mathcal{C}_J, \mathcal{E}_J)$ be a junction tree for $\mathcal{G}$. Recall that the junction

Algorithm $\text{EfficientMaxEntLearning}(P, \mathcal{F}, \mathcal{J})$
Inputs: Distribution $P(\boldsymbol{X}|\boldsymbol{Y})$; a set of features $\mathcal{F}$; junction tree $\mathcal{J}$ capturing conditional independence relationships as assumed by $\mathcal{F}$

1. Initialization:
   (a) Assign each $\boldsymbol{\lambda}_f \in \boldsymbol{\lambda}$ to one of the cliques $C \in \mathcal{C}$ such that $D_x(f) \subseteq C$;
   - Denote the set of λ_f assigned to clique C by $\boldsymbol{\lambda}_C$
   - Denote the set of f with λ_f assigned to C by $\mathcal{F}_C$

   (b) Set $\lambda_f = 0$ for all $f \in \mathcal{F}$
2. Repeat step (3) until convergence
3. Loop through $C \in \mathcal{C}$
   (a) Set C as the root of $\mathcal{J}$
   (b) Propagate evidence R from the leaves of $\mathcal{J}$
   (c) Update $\boldsymbol{\lambda}_C$ by decreasing $KL(P(\boldsymbol{x}_C|\boldsymbol{y}) \parallel P_{ME}(\boldsymbol{x}_C|\boldsymbol{y}))$ (P_{ME} from Equation 4.14) using any of the update algorithms (IIS, GIS, conjugate gradients)

Output: Set of coefficients $\boldsymbol{\lambda}$ minimizing $KL(P \parallel P_{ME})$

Figure 4.6: Efficient updating of MaxEnt parameters (similar to Bach and Jordan, 2002).

tree exists if and only if the corresponding graph $\mathcal{G}$ is chordal; if $\mathcal{G}$ is not chordal, it has to be triangulated (made chordal) first. For reasons which will be clear a little later, we would like to use a triangulation resulting in a decomposable graph with the smallest total of values of $\boldsymbol{X}$ assignable to each maximal clique. However, the problem of finding such a triangulation is NP-complete (Arngborg et al., 1987), and unless the graph is already decomposable (can be efficiently checked by Maximum Cardinality Search, Tarjan and Yannakakis, 1984), other efficient but possibly suboptimal algorithms for triangulation are often used instead (e.g., Kjærulff, 1990). After the maximal cliques are identified, an optimal junction tree can be constructed in a greedy fashion (Jensen and Jensen, 1994). Once the junction tree $\mathcal{J}$ for a graph $\mathcal{G}$ is obtained, an update of $\boldsymbol{\lambda}$ can be performed without summing over all of $\mathcal{X}$ but only over $|C|$ components of $\boldsymbol{X}$ for all maximal cliques $C \in \mathcal{C}_J$ of $\mathcal{G}$. Bach and Jordan (2002) described the algorithm for decomposed improved iterative scaling (IIS), but any algorithm for learning conditional exponential models can be used inside the loop. A version of this algorithm is shown in Figure 4.6. At the heart of the algorithm is the observation that each update of parameters $\boldsymbol{\lambda}$ can be decomposed into updates of subsets of parameters $\boldsymbol{\lambda}_C$ corresponding to cliques $C \in \mathcal{C}$ of the junction tree. Then each subset of parameters can be updated by first computing evidence $R(\boldsymbol{x}_C, \boldsymbol{y}) = \sum_{\boldsymbol{X}_{\overline{C}}} \exp\left(\sum_{f \notin \mathcal{F}_C} \lambda_f f(\boldsymbol{x}, \boldsymbol{y})\right)$ and then updating $\boldsymbol{\lambda}_C$ to improve

$$P_{ME}(\boldsymbol{x}_C|\boldsymbol{y}) = \frac{1}{Z(\boldsymbol{y})} R(\boldsymbol{x}_C, \boldsymbol{y}) \exp\left(\sum_{f \in \mathcal{F}_C} \lambda_f f(\boldsymbol{x}_C, \boldsymbol{y})\right). \tag{4.14}$$

$R(\boldsymbol{x}_C, \boldsymbol{y})$ can be computed in a message passing manner by propagating evidence R from the leaves of the junction tree to the clique C.[8] Each evidence propagation requires summations over all cliques of $\mathcal{C}$; the algorithm can be sped up by not recomputing the sums for the unchanged paths of $\mathcal{J}$. Updates for each clique requires summation over all values of $\boldsymbol{X}$ for that clique; this sum has a number of terms exponential in the size of the clique, so the computational complexity of the algorithm is at least exponential in the size of the largest clique.

It is worth noting that EfficientMaxEntLearning can also be viewed as a variation of the Expectation Propagation (EP) algorithm (cf. Minka, 2001).

Example 4.3. To illustrate how the parameter estimation algorithm works, consider the following setup. Let $\mathcal{Y} = \emptyset$, $\mathcal{X} = \{0,1\}^6$ with corresponding $\mathcal{V} = \{a, b, c, d, e, f\}$.

[8]Equation 4.14 can be represented in the form of (4.12) by normalizing $R(\boldsymbol{x}_C, \boldsymbol{y})$.

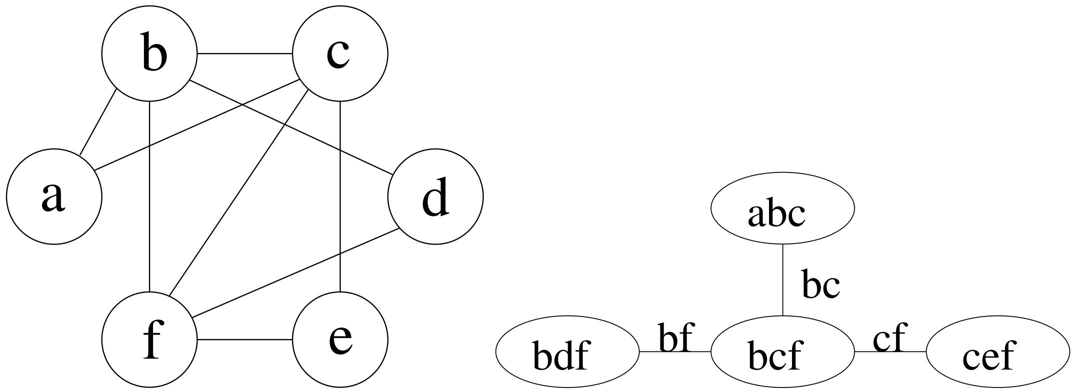

Figure 4.7: Markov network for the Example 4.3 (left) and its junction tree (right).

Let $\mathcal{F} = \{f_a, f_b, f_c, f_d, f_e, f_f, f_{ab}, f_{ac}, f_{bc}, f_{bd}, f_{bf}, f_{ce}, f_{cf}, f_{df}, f_{ef}\}$, and let

$$P_{ME}\left(\boldsymbol{x}|\boldsymbol{\lambda}, \mathcal{F}\right) = \frac{\exp\left(\sum_{f\in\mathcal{F}}\right)}{\sum_{\boldsymbol{X}} \exp\left(\sum_{f\in\mathcal{F}}\right)}$$

be a MaxEnt distribution with features defined as $f_v\left(\boldsymbol{x}\right) = x_v$ and $f_{uv}\left(\boldsymbol{x}\right) = x_u x_v$, and corresponding parameters $\boldsymbol{\lambda} = (\lambda_a, \lambda_b, \lambda_c, \lambda_d, \lambda_e, \lambda_f, \lambda_{ab}, \lambda_{ac}, \lambda_{bc}, \lambda_{bd}, \lambda_{bf}, \lambda_{ce}, \lambda_{cf}, \lambda_{df}, \lambda_{ef})$. The Markov network for this distribution is shown in Figure 4.7 (left). This graph is decomposable, and its junction tree is shown in Figure 4.7 (right).

As the initial step the features and their corresponding parameters are partitioned into the cliques. For this illustration, we choose the partition $\mathcal{F}_{abc} = \{f_a, f_b, f_c, f_{ab}, f_{ac}, f_{bc}\}$, $\mathcal{F}_{bcf} = \{f_f, f_{bf}, f_{cf}\}$, $\mathcal{F}_{bdf} = \{f_d, f_{bd}, f_{df}\}$, and $\mathcal{F}_{cef} = \{f_e, f_{ce}, f_{ef}\}$. For updates, we will set the clique order as (abc, bcf, bdf, cef). To update $\boldsymbol{\lambda}_{abc} = (\lambda_a, \lambda_b, \lambda_c, \lambda_{ab}, \lambda_{ac}, \lambda_{bc})$, we first propagate the unnormalized distributions on the separators of the cliques:

$$\begin{aligned}
R_{bdf}\left(x_b, x_f\right) &= \sum_{X_d} \exp\left(\sum_{f \in \mathcal{F}_{bdf}} \lambda_f f\left(\boldsymbol{x}\right)\right) \\
&= \sum_{X_d} \exp\left(\lambda_d f_d\left(x_d\right) + \lambda_{bd} f_{bd}\left(x_b, x_d\right) + \lambda_{df} f_{df}\left(x_d, x_f\right)\right), \\
R_{cef}\left(x_c, x_f\right) &= \sum_{X_e} \exp\left(\sum_{f \in \mathcal{F}_{cef}} \lambda_f f\left(\boldsymbol{x}\right)\right), \\
R_{bcf}\left(x_b, x_c\right) &= \sum_{X_f} R_{bdf}\left(x_b, x_f\right) R_{cef}\left(x_c, x_f\right) \exp\left(\sum_{f \in \mathcal{F}_{bcf}} \lambda_f f\left(\boldsymbol{x}\right)\right).
\end{aligned}$$

Then $\boldsymbol{\lambda}_{abc}$ is updated to minimize $KL\left(P\left(\boldsymbol{X}_{abc}\right) \| \hat{P}_{ME}\left(\boldsymbol{X}_{abc}\right)\right)$ where

$$\hat{P}_{ME}\left(\boldsymbol{X}_{abc}\right) = \frac{R_{bcf}\left(x_b, x_c\right) \exp\left(\sum_{f \in \mathcal{F}_{abc}} \lambda_f f\left(\boldsymbol{x}\right)\right)}{\sum_{X_a, X_b, X_c} R_{bcf}\left(x_b, x_c\right) \exp\left(\sum_{f \in \mathcal{F}_{abc}} \lambda_f f\left(\boldsymbol{x}\right)\right)}.$$

Next clique to be updated is bcf. First, the evidence is propagated to bcf from the leaves:

$$\begin{aligned}
R_{abc}\left(x_b, x_c\right) &= \sum_{X_a} \exp\left(\sum_{f \in \mathcal{F}_{abc}} \lambda_f f\left(\boldsymbol{x}\right)\right), \\
R_{bdf}\left(x_b, x_f\right) &= \sum_{X_d} \exp\left(\sum_{f \in \mathcal{F}_{bdf}} \lambda_f f\left(\boldsymbol{x}\right)\right) \text{ (no need to recompute)}, \\
R_{cef}\left(x_c, x_f\right) &= \sum_{X_e} \exp\left(\sum_{f \in \mathcal{F}_{cef}} \lambda_f f\left(\boldsymbol{x}\right)\right) \text{ (no need to recompute)}.
\end{aligned}$$

Then $\boldsymbol{\lambda}_{bcf}$ is updated to minimize $KL\left(P\left(\boldsymbol{X}_{bcf}\right) \parallel \hat{P}_{ME}\left(\boldsymbol{X}_{bcf}\right)\right)$ where

$$\hat{P}_{ME}\left(\boldsymbol{X}_{bcf}\right) = \frac{R_{abc}\left(x_b, x_c\right) R_{bdf}\left(x_b, x_f\right) R_{cef}\left(x_c, x_f\right) \exp\left(\sum_{f \in \mathcal{F}_{bcf}} \lambda_f f\left(\boldsymbol{x}\right)\right)}{\sum_{X_b, X_c, X_f} R_{abc}\left(x_b, x_c\right) R_{bdf}\left(x_b, x_f\right) R_{cef}\left(x_c, x_f\right) \exp\left(\sum_{f \in \mathcal{F}_{bcf}} \lambda_f f\left(\boldsymbol{x}\right)\right)}.$$

These updates are applied in sequence to cliques abc, bcf, bdf, and cef until convergence.

4.3.2 Product of Univariate Conditional MaxEnt Model

Let us go back to the original problem of this section. We want to learn a distribution $\hat{P}\left(\boldsymbol{x}|\boldsymbol{y}\right)$ approximating a distribution $P\left(\boldsymbol{x}|\boldsymbol{y}\right)$ without restricting the dependence structure of $\hat{P}$ to a tree. In previous subsections, we considered models with undirected representation of the dependence networks. In this subsection, we consider the directed graph representation for the dependence structure.

Suppose we wish to build an approximation distribution $\hat{P}\left(\boldsymbol{x}|\boldsymbol{y}\right)$ to a target distribution $P\left(\boldsymbol{x}|\boldsymbol{y}\right)$ with a given conditional independence structure in the directed graphical form (Bayesian network). As Theorem 4.1 suggests, we can maximize the log-likelihood or minimize the relative entropy $KL\left(P \parallel \hat{P}\right)$ by setting $\hat{P}\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) = P\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right)$ thus matching all factors of $\hat{P}$ to corresponding factors of P. Direct application of this approach is often impractical as the number of parameters needed to specify each factor is exponential in the number of parent variables thus requiring exponential time complexity in the number of parents for parameter estimation. Also, in practice, P is an empirical distribution computed from data. The number of samples needed to approximate a multi-dimensional grows exponentially with the dimensionality, so a direct assignment of factors of $\hat{P}$ is only feasible for small values of $|pa_x(v)| + |pa_y(v)|$. Thus, we may have to resort to low-dimensional approximation of the factors $P\left(x_v|\boldsymbol{x}_{pa(v)}, \boldsymbol{y}_{pa_y(v)}\right)$. Among such approximations suggested in literature are decision trees (Friedman and Goldszmidt, 1996) and their extension, decision graphs (Chickering et al., 1997). These methods assume that the entries in the full probability table $P\left(x_v|\boldsymbol{x}_{pa(v)}, \boldsymbol{y}_{pa_y(v)}\right)$ can be represented by a small number of values. Thus the conditional probability distribution for each factor can be represented by these few values and the correspondences of the values of parent variables to these few values. These models store the above correspondences in a directed tree or a directed acyclic graph with each leaf node corresponding to entries in the full probability table of values for parents of the modeled variable in the factor. The graphs are constructed such that the number of leaves is relatively small compared to the full probability table.

We propose to model each of the factors in the probability distribution by a univariate conditional MaxEnt distribution introduced in the previous subsection. This representation allows to approximate a full probability table for a conditional distribution by potentially only a small number of numeric parameters. The MaxEnt representation does not necessitate the values in the probability table to be mostly the same; instead, it produces a smooth probability distribution satisfying a number of constraints imposed by the data. In the remainder of the subsection, we will describe the model, the estimation of its parameters, and we will point out how the product of factors of univariate conditional MaxEnt distributions relates to a joint probability distribution with the collective set of features.

We wish to approximate a target distribution $P(\boldsymbol{x}|\boldsymbol{y})$ by a distribution $\hat{P}(\boldsymbol{x}|\boldsymbol{y})$. Assume that a joint distribution $\hat{P}(\boldsymbol{x},\boldsymbol{y}) = P(\boldsymbol{y})\hat{P}(\boldsymbol{x}|\boldsymbol{y})$ has a Bayesian Network representation $\mathcal{G}_B = \{\mathcal{V}_{xy}, \mathcal{E}_B\}$ such that there are no edges originating in $\mathcal{V}$ and ending in $\mathcal{V}_y$:

$$\hat{P}(\boldsymbol{x}|\boldsymbol{y}) = \prod_{v\in\mathcal{V}} \hat{P}\left(\boldsymbol{x}_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right).$$

Assume that $|\mathcal{V}| = M$. Since $\mathcal{G}_B$ is a directed acyclic graph, there is an ordering I on the vertices of $\mathcal{V}$ (a bijection of $\mathcal{V}$ to $\{1,\ldots,M\}$) such that the order of all parents in $\mathcal{V}$ of a given node is smaller than the order of that node, i.e., if $v = I^{-1}(i)$, then $pa_x(v) \subseteq \{u = I^{-1}(k) : k = 1,\ldots,i-1\}$.[9] For notational simplicity, let v_i denote the i-th vertex of $\mathcal{V}$ under ordering I, i.e., $I(v_i) = i$. We will model each of the univariate factors $\hat{P}\left(x_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right)$ as a distribution from the MaxEnt family. Let $\mathcal{F}_i$ be the set of features used to define the above factor. Then $\forall f \in \mathcal{F}_i$ $D_x(f) \subseteq fa_x(v_i) \subseteq \{v_1,\ldots,v_i\}$ and $D_y(f) \subseteq pa_y(v_i) \subseteq \mathcal{V}_y$. Also, $v_i \in D_x(f)$, or the term containing f can be cancelled. Each factor can be parametrized as

$$\begin{aligned}
\hat{P}\left(x_{v_i}|\boldsymbol{x}_{pa_x(v_i)}, \boldsymbol{y}_{pa_y(v_i)}\right) &= \frac{\exp\left(\sum_{f\in\mathcal{F}_i}\delta_f f\left(\boldsymbol{x}_{D_x(f)}, \boldsymbol{y}_{D_y(f)}\right)\right)}{Z_i\left(\boldsymbol{x}_{pa_x(v_i)}, \boldsymbol{y}_{pa_y(v_i)}\right)} \qquad (4.15)\\
\text{where } Z_i\left(\boldsymbol{x}_{pa_x(v_i)}, \boldsymbol{y}_{pa_y(v_i)}\right) &= \sum_{X_{v_i}} \exp\left(\sum_{f\in\mathcal{F}_i}\delta_f f\left(\boldsymbol{x}_{D_x(f)}, \boldsymbol{y}_{D_y(f)}\right)\right),
\end{aligned}$$

and $\delta_f \in \mathbb{R}$. Let $\boldsymbol{\delta}_i$ be the vector of parameters for $\hat{P}\left(x_{v_i}|\boldsymbol{x}_{pa_x(v_i)}, \boldsymbol{y}_{pa_y(v_i)}\right)$, and let $\boldsymbol{\delta} = (\boldsymbol{\delta}_1,\ldots,\boldsymbol{\delta}_M)$ be the vector of parameters for $\hat{P}(\boldsymbol{x}|\boldsymbol{y})$. Let $\mathcal{F}_{PUC} = \{\{\mathcal{F}_1\},\ldots,\{\mathcal{F}_M\}\}$ be the set of feature sets. The functional form of this

[9] As before, $pa_y(v) \subseteq \mathcal{V}_y$.

Product of Univariate Conditional MaxEnt model (PUC-MaxEnt for short) is then

$$P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\delta},\mathcal{F}_{PUC}\right)=\prod_{i=1}^{M}P_{ME}\left(x_{v_i}|\boldsymbol{x}_{pa_x(v_i)},\boldsymbol{y}_{pa_y(v_i)},\boldsymbol{\delta}_i,\mathcal{F}_i\right). \tag{4.16}$$

Given set of features $\mathcal{F}_{PUC}$, the parameter estimation for different factors can be done independently, i.e., each $\boldsymbol{\delta}_i$ can be estimated independently of other $\boldsymbol{\delta}_j$, $j \neq i$. Any of the iterative algorithms mentioned in Section 4.3.1 can be used to learn the parameters $\boldsymbol{\delta}_i$ minimizing $KL\left(P\left(x_{v_i}|\boldsymbol{x}_{pa_x(v_i)},\boldsymbol{y}_{pa_y(v_i)}\right) \parallel P_{ME}\left(x_{v_i}|\boldsymbol{x}_{pa_x(v_i)},\boldsymbol{y}_{pa_y(v_i)},\boldsymbol{\delta}_i,\mathcal{F}_i\right)\right)$. However, for this univariate case, the parameter estimation procedure involves summation only over the conditioning variables (or data points for empirical distributions) and a single variable X_{v_i}. A conjugate gradient update algorithm is derived in Appendix A.2.

4.3.3 Connection Between MaxEnt and PUC-MaxEnt Models

We have described two family of probability distributions, both using MaxEnt models, to parametrize an approximation to a target conditional distribution $P\left(\boldsymbol{x}|\boldsymbol{y}\right)$. Assume that the total set of features $\mathcal{F}$ for both such models is the same. If we are provided with a Bayesian network and a PUC-MaxEnt distribution $P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\delta},\mathcal{F}_{PUC}\right)$, we will consider a MaxEnt distribution $P_{ME}\left(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\lambda},\mathcal{F}\right)$ with the same set of all features $\mathcal{F} = \cup_{i=1}^{M}\mathcal{F}_i$. If we are provided with a MaxEnt distribution $P_{ME}\left(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\lambda},\mathcal{F}\right)$, we can construct a partition $\mathcal{F}_{PUC} = \{\mathcal{F}_1,\ldots,\mathcal{F}_M\}$ by first choosing an ordering I, and then partitioning $\mathcal{F}$ into M sets by assigning features to the set corresponding to the variable with the highest order, $\mathcal{F}_i = \left\{f \in \mathcal{F} : i = \max_{u \in D_x(f)} I\left(u\right)\right\}$. The parents of node v_i would consist of nodes corresponding to variables appearing in features in $\mathcal{F}_i$:

$$\begin{aligned} pa_x(v_i) &= \{u \in \mathcal{V} \setminus \{v_i\} : \exists f \in \mathcal{F}_v \wedge u \in D_x(v_i)\} \text{ and} \\ pa_y(v_i) &= \{u \in \mathcal{V}_y : \exists f \in \mathcal{F}_v \wedge u \in D_y(v_i)\}. \end{aligned}$$

The MaxEnt and PUC-MaxEnt distributions are, respectively,

$$\begin{aligned} P_{ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\lambda}, \mathcal{F}) &= \frac{\exp\left(\sum_{f \in \mathcal{F}} \lambda_f f(\boldsymbol{x}, \boldsymbol{y})\right)}{Z(\boldsymbol{y})} \text{ and} \\ P_{PUC-ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\delta}, \mathcal{F}_{PUC}) &= \frac{\exp\left(\sum_{f \in \mathcal{F}} \delta_f f(\boldsymbol{x}, \boldsymbol{y})\right)}{\prod_{i=1}^{M} Z_i\left(\boldsymbol{x}_{pa_x(v_i)}, \boldsymbol{y}_{pa_y(v_i)}\right)}. \end{aligned} \tag{4.17}$$

In general, a distribution from a PUC-MaxEnt family may not belong to the MaxEnt family or vice versa as each of the families has certain dependency structure imposed by the features and the ordering (in the PUC-MaxEnt). This should be expected as causal models represented by Bayesian (directed) networks and models represented by Markov (undirected) networks while overlapping do not contain one another. For the general class of dependence models, the intersection between the sets of models representable by both Bayesian and Markov networks is the set of decomposable models with dependencies captured by chordal (decomposable) graphs (e.g., see Pearl, 1988). For a fixed domain and the total set of features, the space of models falling into each of PUC-MaxEnt family (directed) and MaxEnt family (undirected) is even smaller than that of the general models. While their intersection must also fall within the class of decomposable models, it could possibly be even more restrictive.

In the rest of this subsection, we will briefly investigate the models belonging to both PUC-MaxEnt and MaxEnt families. We will again consider the setting of approximating a target distribution $P(\boldsymbol{x}|\boldsymbol{y})$. Let $\boldsymbol{\lambda}^\star$ be the set of parameters minimizing $KL(P \parallel P_{ME}(\cdot|\boldsymbol{\lambda}, \mathcal{F}))$, and let $\boldsymbol{\delta}^\star$ be the set of parameters minimizing $KL(P \parallel P_{ME}(\cdot|\boldsymbol{\delta}, \mathcal{F}_{PUC}))$. By one of the properties of MaxEnt models, parameters minimizing the relative entropy KL result in a distribution satisfying constraints associated with each feature. The satisfied constraints are

$$\sum_{\boldsymbol{X}} \sum_{\boldsymbol{Y}} P(\boldsymbol{y}) P_{ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\lambda}^\star, \mathcal{F}) f(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\boldsymbol{X}} \sum_{\boldsymbol{Y}} P(\boldsymbol{x}, \boldsymbol{y}) f(\boldsymbol{x}, \boldsymbol{y}) \tag{4.18}$$

for all $f \in \mathcal{F}$ for MaxEnt model and

$$\begin{aligned} \sum_{\boldsymbol{X}_{fa_x(v)}} \sum_{\boldsymbol{Y}_{pa_y(v)}} & P\left(\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) P_{PUC-ME}\left(\boldsymbol{x}_v|\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}, \boldsymbol{\delta}^\star, \mathcal{F}_{PUC}\right) f(\boldsymbol{x}, \boldsymbol{y}) \\ &= \sum_{\boldsymbol{X}_{fa_x(v)}} \sum_{\boldsymbol{Y}_{pa_y(v)}} P\left(\boldsymbol{x}_{fa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) f(\boldsymbol{x}, \boldsymbol{y}) \end{aligned}$$

for all $v \in \mathcal{V}$ for all $f \in \mathcal{F}_i$ where $i = I(v)$. If a MaxEnt distribution with features $\mathcal{F}$ satisfies constraints in (4.18) for all $f \in \mathcal{F}$, then that distribution is the same as $P_{ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\lambda}^\star, \mathcal{F})$ since such a distribution is unique (e.g., Della Pietra et al., 1997). By identifying the conditions under which $P_{PUC-ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\delta}^\star, \mathcal{F}_{PUC})$ belongs to the MaxEnt family *and* satisfies constraints in (4.18) for all $f \in \mathcal{F} = \cup_{i=1}^M \mathcal{F}_i$, we will obtain $P_{PUC-ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\delta}^\star, \mathcal{F}_{PUC}) \equiv P_{ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\lambda}^\star, \mathcal{F})$. The condition of constraint satisfaction can be addressed under a very strong assumption.

Theorem 4.4. *Let I be an ordering of nodes of $\mathcal{V}$ as defined on page 70. Assume* *that satisfying constraints*

$$\sum_{\boldsymbol{X}}\sum_{\boldsymbol{Y}} P\left(\boldsymbol{y}\right) P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\delta}^{\star}, \mathcal{F}_{PUC}\right) f\left(\boldsymbol{x}, \boldsymbol{y}\right) = \sum_{\boldsymbol{X}}\sum_{\boldsymbol{Y}} P\left(\boldsymbol{x}, \boldsymbol{y}\right) f\left(\boldsymbol{x}, \boldsymbol{y}\right)$$

for all $f \in \cup_{k=1}^{i-1}\mathcal{F}_k$ *implies* $P\left(\boldsymbol{y}\right) P_{PUC-ME}\left(\boldsymbol{x}_{pa_x(v)}|\boldsymbol{y}, \boldsymbol{\delta}^{\star}, \mathcal{F}_{PUC}\right) = P\left(\boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}\right)$ *on* $\mathcal{X}_{pa_x(v_i)} \times \mathcal{Y}$, *then all of the constraints associated with features in* $\mathcal{F}$ *are satisfied by* $P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\delta}^{\star}, \mathcal{F}_{PUC}\right)$*:*

$$\sum_{\boldsymbol{X}}\sum_{\boldsymbol{Y}} P\left(\boldsymbol{y}\right) P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\delta}^{\star}, \mathcal{F}_{PUC}\right) f\left(\boldsymbol{x}, \boldsymbol{y}\right) = \sum_{\boldsymbol{X}}\sum_{\boldsymbol{Y}} P\left(\boldsymbol{x}, \boldsymbol{y}\right) f\left(\boldsymbol{x}, \boldsymbol{y}\right) \quad \forall f \in \mathcal{F}.$$

The conditions needed for distributions from the PUC-MaxEnt family to belong also to the MaxEnt family are unclear. For the case of $\mathcal{Y} = \emptyset$, each of the PUC-MaxEnt normalization functions $Z_i\left(\boldsymbol{x}_{pa(v_i)}\right) > 0$ can be viewed as an unnormalized probability distributions over $\mathcal{X}_{pa(v_i)}$. Let

$$P_{Z_i}\left(\boldsymbol{x}_{pa(v_i)}\right) = Z_i\left(\boldsymbol{x}_{pa(v_i)}\right) / \left(\sum_{\boldsymbol{X}_{pa(v_i)}} Z_i\left(\boldsymbol{x}_{pa(v_i)}\right)\right).$$

Let $\mathcal{F}_{pa(v)} = \{f \in \mathcal{F} : D\left(f\right) \subseteq pa\left(v\right)\}$. If for each $i = 1, \ldots, M$, $P_{Z_i}\left(\boldsymbol{x}_{pa(v_i)}\right)$ can be represented by a MaxEnt distribution $P_{ME}\left(\left(\boldsymbol{x}_{pa(v_i)}\right)|\boldsymbol{\xi}_{v_i}, \mathcal{F}_{pa(v_i)}\right)$, then $P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{\delta}^{\star}, \mathcal{F}_{PUC}\right)$ is a member of the MaxEnt family with features $\mathcal{F}$. This result follows from the functional form of the P_{PUC-ME} (Equation 4.17):

$$P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{\delta},\mathcal{F}_{PUC}\right)=\frac{\exp\left(\sum_{f\in\mathcal{F}}\delta_f f\left(\boldsymbol{x}\right)\right)}{\prod_{i=1}^{M}Z_i\left(\boldsymbol{x}_{pa_x(v_i)}\right)}$$

$$=\quad\frac{\exp\left(\sum_{f\in\mathcal{F}}\delta_f f\left(\boldsymbol{x}\right)\right)}{\prod_{i=1}^{M}P_{ME}\left(\boldsymbol{x}_{pa(v_i)}|\boldsymbol{\xi}_{v_i},\mathcal{F}_{pa(v_i)}\right)}\left(\prod_{i=1}^{M}\sum_{\boldsymbol{X}_{pa(v_i)}}Z_i\left(\boldsymbol{x}_{pa(v_i)}\right)\right)^{-1}$$

$$=\quad\frac{1}{Z}\times\frac{\exp\left(\sum_{f\in\mathcal{F}}\delta_f f\left(\boldsymbol{x}\right)\right)}{\prod_{i=1}^{M}\exp\left(\sum_{f\in\mathcal{F}_{pa(v_i)}}\xi_{v_i,f}f\left(\boldsymbol{x}_{pa(v_i)}\right)\right)}$$

$$=\quad\frac{1}{Z}\times\exp\left(\sum_{f\in\mathcal{F}}\delta_f f\left(\boldsymbol{x}\right)-\sum_{i=1}^{M}\sum_{f\in\mathcal{F}_{pa(v_i)}\subseteq\mathcal{F}}\xi_{v_i,f}f\left(\boldsymbol{x}_{pa(v_i)}\right)\right)$$

where

$$Z=\prod_{i=1}^{M}\frac{\sum_{\boldsymbol{X}_{pa(v_i)}}Z_i\left(\boldsymbol{x}_{pa(v_i)}\right)}{\sum_{\boldsymbol{X}_{pa(v_i)}}\exp\left(\sum_{f\in\mathcal{F}_{pa(v_i)}}\xi_{v_i,f}f\left(\boldsymbol{x}_{pa(v_i)}\right)\right)}.$$

If $P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{\delta}^{\star},\mathcal{F}_{PUC}\right)$ satisfies both the condition above and the assumption of the Theorem 4.4, then

$$P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{\delta}^{\star},\mathcal{F}_{PUC}\right)=P_{ME}\left(\boldsymbol{x}|\boldsymbol{\lambda}^{\star},\mathcal{F}\right).$$

If needed, the correspondence between $\boldsymbol{\lambda}^{\star}$ and $\boldsymbol{\delta}^{\star}$ can be established as

$$\lambda_f^{\star}=\delta_f^{\star}-\sum_{v\in\mathcal{V}:D(f)\subseteq pa(v)}\xi_{v,f}\ \forall f\in\mathcal{F}.$$

Example 4.4. Consider the same domain and the same set $\mathcal{F}$ of features as in

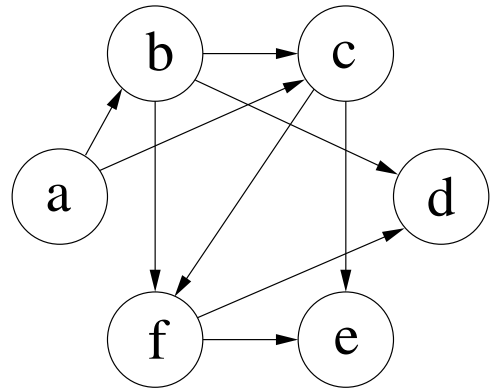

Figure 4.8: Bayesian network for PUC-MaxEnt distribution in Example 4.4.

Example 4.3. Let

$$\begin{aligned}
&\mathcal{F}_a = \{f_a\}\,, && \mathcal{F}_b = \{f_b, f_{ab}\}\,, && \mathcal{F}_c = \{f_c, f_{ac}, f_{bc}\}\,,\\
&\mathcal{F}_d = \{f_d, f_{bd}, f_{df}\}\,, && \mathcal{F}_e = \{f_e, f_{ce}, f_{ef}\}\,, && \mathcal{F}_f = \{f_f, f_{bf}, f_{cf}\}\,,
\end{aligned}$$

and let $\mathcal{F}_{PUC} = \{\mathcal{F}_v : v \in \mathcal{V}\}$. For each variable, we can compute the set of parent variables by combining the domains of corresponding features and removing the variable itself: $pa\,(a) = \emptyset$, $pa\,(b) = \{a\}$, $pa\,(c) = \{a, b\}$, $pa\,(d) = \{b, f\}$, $pa\,(e) = \{c, f\}$, $pa\,(f) = \{b, c\}$. A PUC-MaxEnt distribution with a set $\mathcal{F}_{PUC}$ of feature sets is then defined as

$$P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{\delta}, \mathcal{F}_{PUC}\right) = \prod_{v\in\mathcal{V}} P_{ME}\left(x_v|\boldsymbol{x}_{pa(v)}, \boldsymbol{\delta}_v, \mathcal{F}_{PUC}\right)$$

with a vector of parameters $\boldsymbol{\delta}$, one parameter per corresponding feature in $\mathcal{F}$. This PUC-MaxEnt distribution has the conditional independence structure captured by the Bayesian network shown in Figure 4.8.

To check whether $P_{PUC-ME}\left(\boldsymbol{x}|\boldsymbol{\delta},\mathcal{F}_{PUC}\right)$ can be represented by a MaxEnt distribution, we will examine feature sets $\mathcal{F}_p a\left(v\right)$ for $v\in\mathcal{V}$:

$$\begin{aligned}
&\mathcal{F}_{pa(a)}=\emptyset, && \mathcal{F}_{pa(b)}=\left\{f_a\right\}, && \mathcal{F}_{pa(c)}=\left\{f_a,f_b,f_{ab}\right\},\\
&\mathcal{F}_{pa(d)}=\left\{f_b,f_f,f_{bf}\right\}, && \mathcal{F}_{pa(e)}=\left\{f_c,f_f,f_{cf}\right\}, && \mathcal{F}_{pa(f)}=\left\{f_b,f_c,f_{bc}\right\}.
\end{aligned}$$

For this example, each set $\mathcal{F}_{pa(v)}$ can be used with MaxEnt model to model any probability distribution on $\mathcal{X}_{pa(v)}$. Consider, for example, a bivariate domain $\mathcal{X}_{b,f}=\mathcal{X}_{pa(d)}$. Any distribution P on $\mathcal{X}_{b,f}$ can be modeled by satisfying constraints

$$\begin{aligned}
&P_{ME}\left(X_b=1|\left\{f_b,f_f,f_{bf}\right\}\right)=\sum_{X_{b,f}}P_{ME}\left(x_b,x_f|\left\{f_b,f_f,f_{bf}\right\}\right)f_b\left(x_b\right)\\
&=\sum_{X_{b,f}}P\left(x_b,x_f\right)f_b\left(x_b\right)=P\left(X_b=1\right),\\
&P_{ME}\left(X_f=1|\left\{f_b,f_f,f_{bf}\right\}\right)=\sum_{X_{b,f}}P_{ME}\left(x_b,x_f|\left\{f_b,f_f,f_{bf}\right\}\right)f_f\left(x_f\right)\\
&=\sum_{X_{b,f}}P\left(x_b,x_f\right)f_f\left(x_f\right)=P\left(X_f=1\right),
\end{aligned}$$

and

$$\begin{aligned}
&P_{ME}\left(X_b=1,X_f=1|\left\{f_b,f_f,f_{bf}\right\}\right)=\sum_{X_{b,f}}P_{ME}\left(x_b,x_f|\left\{f_b,f_f,f_{bf}\right\}\right)f_{bf}\left(x_b,x_f\right)\\
&=\sum_{X_{b,f}}P\left(x_b,x_f\right)f_{bf}\left(x_b,x_f\right)=P\left(X_b=1,X_f=1\right).
\end{aligned}$$

Thus under this set of features $\mathcal{F}$, the $P_{PUC-ME}\left(\boldsymbol{x}|\mathcal{F}_{PUC}\right)$ distribution can also be represented by a MaxEnt distribution $P_{ME}\left(\boldsymbol{x}|\mathcal{F}\right)$. The assumption for Theorem 4.4 holds as well.

4.3.4 Selection of Features

Up to this point in this section, we have assumed that the set of features for the MaxEnt and PUC-MaxEnt models is known. However, for most problems the set of features is unknown and also has to be learned from the data. Feature selection can be accomplished by search over the space of allowed feature sets. The objective of the search is to find the set of features with the highest *score*, a measure of the goodness of fit of the resulting distribution to the data. Such a set is usually found by starting out with an empty set or a random subset of all possible features, and iteratively changing the set of features until the resulting set is satisfactory. Candidate feature sets can be obtained from the current ones by simple operations; for example, feature addition or deletion. The search strategy determines which of the candidates to take as the new set according to the scores of the candidates.

The algorithm for learning both the structure and the parameters given the structure can be viewed as two nested loops; the outer loop searches for the structure while the inner loop optimizes the parameters given the structure (Figure 4.9). There are a number of possibilities for functions $GenerateCandidates$, $score$, and $select$. The scoring function should both reflect the goodness of fit of the model with the candidate set and to also address potential overfitting. This can be accomplished by a regularization term. For example, with the Bayesian approach, the regularization term is a prior on the sets of features possibly penalizing more complex structures by assigning them a low probability mass. Alternatively, candidate sets can be restricted not to exceed a predefined complexity. Bach and Jordan (2002) consider sets of features such that the resulting junction tree has bounded width. Finding the optimal structure is often impractical or intractable (e.g., Chickering, 1996), so a heuristic approach may be used instead of an exhaustive search. If the set of candi-

Algorithm $\textsc{StructureLearning}(P, \mathcal{A})$
Inputs: Distribution $P(\boldsymbol{x}, \boldsymbol{y})$; a set of all possible features $\mathcal{A}$
Functions:$GenerateCandidates\,()$ to generate candidate subsets of features; score function $Score\,()$; selector $Select\,()$ to determine which candidate to pick given all of the scores; parameter update $LearnParameters\,()$

1. Initialization:
   (a) Select initial set of features $\mathcal{F} \subseteq \mathcal{A}$
   (b) Learn parameters $\boldsymbol{\lambda} = LearnParameters\,(\mathcal{F}, P)$
2. Loop until convergence
   (a) Compute candidate sets $\mathcal{CD} = GenerateCandidates\,(\boldsymbol{\lambda}, \mathcal{F})$
   (b) For all $c \in \mathcal{CD}$
      - Compute score $Score\,(c, \mathcal{F}, \boldsymbol{\lambda}, P)$
   (c) Update set of features $\mathcal{F} = Select\,(\mathcal{S})$ where $\mathcal{S} = \{score\,(c) : c \in \mathcal{CD}\}$
   (d) Update the parameters $\boldsymbol{\lambda} = LearnParameters\,(\mathcal{F}, P)$

Output: Set of features $\mathcal{F}$ and set of coefficients $\boldsymbol{\lambda}$

Figure 4.9: Feature selection for exponential models.

dates is constructed in a greedy fashion, new candidates are constructed by adding a single new feature to the existing set. Alternatively, new candidates can be obtained by allowing the removal of features from the current set. While we would like the scoring function to be an accurate predictor for the goodness of fit, we also want it to be cheap to compute since its computation is inside two nested loops. Some methods like Minimum Description Length (used by Friedman (1997) for Bayesian networks structure learning) require parameter learning for each candidate set of features, a computationally intensive procedure. Methods that only learn the values of the parameters corresponding to the added features (Della Pietra et al., 1997) are less accurate but faster computationally. There are many possibilities for selection of the next set of features; commonly, the next set is selected by choosing a candidate with a highest score (greedy approach).

4.3.5 PUC-MaxEnt Models for Bivariate Interactions of Binary Data

We illustrate some of the ideas from the previous two subsections by considering a special case where all of the allowed features are defined on either a single variable or a pair of binary ($\{0,1\}$) variables. Multi-site precipitation occurrence modeling (as described in Section 1.1) serves as one of the motivations as we want our precipitation occurrence model to capture pairwise interactions between the stations as well as interactions of observations on consecutive days. We allow three types of features:

- Univariate: $v \in \mathcal{V} \quad f_v(x_v) = x_v$;
- Bivariate in $\mathcal{X}$: $u, v \in \mathcal{V},\ u \neq v \quad f_{uv}(x_u, x_v) = x_u x_v$;
- Bivariate in $\mathcal{X} \times \mathcal{Y}$: $u \in \mathcal{V},\ v \in \mathcal{V}_y \quad f_{uv}(x_u, y_v) = x_u y_v$.

We will employ PUC-MaxEnt models for this task. One of the reasons for choosing a model with a causal structure instead of a Markov random field is the need to model a conditional distribution, so a directed structure is already implied. The set of features will be constructed in a greedy fashion, adding one feature at a time. Initially, the set of features will contain only all of the univariate features $f_v(x_v)$. Bivariate feature selection problem can then be viewed as edge induction. After the addition of each edge, only the parameters of the factor affected by the edge need to be recalculated while the parameters for the other $M-1$ factors are left unchanged.

The algorithm is described in Figure 4.10. To score potential directed edges, we use the *Gain* function described by Della Pietra et al. (1997). Assume we are considering adding feature f to a set of features $\mathcal{F}_v$ for the factor corresponding to node $v \in \mathcal{V}$. Gain is then defined as

$$\begin{aligned} & Gain\left(P, \mathcal{F}_{PUC}, \boldsymbol{\delta}, f\right) \\ = \quad & \max_{\delta_f} KL\left(P \parallel P_{PUC-ME}\left(\cdot | \left(\mathcal{F}_{PUC} \setminus \{\mathcal{F}_v\}\right) \cup \{\mathcal{F}_v \cup \{f\}\}, (\boldsymbol{\delta}, \delta_f)\right)\right) \\ & -KL\left(P \parallel P_{PUC-ME}\left(\cdot | \mathcal{F}_{PUC}, \boldsymbol{\delta}\right)\right), \end{aligned}$$

the change in divergence between the model with the original set of features, and the model with the added feature f to $\mathcal{F}_v$ where only the corresponding parameter δ_f is optimized. Gain is guaranteed to be non-negative as setting $\delta_f = 0$ makes the divergences equal. Gain is also a lower bound on the change in divergence (log-likelihood) when all of the parameters are optimized for the new set of features. Gain is reasonably fast to compute using an iterative optimization algorithm (Newton-Raphson), and in practice requires only a small number of iterations. The full parameter update algorithm $LearnParameters$ consists of computing parameters of M univariate conditional factors. Except for the initialization, only one of the factors requires the reestimation of its parameters, and this update can be carried out using the equations in Appendix A.2.

Note that there is no explicit regularization term in the score function. The score is always non-negative, and by adding extra features to $\mathcal{F}_{PUC}$ we are guaranteed not to decrease the log-likelihood of the data. However, to prevent overfitting, we stop adding features when the potential improvement in log-likelihood (the Gain score scaled to the amount of data) becomes too small.

Algorithm STRUCTURELEARNINGPUC-MAXENT(P)
Inputs: Distribution $P(\boldsymbol{x}, \boldsymbol{y})$
Functions: Score function $\mathit{Gain}()$; parameter update $\mathit{LearnParameters}()$

1. Initialization:
   (a) Initializing factors: $\forall v \in \mathcal{V}$
      i. Put univariate features into corresponding factor: $\mathcal{F}_v = \{f_v(x_v)\}$
      ii. Learn parameters $\boldsymbol{\delta}_v = \mathit{LearnParameters}(\mathcal{F}_v, P)$
   (b) Set list of candidate edges to all edges from $\mathcal{V}_{xy}$ to $\mathcal{V}$ except for self-loops: $\mathcal{C} = \{(u, v) \in \mathcal{V}_{xy} \times \mathcal{V} : u \neq v\}$
   (c) Compute scores of the candidates: $\forall (u, v) \in \mathcal{C}$
      i. $\mathit{score}((u, v)) = \mathit{Gain}(P, \mathcal{F}_{PUC}, \boldsymbol{\delta}, f_{uv})$
2. While $\exists (u, v) \in \mathcal{C}$ with $\mathit{score}((u, v)) > \mathit{threshold}$
   (a) Pick best candidate: $(u^\star, v^\star) = \mathit{arg}\max_{(u,v)\in\mathcal{C}} \mathit{score}((u, v))$
   (b) Update the set of candidates $\mathcal{C}$ to remove the loops and repetitions associated with adding edge $(u^\star, v^\star)$
   (c) Update factor $v^\star$:
      i. Update the list of features: $\mathcal{F}_{v^\star} = \mathcal{F}_{v^\star} \cup \{f_{u^\star v^\star}\}$
      ii. Update parameters: $\boldsymbol{\delta}_v = \mathit{LearnParameters}(\mathcal{F}_v, P)$
      iii. Update scores: $\forall (u, v^\star) \in \mathcal{C}$
         A. $\mathit{score}((u, v^\star)) = \mathit{Gain}(P, \mathcal{F}_{PUC}, \boldsymbol{\delta}, f_{uv^\star})$

Output: Set of features $\mathcal{F}_{PUC}$ and set of coefficients $\boldsymbol{\delta}$

Figure 4.10: Structure and parameter learning for PUC-MaxEnt.

4.4 Other Possibilities

4.4.1 Multivariate Probit Model

Consider a problem related to the setup in Example 4.2. We are given all of the bivariate marginals of the target distribution P either explicitly or implicitly via a joint distribution on $\mathcal{X} = \{0, 1\}^M$. We want to construct a distribution $Q(\boldsymbol{x})$ such that all the bivariate marginals of Q match the corresponding marginals of P, i.e., $\forall (u, v) \in \mathcal{V}, \quad P(x_u, x_v) = Q(x_u, x_v)$. Such a distribution Q can then be used to generate multivariate binary data with specified bivariate distributions. For example, in precipitation occurrence modeling, this model can be used to simulate multi-site rainfall occurrences while preserving pairwise interactions. One possibility is to employ a maximum entropy model with features $\forall u, v \in \mathcal{V}, \quad f(x_u) = x_u$ and $f(x_u, x_v) = x_u x_v$ as described in Example 4.2.[10] The drawback of using this maximum entropy model is that the parameter estimation using iterative proportional fitting or by another method requires multiple iterations; each iteration requires a re-estimation of $Q(x_u, x_v)$ for all $u, v \in \mathcal{V}$ requiring an exponential number of computations in M for the exact result. Alternatively, an approximation can be obtained via simulation or belief propagation, still requiring a large amount of computation. In this section we consider another type of distribution that can match all bivariate marginal distributions for the binary-valued multivariate case. This model can be learned very quickly (in time proportional to M^2), and can be used for generating data that agrees with P on all pairs of variables.

This model is based on a different parametrization of a binary distribution. We first consider the univariate case ($M = 1$). Suppose a random variable $\boldsymbol{X}$ takes

[10] This model is also sometimes referred to as *auto-logistic* model.

values from $\{-1,1\}^M$.[11] There are multiple ways to represent a univariate probability distribution for the binary domain. The univariate logistic function (a special case of the exponential model) can be represented as

$$P(x|\lambda) = \frac{1}{1+\exp(x\lambda)}$$

and can be easily extended to regression. The univariate *probit* model uses the cumulative distribution function (CDF) Φ of a univariate normal with zero mean and unit variance instead:

$$P_{PB}(x) = \Phi(xb) = \int_a^b \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2} \mathrm{d}t, \tag{4.19}$$

where

$$(a,b) = \begin{cases} (-\infty, \beta) & \text{for } x = 1, \\ (\beta, \infty) & \text{for } x = -1. \end{cases}$$

For the multivariate case, we can model an M-variate binary $\boldsymbol{X}$ using a zero-mean Gaussian with a unit-diagonal covariance matrix $\boldsymbol{\Sigma}$, so for a probit model, the covariance matrix of the underlying Gaussian is its correlation matrix. Let ρ_{ij} be the correlation between X_i and X_j. Vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_M)$ determines the boundaries of the orthants:

$$P_{PB}(\boldsymbol{x}) = \int_{a_1}^{b_1} \cdots \int_{a_M}^{b_M} \frac{1}{(2\pi)^{\frac{M}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}\boldsymbol{t}'\boldsymbol{\Sigma}^{-1}\boldsymbol{t}} \mathrm{d}t_1 \ldots \mathrm{d}t_M$$

[11] It is more convenient for this chapter to use $\{-1,1\}$ for the binary domain rather than $\{0,1\}$ used elsewhere in the thesis.

where

$$(a_i, b_i) = \begin{cases} (-\infty, \beta_i) & \text{for } x_i = 1, \\ (\beta_i, \infty) & \text{for } x_i = -1. \end{cases}$$

It is not hard to show that P_{PB} is a probability distribution on $\{-1, 1\}$. Intuitively, the probability distribution of a multivariate probit model can be interpreted in the following manner. An M-dimensional real space $\mathbb{R}^M$ is split into 2^M orthants (corners) by M hyperplanes $x_i = \beta_i$. Region $(-\infty, \beta_i)$ of axis i corresponds to value 1 for x_i; region $[\beta_i, \infty)$ corresponds to -1. The probability value for each of 2^M values is determined by the probability mass of the normal with zero mean and covariance matrix with unit diagonal in the respective orthant. Figure 4.11 contains an illustration of a bivariate probit distribution.

As a general property of a normal distribution, any orthogonal projection is also a normal distribution. For any ordered subset of indices $(i_1, \ldots, i_L)$, $L \leq M$,

$$P_{PB}(x_{i_1}, \ldots, x_{i_L}) = \int_{a_{i_1}}^{b_{i_1}} \cdots \int_{a_{i_L}}^{b_{i_L}} \frac{1}{(2\pi)^{\frac{L}{2}} |\mathbf{\Sigma}_L|^{\frac{1}{2}}} e^{-\frac{1}{2}\boldsymbol{t}'\mathbf{\Sigma}_L^{-1}\boldsymbol{t}} \mathrm{d}t_{i_1} \ldots \mathrm{d}t_{i_L}$$

where $\mathbf{\Sigma}_L$ is a submatrix of $\mathbf{\Sigma}$ obtained by keeping rows and columns with indices $(i_1, \ldots, i_L)$, and $\boldsymbol{t}$ here is a L-dimensional vector. Each univariate marginal has the form of (4.19)

$$P_{PB}(x_i) = \int_{a_i}^{b_i} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t_i^2} \mathrm{d}t_i, \tag{4.20}$$

while each bivariate marginal is a two-dimensional normal integral

$$P_{PB}(x_i, x_j) = \int_{a_i}^{b_i} \int_{a_j}^{b_j} \frac{1}{2\pi\sqrt{1-\rho_{ij}^2}} e^{-\frac{1}{2\left(1-\rho_{ij}^2\right)}\left(t_i^2 - 2\rho_{ij}t_it_j + t_j^2\right)} \mathrm{d}t_i \mathrm{d}t_j. \tag{4.21}$$

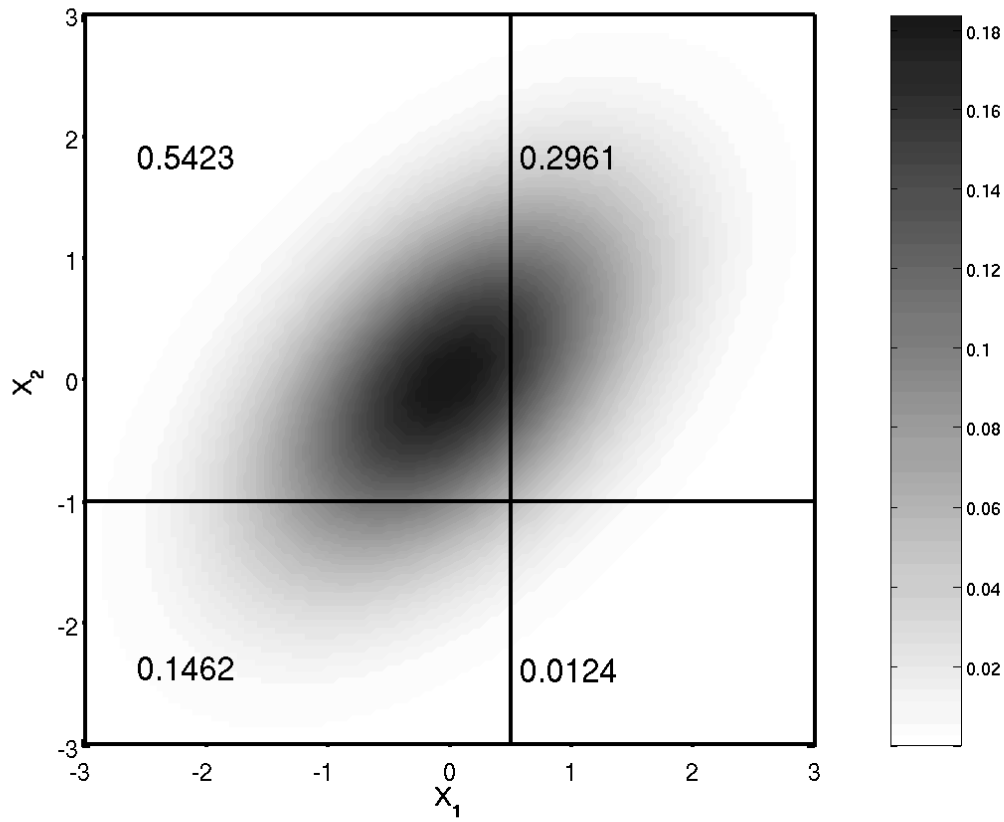

Figure 4.11: Bivariate probit distribution with $\beta_1 = 0.5$, $\beta_2 = -1$, and correlation $\sigma_{12} = 0.5$. The probabilities for each of four pairs of values are equal to the probability mass of the Gaussian in the corresponding quadrant: $P(X_1 = 1, X_2 = 1) = 0.1462$, $P(X_1 = 1, X_2 = -1) = 0.5423$, $P(X_1 = -1, X_2 = 1) = 0.0124$, and $P(X_1 = -1, X_2 = -1) = 0.2961$.

We are interested in solving the following problem: given a distribution $P(\boldsymbol{x})$ on $\{-1, 1\}^M$, we want to find a distribution $P_{PB}(\boldsymbol{x})$ matching all bivariate marginals of P. To find the parameters $(\beta_1, \ldots, \beta_M)$ and $(\rho_{12}, \ldots, \rho_{M-1,M})$, we first need to solve M independent equations

$$P_{PB}(X_i = 1) = \int_{-\infty}^{\beta_i} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t_i^2} \mathrm{d}t_i = P(X_i = 1) \tag{4.22}$$

to find β_i, and then to solve $M\left(M-1\right)/2$ independent equations

$$\begin{aligned} P_{PB}\left(X_i=1, X_j=1\right) &= \int_{-\infty}^{\beta_i}\int_{-\infty}^{\beta_j}\frac{1}{2\pi\sqrt{1-\rho_{ij}^2}}e^{-\frac{1}{2\left(1-\rho_{ij}^2\right)}\left(t_i^2-2\rho_{ij}t_it_j+t_j^2\right)}\mathrm{d}t_i\mathrm{d}t_j \quad (4.23)\\ &= P\left(X_i=1, X_j=1\right), \quad 1\le i,j\le M,\ i\ne j \end{aligned}$$

to find ρ_{ij}. Equation 4.22 can be solved by taking the inverse of Φ (e.g., Wichura, 1988). Equation 4.23 can be solved as an inverse problem of finding the rectangular bivariate normal, an integral of a normal distribution over a rectangular region with boundaries parallel to the coordinate planes; rectangular bivariate normals, in turn, can be computed with high accuracy (e.g., Genz, 2004).

To sample from $P_{PB}\left(\boldsymbol{x}\right)$, we first sample $\boldsymbol{Z}\in\mathbb{R}^M$ from the M-variate normal $\mathcal{N}\left(\mathbf{0},\boldsymbol{\Sigma}\right)$. Then $\boldsymbol{X}=\left(X_1,\ldots,X_M\right)$ is determined by

$$X_i=\begin{cases}1 & z_i<\beta_i,\\ -1 & z_i\ge\beta_i.\end{cases}$$

We can employ multivariate probit models to model conditional distributions as well. For $\boldsymbol{y}\in\mathbb{R}^M$, we can employ generalized linear models (e.g., McCullagh and Nelder, 1989) to model $P_{PB}\left(\boldsymbol{x}|\boldsymbol{y}\right)$. $\boldsymbol{y}$ can be incorporated into the model by estimating a matrix $\boldsymbol{\Gamma}$ such that $\boldsymbol{\beta}=\Gamma\boldsymbol{y}$ instead of estimating $\boldsymbol{\beta}$.

There are a number of problems with using multivariate probit models to approximate multivariate binary distributions. First, not all multivariate binary distributions can be represented by a multivariate probit distribution; while all 2×2 submatrices of the correlation matrix obtained by selecting i-th and j-th rows and columns are positive definite, this does not guarantee that the matrix obtained by

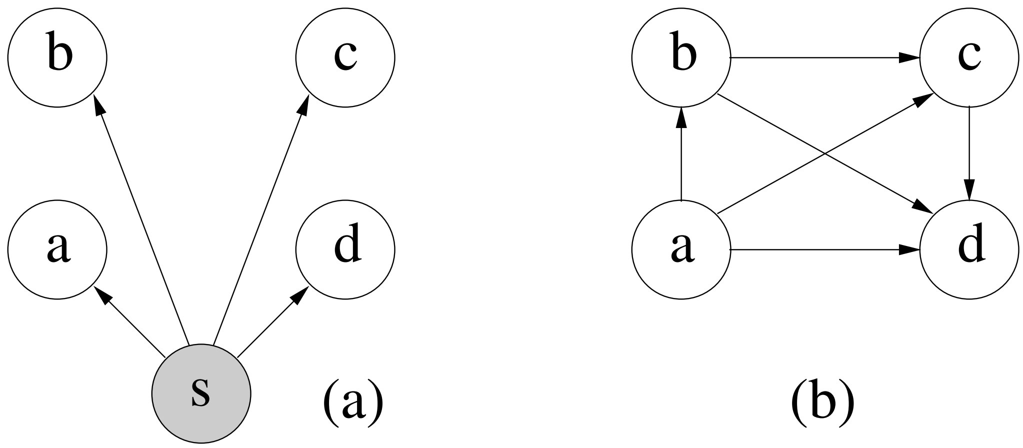

Figure 4.12: Example of potential benefits of latent variables. If a latent variable X_s is included in the model, the Bayesian network (a) of the distribution is sparser and requires fewer parameters than if X_s is ignored (network (b)).

solving equations 4.22 and 4.23 is positive definite. Second problem, the probit model does not generalize to domains with more than two values per dimension. Also, the probability of a vector under the probit model cannot be evaluated exactly except for a few special cases of the covariance matrix. To evaluate such a probability, we need to evaluate an M-dimensional rectangular normal integral. While this can be done accurately for small values of M (e.g., Genz, 1993; Kotz et al., 2000), evaluation of the integral for larger values of M with good accuracy requires an exponential number of samples in M. Lastly, while a maximum likelihood estimator for multivariate probit models exists and is unique (Lesaffre and Kaufmann, 1992), the estimation cannot be efficiently performed.

4.4.2 Mixtures

As was mentioned in passing, adding hidden variables to the set of observed variables can sometimes lead to sparser conditional independence structure and, thus, to fewer free parameters overall. (See Figure 4.12 for an illustration.) In this section, we consider modeling multivariate categorical distributions using nested finite mixtures. A *finite mixture model* for an M-variate categorical random variable $\boldsymbol{X}$ taking values on $\mathcal{X}$ consists of a latent variable Z taking on the values $\{1, \dots, K\}$, $K < \infty$, and K mutually independent M-variate categorical probability distributions $P_M^i(\boldsymbol{x})$, $i = 1, \dots, K$. The probability $P_M(\boldsymbol{x})$ under the mixture model is defined as

$$P_M(\boldsymbol{x}) = \sum_{i=1}^{K} P(Z = i)\, P_M^i(\boldsymbol{x}).$$

The advantage of mixtures is that they allow us to combine probability distributions with simple dependency structures to create probability distributions capable of capturing more complex dependency structures. For example, a class of mixtures of Chow-Liu trees is considerably more expressive than the class models with a single tree dependence structure (Meilă and Jordan, 2000). Also note that the idea of mixtures can be extended to allow not one latent variable, but a finite hierarchy of latent variables; in other words, the categorical probability distributions $P_i(\boldsymbol{x})$ are allowed to be mixtures themselves.

Finite mixture models require K parameters $\boldsymbol{\pi} = (\pi_1, \dots, \pi_K\}$ to specify $P(Z = i) = \pi_i$ otherwise called *mixing parameters*. Since $\sum_{i=1}^{K} \pi_i = 1$, only $K - 1$ of π_i are free parameters. Finite mixture models also require parameters $\boldsymbol{\zeta} = \{\boldsymbol{\zeta}_1, \dots, \boldsymbol{\zeta}_K\}$ with $\boldsymbol{\zeta}_i$ specifying $P_M^i(\boldsymbol{x})$. Given the parameters, the probability of a vector $\boldsymbol{X}$ under the mixture model can be expressed as

$$P_M(\boldsymbol{x}|\boldsymbol{\pi}, \boldsymbol{\zeta}) = \sum_{i=1}^{K} \pi_i P_M^i(\boldsymbol{x}|\boldsymbol{\zeta}_i).$$

Under the framework established earlier in this chapter, we want to learn the parameters $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{\zeta}\}$ to minimize $KL(P \parallel P_M)$ where $P(\boldsymbol{x})$ is the target (and often empirical) probability distribution on $\mathcal{X}$. This can be accomplished by using the iterative Expectation-Maximization (EM) algorithm (Dempster et al., 1977). In each iteration of the algorithm, the parameters $\boldsymbol{\theta}$ are changed to decrease the objective function $KL\left(P \parallel P_M\right)$ (or, alternatively, to increase the log-likelihood $l\left(\boldsymbol{\theta}\right) = \sum_{\boldsymbol{X}} P\left(\boldsymbol{x}\right) \ln P_M\left(\boldsymbol{x}|\boldsymbol{\theta}\right)$). Given parameters $\boldsymbol{\theta}$ and new parameters $\boldsymbol{\theta}'$, the difference in log-likelihood can be represented as a two component sum (similar to Equations 3.11 and 3.12):

$$\begin{aligned} l\left(\boldsymbol{\theta}'\right) - l\left(\boldsymbol{\theta}\right) &= \sum_{\boldsymbol{X}} P\left(\boldsymbol{x}\right) \sum_{i=1}^{K} P_M\left(Z = i|\boldsymbol{x}, \boldsymbol{\theta}\right) \ln \frac{P_M\left(Z = i, \boldsymbol{x}|\boldsymbol{\theta}'\right)}{P_M\left(Z = i, \boldsymbol{x}|\boldsymbol{\theta}\right)} \\ &\quad + KL\left(P_M\left(z|\boldsymbol{x}, \boldsymbol{\theta}\right) \parallel P_M\left(z|\boldsymbol{x}, \boldsymbol{\theta}'\right)\right). \end{aligned}$$

Since KL-divergence is always non-negative,

$$l\left(\boldsymbol{\theta}'\right) - l\left(\boldsymbol{\theta}\right) \geq \sum_{\boldsymbol{X}} P\left(\boldsymbol{x}\right) \sum_{i=1}^{K} P_M\left(Z = i|\boldsymbol{x}, \boldsymbol{\theta}\right) \ln \frac{P_M\left(Z = i, \boldsymbol{x}|\boldsymbol{\theta}'\right)}{P_M\left(Z = i, \boldsymbol{x}|\boldsymbol{\theta}\right)},$$

and $l\left(\boldsymbol{\theta}'\right) - l\left(\boldsymbol{\theta}\right)$ can be made non-negative by maximizing

$$\begin{aligned} Q\left(\boldsymbol{\theta}', \boldsymbol{\theta}\right) &= \sum_{\boldsymbol{X}} P\left(\boldsymbol{x}\right) \sum_{i=1}^{K} P_M\left(Z = i|\boldsymbol{x}, \boldsymbol{\theta}\right) \ln P_M\left(Z = i, \boldsymbol{x}|\boldsymbol{\theta}'\right) \\ &= E_P\left[E_{P_M\left(z|\boldsymbol{x}, \boldsymbol{\theta}\right)}\left[\ln P_M\left(Z = i, \boldsymbol{x}|\boldsymbol{\theta}'\right)\right]\right]. \end{aligned}$$

taking into the account constraints imposed on $\boldsymbol{\theta}'$. Each iteration consists of two steps: E-step, computing $P_M\left(Z = i|\boldsymbol{x}, \boldsymbol{\theta}\right) \propto \pi_i P_M^i\left(\boldsymbol{x}|\boldsymbol{\theta}\right)$ $\forall \boldsymbol{x} \in \mathcal{X}$ such that $P\left(\boldsymbol{x}\right) > 0$, and M-step, maximizing $Q\left(\boldsymbol{\theta}', \boldsymbol{\theta}\right)$ with respect to $\boldsymbol{\theta}'$. Under these update rules,

$$\pi_i' = \frac{\sum_{\boldsymbol{X}} P\left(\boldsymbol{x}\right) P_M\left(Z = i|\boldsymbol{x}, \boldsymbol{\theta}\right)}{\sum_{\boldsymbol{X}} \sum_{i=1}^{K} P\left(\boldsymbol{x}\right) P_M\left(Z = i|\boldsymbol{x}, \boldsymbol{\theta}\right)} = \sum_{\boldsymbol{X}} P\left(\boldsymbol{x}\right) P_M\left(Z = i|\boldsymbol{x}, \boldsymbol{\theta}\right). \qquad (4.24)$$

In order to find $\boldsymbol{\zeta}'$, we need to maximize for each $i = 1, \ldots, K$

$$\sum_{\boldsymbol{X}} P(\boldsymbol{x}) P_M(Z = i|\boldsymbol{x}, \boldsymbol{\theta}) \ln P_M^i(\boldsymbol{x}|\boldsymbol{\zeta}_i') = C \times \sum_{\boldsymbol{X}} P^i(\boldsymbol{x}) \ln P_M^i(\boldsymbol{x}|\boldsymbol{\zeta}_i') \tag{4.25}$$

where

$$C = \sum_{\boldsymbol{X}} P(\boldsymbol{x}) P_M(Z = i|\boldsymbol{x}, \boldsymbol{\theta}) \text{ and } P^i(\boldsymbol{x}) = \frac{1}{C} P(\boldsymbol{x}) P_M(Z = i|\boldsymbol{x}, \boldsymbol{\theta}). \tag{4.26}$$

Alternatively, for each $i = 1, \ldots, K$, we can minimize $KL(P^i \parallel P_M^i(\cdot|\boldsymbol{\zeta}_i'))$ which is the same form as the original objective function. In practice, $P(\boldsymbol{x})$ is never explicitly computed or stored. Instead, each of the of the examples in the original data set is assigned a weight, and this weight is computed anew for each P^i.

4.5 Summary

In this chapter we described models with different degrees of variable dependence for modeling of multivariate categorical data. Among the contributions of this chapter are several new models for multivariate conditional and joint distributions. Conditional Chow-Liu forests, an extension of Chow-Liu model, is an efficient and provably optimal in its class model for conditional multivariate distributions. Another new model, a Bayesian network of conditional maximum entropy models (PUC-MaxEnt) is an alternative to maximum entropy models for conditional and joint probability distributions. These and other models presented in this chapter can be used for emission probability distribution with HMMs as suggested in Section 3.4. Such hidden Markov models are considered further in Chapter 6.

Chapter 5

Models for Multivariate Real-Valued Distributions

This chapter is a real-valued counterpart to Chapter 4. The emphasis of the chapter is not to provide a complete overview of methods for modeling multivariate continuous-valued data, but to show how some of the models in Chapter 4 can be applied to real-valued data, and to draw contrasts and parallels between categorical and real-valued cases.

5.1 Independence

Univariate data is usually easier to model than multivariate data. Univariate distributions have been studied longer; they often have nicer computational properties than their multivariate extensions, and multivariate counterparts of certain univariate functional forms do not always exist. An easy approach to modeling multivariate data is to assume independence of individual variables. Assuming a functional form on each of the marginal probability density functions (PDFs) $p_I(x_i)$, let $\boldsymbol{\theta}_i$ denote the parameters needed to specify $p_I(x_i)$, and let us further assume that the $\boldsymbol{\theta}_i$'s are mutually independent. Let $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_M)$. Then

$$p_I(\boldsymbol{x}|\boldsymbol{\Theta}) = \prod_{i=1}^{M} p_I(x_i|\boldsymbol{\theta}_i).$$

If $p(\boldsymbol{x})$ is a(n) (empirical) PDF defined on $\mathbb{R}^M$, then

$$KL(p(\boldsymbol{x}) \parallel p_I(\boldsymbol{x}|\boldsymbol{\Theta})) = \sum_{i=1}^{M} KL(p(x_i) | p_I(x_i|\boldsymbol{\theta}_i)),$$

so finding the $\boldsymbol{\theta}_i$'s minimizing $KL(p(\boldsymbol{x}) \parallel p_I(\boldsymbol{x}|\boldsymbol{\Theta}))$ can be done by minimizing $KL(p(x_i) \parallel p_I(x_i|\boldsymbol{\theta}_i))$ independently.

5.1.1 Modeling Daily Rainfall Amounts

The functional form of the univariate PDF for each variable depends on the domain of the data. In the context of precipitation, daily rainfall amounts for a single location are often modeled using a mixture model consisting of (a) one component, a Dirac's delta function $\delta(x)$, for zero precipitation, and (b) several other components for non-zero precipitation. Non-zero daily amounts are often modeled by a K-component ($K \geq 1$) mixture of exponential distributions:

$$p_{amounts}(x) = \pi_0 \delta(x) + \sum_{k=1}^{K} \pi_k \alpha_k e^{-\alpha_k x},$$

where $\pi_0, \ldots, \pi_K$ are the mixing probabilities, and α_k's are the parameters of the exponential distributions. Sometimes, gamma distributions or negative binomial distributions are used instead of exponentials (Bellone, 2000).

5.2 Multivariate Normal Distributions

The multivariate normal distribution is the most studied and most commonly used probability distribution on multivariate real-valued domains. Among the reasons for its widespread use are intuition for parameter interpretation, ease of calculation of marginals, relative ease of parameter estimation.

Suppose that the domain is an M-dimensional space of real numbers $\mathbb{R}$. The probability density function for a normal distribution is defined in terms of a mean vector $\boldsymbol{\mu} \in \mathbb{R}^M$ and a symmetric positive definite covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{M \times M}$:

$$p_{\mathcal{N}}\left(\boldsymbol{x}\right) = \frac{1}{\sqrt{\left(2\pi\right)^M \left|\mathbf{\Sigma}\right|}} \exp\left(-\frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{\mu}\right)' \mathbf{\Sigma}^{-1}\left(\boldsymbol{x} - \boldsymbol{\mu}\right)\right).$$

It turns out that we can impose conditional independence structure on the variables of $\boldsymbol{X}$ by constraining the inverse covariance matrix $\mathbf{\Sigma}^{-1} = \mathbf{K} = \{k_{uv}\}_{u,v \in \mathcal{V}}$, often referred to as *precision* or *concentration* matrix. Zeros of the concentration matrix correspond to conditional independence relations of the variables (Lauritzen, 1996, Proposition 5.2):

$$k_{uv} = 0 \Longleftrightarrow p_{\mathcal{N}}\left(x_u, x_v | \boldsymbol{x}_{\mathcal{V}\setminus\{u,v\}}\right) = p_{\mathcal{N}}\left(x_u, | \boldsymbol{x}_{\mathcal{V}\setminus\{u,v\}}\right) p_{\mathcal{N}}\left(x_v | \boldsymbol{x}_{\mathcal{V}\setminus\{u,v\}}\right).$$

Thus, building a network of conditional independencies for a multivariate Gaussian is equivalent to selection of zero entries in the concentration matrix, and is often called *covariance selection*. Once the selection of zero entries in the concentration matrix is complete, the non-zero entries in the concentration matrix can either be found in closed form if the resulting model is decomposable (Lauritzen, 1996, Chapter 5.2), or by iterative proportional scaling (Speed and Kiiveri, 1986) if the model is not decomposable. For more details, see Lauritzen (1996); Cox and Wermuth (1996); Whittaker (1990).

In this section, we consider an approximation of a (possibly empirical) PDF $p(\mathbf{x})$ with a multivariate normal $p_{\mathcal{N}}(\mathbf{x})$ subject to minimization of the KL-divergence $KL(p \parallel p_{\mathcal{N}})$. We start out with a model with a full covariance matrix (a *saturated* model), and then consider a multivariate normal counterpart to Chow-Liu trees, a tree-structured multivariate normal distribution.

5.2.1 Unconstrained Normal Distribution

If the values of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are not constrained (except for the symmetric positive definite property of $\boldsymbol{\Sigma}$), the set of parameters $\boldsymbol{\upsilon} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ consists of $M(M+1)/2$ free parameters. Such an unconstrained model has the desirable property of preserving the correlation between any pair of variables. Also, if bivariate marginals of the distribution $p(\boldsymbol{x})$ are bivariate normals, then $p_{\mathcal{N}}(\boldsymbol{x})$ will match them exactly.

To learn the parameters we need to find $\boldsymbol{\nu}$ minimizing $KL(p \parallel p_{\mathcal{N}}(\cdot|\boldsymbol{\nu}))$, or alternatively, maximizing

$$
\begin{aligned}
Q &= \int_{\mathbb{R}^M} p(\boldsymbol{x}) \ln p_{\mathcal{N}}(\boldsymbol{x}|\boldsymbol{\upsilon}) \,\mathrm{d}\boldsymbol{x} \\
&= -\frac{1}{2} \int_{\mathbb{R}^M} p(\boldsymbol{x}) \left(M \ln(2\pi) - \ln|\mathbf{K}| + (\boldsymbol{x}-\boldsymbol{\mu})' \mathbf{K} (\boldsymbol{x}-\boldsymbol{\mu}) \right) \mathrm{d}\boldsymbol{x}.
\end{aligned}
$$

By taking the partial derivative of Q with respect to $\boldsymbol{\mu}$ we obtain

$$
\frac{\partial Q}{\partial \boldsymbol{\mu}} = \int_{\mathbb{R}^M} p(\boldsymbol{x}) \mathbf{K} (\boldsymbol{x}-\boldsymbol{\mu}) \,\mathrm{d}\boldsymbol{x} = \mathbf{K} \left(E_p(\boldsymbol{X}) - \boldsymbol{\mu} \right). \tag{5.1}
$$

By setting the derivative to $\mathbf{0}$, we obtain a maximum likelihood estimate $\hat{\boldsymbol{\mu}} = E_p(\mathbf{X})$. Before taking the partial derivative with respect to $\mathbf{K}$, we note that for any $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{M \times M}$,

$$\frac{\mathrm{d}\,|\mathbf{A}|}{\mathrm{d}\mathbf{A}} = 2\mathbf{A}^{-1} - diag\left(\mathbf{A}^{-1}\right) \text{ and } \frac{\mathrm{d}tr\left(\mathbf{AB}\right)}{\mathrm{d}\mathbf{A}} = \mathbf{B} + \mathbf{B}^t - diag\left(\mathbf{B}\right).$$

Then noticing that $(\boldsymbol{x} - \boldsymbol{\mu})' \mathbf{K} (\boldsymbol{x} - \boldsymbol{\mu}) = tr\left(\mathbf{K}(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})'\right)$, we find

$$\begin{aligned}
\frac{\partial Q}{\partial \mathbf{K}} &= \frac{1}{2}\int_{\mathbb{R}^M} p(\boldsymbol{x}) \left(\frac{\partial \ln |\mathbf{K}|}{\partial \mathbf{K}} - \frac{\partial}{\partial \mathbf{K}} tr\left(\mathbf{K}(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})'\right)\right) \mathrm{d}\boldsymbol{x} \\
&= \frac{1}{2}\int_{\mathbb{R}^M} p(\boldsymbol{x}) \left(2\boldsymbol{\Sigma} - diag\left(\boldsymbol{\Sigma}\right)\right) \mathrm{d}\boldsymbol{x} \\
&\quad -\frac{1}{2}\int_{\mathbb{R}^M} p(\boldsymbol{x}) \left(2(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^t - diag\left((\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^t\right)\right) \mathrm{d}\boldsymbol{x}.
\end{aligned}$$

By setting $\frac{\partial Q}{\partial \mathbf{K}}$ to zero, and by using $\hat{\boldsymbol{\mu}} = E_p(\boldsymbol{X})$, we find that

$$\hat{\boldsymbol{\Sigma}} = Cov_p(\boldsymbol{X}) = \int_{\mathbb{R}^M} p(\boldsymbol{x}) (\boldsymbol{x} - E_p(\boldsymbol{X})) (\boldsymbol{x} - E_p(\boldsymbol{X}))' \mathrm{d}\boldsymbol{x}.$$

Assuming p is an empirical distribution collected from N samples, the parameters of the saturated model can be obtained in time $O(NM + NM^2) = O(NM^2)$ with storage requirements of $O(M^2)$. Curiously, the corresponding problem for M-variate binary data, finding parameters of a distribution from an exponential family matching all bivariate marginals, would be much more intensive computationally. Not only there is no closed form solution for multivariate categorical data, this problem is usually solved using iterative scaling (see Section 4.3), and each iteration requires the computation of $P(x_v)$ and $P(x_u, x_v)$ for some pair $u, v \in \mathcal{V}$. Unfortunately, for exponential models for binary data, there is no efficient way to compute these marginals. The exact computation requires the summation over all other variables of $\boldsymbol{X}$, an $O\left(2^{M-1}\right)$ time complexity.

In order to compute $p_{\mathcal{N}}(\boldsymbol{x}|\boldsymbol{\nu})$, one needs to compute $\mathbf{K}$ and $|\boldsymbol{\Sigma}|$ first, with time complexity $O(M^3)$[1], and each evaluation of $p_{\mathcal{N}}(\boldsymbol{x}|\boldsymbol{\nu})$ will require $O(M^2)$ operations. Moreover, if $N \leq M$, the resulting estimate $\hat{\boldsymbol{\Sigma}}$ would be singular and thus not positive definite (Lauritzen, 1996, Theorem 5.1). Also, when $\boldsymbol{\Sigma}$ is close to singular, the accuracy of the inverse operation is severely hindered by the round-off error.

To sample from $p_{\mathcal{N}}(\boldsymbol{x}|\boldsymbol{\nu})$, we first need to compute a Cholesky decomposition $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}'$ where $\mathbf{L}$ is a lower-triangular matrix. Then a random sample $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be obtained as $\boldsymbol{X} = \boldsymbol{\mu} + \mathbf{L}\boldsymbol{Y}$ where $Y \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_M)$, and $\mathbf{I}_M$ is the $M \times M$ identity matrix, an $O(M^2)$ operation for each sample.

By discovering conditional independence relations in the variables of $\boldsymbol{X}$,, otherwise known as covariance selection, we are able to reduce not only the number of samples needed to estimate $\boldsymbol{\Sigma}$ (Lauritzen, 1996, Proposition 5.9), but also the computational complexity of evaluation of and sampling from $p_{\mathcal{N}}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

5.2.2 Tree-Structured Normal Distributions

As in Section 4.2, we will use tree-structured dependency between variables to specify the normal distribution. As with Chow-Liu trees for categorical data, we want the approximating probability distribution to be represented as a normalized product of the bivariate probability distributions on its edges. Let $p_{\mathcal{N}CL}(\boldsymbol{x})$ denote a normal distribution with such tree structure. Assume the dependency tree structure $\mathcal{G}_T =$

[1]The inverse matrix could be computed with better asymptotic complexity by more involved algorithms or by algorithms using parallel computation.

$(\mathcal{V}, \mathcal{E}_T)$ is known. Then,

$$p_{\mathcal{N}CL}(\boldsymbol{x}) = \left(\prod_{v \in \mathcal{V}} p_{\mathcal{N}}(x_v)\right)\left(\prod_{(u,v)\in\mathcal{E}_T} \frac{p_{\mathcal{N}}(x_u, x_v)}{p_{\mathcal{N}}(x_u)\, p_{\mathcal{N}}(x_v)}\right).$$

For $v \in \mathcal{V}$, assume $p_{\mathcal{N}}(x_v)$ has mean μ_v and variance σ_v^2. For each edge $\{u, v\} \in \mathcal{E}_T$, assume $p_{\mathcal{V}}(x_u, x_v)$ has parameters

$$\boldsymbol{\mu}_{uv} = (\mu_u, \mu_v)' \text{ and } \boldsymbol{\Sigma}_{uv} = \begin{pmatrix} \sigma_u^2 & \sigma_u \sigma_v r_{uv} \\ \sigma_u \sigma_v r_{uv} & \sigma_v^2 \end{pmatrix}$$

where $r_{uv} \in (-1, 1)$. Then

$$\begin{aligned} & \frac{p_{\mathcal{N}}(x_u, x_v)}{p_{\mathcal{N}}(x_u)\, p_{\mathcal{N}}(x_v)} \\ = \quad & \frac{1}{\sqrt{1 - r_{uv}^2}} \exp\left(-\frac{r_{uv}^2 \left(\frac{x_u - \mu_u}{\sigma_u}\right)^2 + r_{uv}^2 \left(\frac{x_v - \mu_v}{\sigma_v}\right)^2 - 2 r_{uv} \left(\frac{x_u - \mu_u}{\sigma_u}\right)\left(\frac{x_v - \mu_v}{\sigma_v}\right)}{2\left(1 - r_{uv}^2\right)}\right). \end{aligned}$$

$p_{\mathcal{N}CL}(\boldsymbol{x})$ is an M-variate Gaussian,

$$\begin{aligned} p_{\mathcal{N}CL}(\boldsymbol{x}) &= \prod_{v \in \mathcal{V}} p_{\mathcal{N}}\left(\boldsymbol{x}_v | \mu_v, \sigma_v^2\right) \prod_{\{u,v\} \in \mathcal{E}_T} \frac{p_{\mathcal{N}}(x_u, x_v | \boldsymbol{\mu}_{uv}, \boldsymbol{\Sigma}_{uv})}{p_{\mathcal{N}}(\boldsymbol{x}_u | \mu_u, \sigma_u^2)\, p_{\mathcal{N}}(\boldsymbol{x}_v | \mu_v, \sigma_v^2)} \\ &= \left((2\pi)^M \prod_{v \in \mathcal{V}} \sigma_v^2 \prod_{\{u,v\} \in \mathcal{E}_T} \left(1 - r_{uv}^2\right)\right)^{-\frac{1}{2}} \\ &\quad \times \exp\left(-\frac{1}{2} \sum_{u \in \mathcal{V}} \sum_{v \in \mathcal{V}} K_{uv} (x_u - \mu_u)(x_v - \mu_v)\right), \end{aligned}$$

with mean $\boldsymbol{\mu}_{CL} = \{\mu_v\}_{v\in\mathcal{V}}$ and covariance matrix $\boldsymbol{\Sigma}_{CL}$ such that $\boldsymbol{\Sigma}^{-1} = \mathbf{K}_{CL} = \{K_{uv}\}_{u,v\in\mathcal{V}}$ can be computed as

$$K_{uv} = \begin{cases} \frac{1}{\sigma_{vv}^2}\left(1+\sum_{u:\{u,v\}\in\mathcal{E}_T}\frac{r_{uv}^2}{1-r_{uv}^2}\right) & u=v, \\ -\frac{r_{uv}}{|\sigma_u\sigma_v|(1-r_{uv}^2)} & \{u,v\}\in\mathcal{E}_T, \\ 0 & \{u,v\}\notin\mathcal{E}_T \text{ and } u\neq v. \end{cases}$$

The covariance matrix $\boldsymbol{\Sigma}_{CL}$ can be efficiently computed in closed form. Let $\mathbf{R}_{CL}$ be the correlation matrix of $p_{\mathcal{N}-CL}$, so $\boldsymbol{\Sigma}_{CL} = \mathbf{V}\mathbf{R}_{CL}\mathbf{V}$ where $\mathbf{V}$ is a diagonal matrix with entries $V_{vv} = |\sigma_{vv}|$, and $\mathbf{R}_{CL}^{-1} = \mathbf{V}\mathbf{K}_{CL}\mathbf{V}$.[2] We will utilize the tree property of $\mathcal{G}_T$: for any pair of distinct vertices $(u,v)\in\mathcal{V}\times\mathcal{V}$, there is a unique path $\mathcal{P}_{uv} = (u,\ldots,v)$ connecting u and v. Let $|\mathcal{P}_{uv}|$ be the number of nodes in the path $\mathcal{P}_{uv}$, and let $\mathcal{P}_{uv}(i)$ be the i-th node on the path $\mathcal{P}_{uv}$. We are now ready to compute the correlation matrix $\mathbf{R}_{CL}$.

Theorem 5.1. *The correlation matrix $\mathbf{R}_{CL} = \{r_{uv}\}_{u,v\in\mathcal{V}}$ of a tree-structured multivariate normal distribution $p_{\mathcal{N}CL}(\mathbf{x})$ is*

$$r_{uv} = \begin{cases} 1 & u=v, \\ \prod_{i=1}^{|\mathcal{P}_{uv}|-1} r_{\mathcal{P}_{uv}(i)\mathcal{P}_{uv}(i+1)} & u\neq v. \end{cases} \tag{5.2}$$

Proof. Let $\mathbf{A} = \{a_{uv}\}_{u,v\in\mathcal{V}}$ be a matrix with $a_{uv} = r_{uv}$ defined as in (5.2), and let $\mathbf{B} = \{b_{uv}\}_{u,v\in\mathcal{V}} = \mathbf{R}_{CL}^{-1}\mathbf{A}$. Since both $\mathbf{R}_{CL}^{-1}$ and $\mathbf{A}$ are symmetric, $\mathbf{A}\mathbf{R}_{CL}^{-1} = \mathbf{R}_{CL}^{-1}\mathbf{A} = \mathbf{B}$. We will show that $\mathbf{B} = \mathbf{I}_M$ which would mean that $\mathbf{A} = \mathbf{R}_{CL}$.

Let $\mathbf{R}_{CL}^{-1} = \{r'_{uv}\}_{u,v,\in\mathcal{V}}$ where $\mathbf{R}_{CL}^{-1} = \mathbf{V}\mathbf{K}_{CL}\mathbf{V}$. First, we compute the entries on the diagonal of $\mathbf{B}$. Let $di(u) = \mathcal{V}\setminus(\{u\}\cup ne(u))$ be the set of vertices not adjacent

[2] We assume $|\mathbf{V}| > 0$.

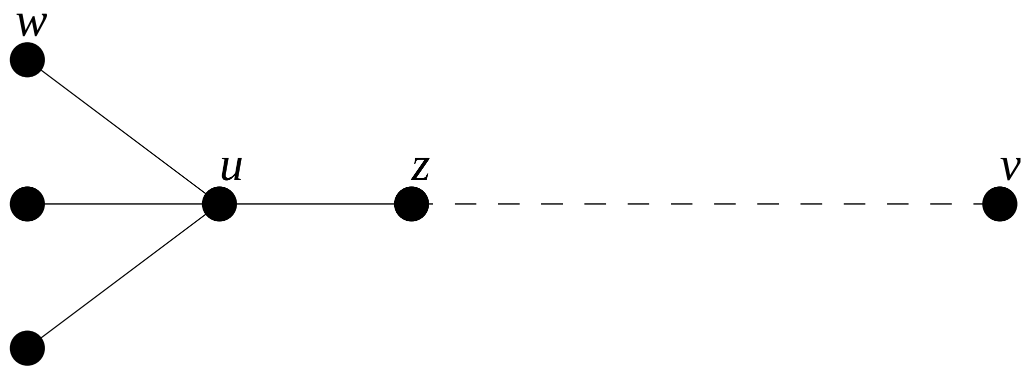

Figure 5.1: Illustration of the unique path property of the tree. For two nodes $u, v \in \mathcal{V}$, there is a unique neighbor z of u on the path from u to v.

to u (i.e., distant). Then

$$\begin{aligned}
b_{uu} &= \sum_{v\in\mathcal{V}} r'_{uv}a_{uv} = r'_{uu}a_{uu} + \sum_{v\in ne(u)} r'_{uv}a_{vu} + \sum_{v\in di(u)} r'_{uv}a_{vu} \\
&= \left(1 + \sum_{v\in ne(u)} \frac{r_{uv}^2}{1-r_{uv}^2}\right) + \sum_{v\in ne(u)} \left(-\frac{r_{uv}}{1-r_{uv}^2}\right) r_{uv} + \sum_{v\in di(u)} 0\times a_{vu} = 1.
\end{aligned}$$

Non-diagonal entries are somewhat trickier to compute:

$$\begin{aligned}
b_{uv} &= \sum_{w\in\mathcal{V}} r'_{uw}a_{wv} = r'_{uu}a_{uv} + \sum_{w\in ne(u)} r'_{uw}a_{wv} + \sum_{w\in di(u)} r'_{uw}a_{wv} \\
&= \left(1 + \sum_{w\in ne(u)} \frac{r_{uw}^2}{1-r_{uw}^2}\right) a_{uv} + \sum_{w\in ne(u)} \left(-\frac{r_{uw}}{1-r_{uw}^2}\right) a_{wv} + \sum_{w\in di(u)} 0\times a_{wv}.
\end{aligned}$$

By the property of the tree, when $u \neq v$, there is exactly one path $\mathcal{P}_{uv}$ connecting u and v. Among the neighbors $ne(u)$, there is a unique node belonging to $\mathcal{P}_{uv}$. (Existence of more than one neighbor of the starting point belonging to the path would result in a cycle.) We will denote that node by $z = \mathcal{P}_{uv}(2)$. (See Figure 5.1.)

Then

$$a_{uv} = \prod_{i=1}^{|\mathcal{P}_{uv}|-1} r_{\mathcal{P}_{uv}(i)\mathcal{P}_{uv}(i+1)} = r_{\mathcal{P}_{uv}(1)\mathcal{P}_{uv}(2)} \left(\prod_{i=2}^{|\mathcal{P}_{uv}|-1} r_{\mathcal{P}_{uv}(i)\mathcal{P}_{uv}(i+1)} \right) = \rho_{uz}\zeta_{zv}.$$

For other neighbors of u, a path from them to v consists of the edge to u and the path from u to v (see Figure 5.1):

$$\begin{aligned}
\forall w \in ne\,(u) \setminus \{z\}\,,\ \mathcal{P}_{wv} &= (w, u, \ldots, v)\,, \\
\mathcal{P}_{wv}\,(i) &= \begin{cases} w & i = 1, \\ \mathcal{P}_{uv}\,(i-1) & i = 2, \ldots, |\mathcal{P}_{uv}| + 1, \end{cases} \\
a_{wv} &= r_{wu} a_{uv}.
\end{aligned}$$

Going back to b_{uv},

$$\begin{aligned}
b_{uv} &= \left(1 + \sum_{w \in ne(u)} \frac{r_{uw}^2}{1 - \rho_{uw}^2}\right) a_{uv} - \sum_{w \in ne(u)} \frac{r_{uw}}{1 - r_{uw}^2} a_{wv} \\
&= \left(1 + \sum_{w \in ne(u)} \frac{r_{uw}^2}{1 - r_{uw}^2}\right) a_{uv} - \frac{r_{uz} a_{zv}}{1 - r_{uz}^2} + \sum_{w \in ne(u)\setminus\{z\}} \frac{r_{uw} a_{wv}}{1 - r_{uw}^2} \\
&= \left(1 + \sum_{w \in ne(u)} \frac{r_{uw}^2}{1 - r_{uw}^2}\right) a_{uv} - \frac{a_{uv}}{1 - r_{uz}^2} + \sum_{w \in ne(u)\setminus\{z\}} \frac{r_{uw}^2 a_{uv}}{1 - r_{uw}^2} \\
&= a_{uv} \left(1 + \sum_{w \in ne(u)} \frac{r_{uw}^2}{1 - r_{uw}^2} - \frac{1}{1 - r_{uz}^2} + \sum_{w \in ne(u)\setminus\{z\}} \frac{r_{uw}^2}{1 - r_{uw}^2}\right) \\
&= a_{uv} \left(1 + \frac{r_{uz}^2}{1 - r_{uz}^2} - \frac{1}{1 - r_{uz}^2}\right) = 0.
\end{aligned}$$

Thus $\mathbf{B} = \mathbf{I}_M$, and $\mathbf{A} = \mathbf{R}_{CL}$. □

The result can be extended to forests as well. To see this, we can connect the trees in the forest into one tree with added edges having correlations set to 0.

For each $v \in \mathcal{V}$, we can convert $\mathcal{G}_T$ into a directed tree with root v. Then $p_{\mathcal{N}CL}(\boldsymbol{x})$ can be rewritten as

$$
\begin{aligned}
p_{\mathcal{N}CL}(\boldsymbol{x}) &= p_{\mathcal{N}}(x_v) \prod_{\{u,pa(u)\}\in\mathcal{E}_T} \frac{p_{\mathcal{N}}\left(x_u, x_{pa(u)}\right)}{p_{\mathcal{N}}\left(x_{pa(u)}\right)} = p_{\mathcal{N}}(x_v) \prod_{u\in\mathcal{V}\setminus v} p_{\mathcal{N}}\left(x_u | x_{pa(u)}\right), \\
p_{\mathcal{N}}\left(x_u | x_{pa(u)}\right) &= \frac{1}{\sqrt{2\pi\sigma^2_{u|pa(u)}}} \exp\left(-\frac{1}{2}\left(\frac{x_u - \mu_{u|pa(u)}}{\sigma_{u|pa(u)}}\right)^2\right), \\
\sigma^2_{u|pa(u)} &= \sigma^2_u\left(1 - r^2_{u,pa(u)}\right), \ \mu_{u|pa(u)} = \mu_u - \frac{r_{uv}\sigma_u(x_v - \mu_v)}{\sigma_v}.
\end{aligned}
$$

The objective function, the KL-divergence between the target distribution and the tree-structured normal, can be written as

$$
\begin{aligned}
KL(p \parallel p_{\mathcal{N}CL}) &= \int_{\mathbb{R}^M} p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{p_{\mathcal{N}}(x_v) \prod_{u\in\mathcal{V}\setminus v} p_{\mathcal{N}}\left(x_u | x_{pa(u)}\right)} \mathrm{d}\boldsymbol{x} \\
&= KL\left(p(x_v) \parallel p_{\mathcal{N}}(x_v)\right) \\
&\quad + \sum_{u\in\mathcal{V}\setminus v} KL\left(p\left(x_u | x_{pa(u)}\right) \parallel p_{\mathcal{N}}\left(x_u | x_{pa(u)}\right)\right).
\end{aligned}
$$

Each of the KL-divergences can be optimized independently from others. The resulting maximum likelihood estimate of the mean parameters is $\hat{\boldsymbol{\mu}}_{CL} = E_p(\boldsymbol{X})$; estimates of univariate variances are $\hat{\sigma}^2_v = Var_p(X_v)$ for $v \in \mathcal{V}$, and correlations for edge variables $\hat{r}_{uv} = \rho_{uv} = \frac{E_p(X_u-\mu_u)(X_v-\mu_v)}{\sqrt{\sigma^2_u\sigma^2_v}}$. The rest of the entries in the correlation matrix can be found using the expression of the Theorem 5.1.

5.2.3 Learning Structure of Tree-Structured Normal Distributions

We are now interested in finding a set of edges $\mathcal{E}_T$ for the Chow-Liu tree to minimize

$$\begin{aligned} KL\left(p \parallel p_{\mathcal{N}CL}\right) &= \int_{\mathbb{R}^M} p\left(\boldsymbol{x}\right) \ln p\left(\boldsymbol{x}\right) \mathrm{d}\boldsymbol{x} \\ &\quad - \int_{\mathbb{R}^M} p\left(\boldsymbol{x}\right) \ln \left(\prod_{v \in \mathcal{V}} p_{\mathcal{N}}\left(x_v\right) \prod_{(u,v) \in \mathcal{E}_T} \frac{p_{\mathcal{N}}\left(x_u, x_v\right)}{p_{\mathcal{N}}\left(x_u\right) p_{\mathcal{N}}\left(x_v\right)} \right) \mathrm{d}\boldsymbol{x} \\ &= -H_p\left[\boldsymbol{X}\right] - \sum_{v \in \mathcal{V}} \int_{\mathbb{R}} p\left(x_v\right) \ln p_{\mathcal{N}}\left(x_v\right) \mathrm{d}x_v \\ &\quad - \sum_{(u,v) \in \mathcal{E}_T} \int_{\mathbb{R}} \int_{\mathbb{R}} p\left(x_u, x_v\right) \ln \frac{p_{\mathcal{N}}\left(x_u, x_v\right)}{p_{\mathcal{N}}\left(x_u\right) p_{\mathcal{N}}\left(x_v\right)} \mathrm{d}x_u \mathrm{d}x_v. \end{aligned} \tag{5.3}$$

Expression (5.3) is independent of $\mathcal{E}_T$, so to find the set of edges $\mathcal{E}_T$, we need to minimize (5.3) (or to maximize its negative). Let

$$MI_p\left(x_u, x_v\right) = \int_{\mathbb{R}} \int_{\mathbb{R}} p\left(x_u, x_v\right) \ln \frac{p_{\mathcal{N}}\left(x_u, x_v\right)}{p_{\mathcal{N}}\left(x_u\right) p_{\mathcal{N}}\left(x_v\right)} \mathrm{d}x_u \mathrm{d}x_v. \tag{5.4}$$

Just as with the categorical case, to minimize $KL\left(p \parallel p_{\mathcal{N}CL}\right)$, we need to find tree edges $\mathcal{E}_T$ maximizing $\sum_{\{u,v\} \in \mathcal{E}_T} MI_p\left(x_u, x_v\right)$ which is equivalent to solving a maximum spanning tree problem for a complete graph with nodes $\mathcal{V}$ and weights $MI_p\left(x_u, x_v\right)$ for edges $\{u, v\}$. It turns out that the mutual information MI_p depends only on the correlation coefficient ρ_{uv}. First, we simplify the expression inside the log:

$$\begin{aligned} &\frac{p_{\mathcal{N}}\left(x_u, x_v\right)}{p_{\mathcal{N}}\left(x_u\right) p_{\mathcal{N}}\left(x_v\right)} \\ &= \frac{1}{\sqrt{1 - \hat{r}_{uv}^2}} \exp\left(- \frac{\hat{r}_{uv}^2 \left(\frac{x_u - \hat{\mu}_u}{\hat{\sigma}_u}\right)^2 + \hat{r}_{uv}^2 \left(\frac{x_v - \hat{\mu}_v}{\hat{\sigma}_v}\right)^2 - 2\hat{r}_{uv} \left(\frac{x_u - \hat{\mu}_u}{\hat{\sigma}_u}\right)\left(\frac{x_v - \hat{\mu}_v}{\hat{\sigma}_v}\right)}{2\left(1 - \hat{r}_{uv}^2\right)} \right). \end{aligned} \tag{5.5}$$

Then from (5.4)-(5.5)

$$
\begin{aligned}
MI_p\left(x_u, x_v\right) = &-\frac{1}{2}\ln\left(1-\hat{r}^2\right) + \frac{\hat{r}_{uv}}{1-\hat{r}_{uv}^2} E_p\left[\left(\frac{X_u-\hat{\mu}_u}{\hat{\sigma}_u}\right)\left(\frac{X_v-\hat{\mu}_v}{\hat{\sigma}_v}\right)\right] \\
&-\frac{\hat{r}_{uv}^2}{2\left(1-\hat{r}_{uv}^2\right)}\left(E_p\left[\left(\frac{X_u-\hat{\mu}_u}{\hat{\sigma}_u}\right)^2\right] + E_p\left[\left(\frac{X_v-\hat{\mu}_v}{\hat{\sigma}_v}\right)^2\right]\right) \\
= &-\frac{1}{2}\ln\left(1-\rho_{uv}^2\right) + \frac{\rho_{uv}}{1-\rho_{uv}^2}\times\rho_{uv} - \frac{\rho_{uv}^2}{2\left(1-\rho_{uv}^2\right)}(1+1) \\
= &-\frac{1}{2}\ln\left(1-\rho_{uv}^2\right).
\end{aligned}
$$

Note that for $\rho_{uv} \in [0,1)$, $-\frac{1}{2}\ln\left(1-\rho_{uv}^2\right)$ is a strictly increasing function, so we can use $|\rho_{uv}|$ instead of $-\frac{1}{2}\ln\left(1-\rho_{uv}^2\right)$ for the MST problem. We should also note that the Chow-Liu algorithm chooses the edges to minimize the determinant of the covariance matrix:

$$
|\mathbf{\Sigma}_{CL}| = \min_{\mathcal{G}_T}\prod_{v\in\mathcal{V}}\sigma_v^2\prod_{\{u,v\}\in\mathcal{E}_T}\left(1-\rho_{uv}^2\right).
$$

In order to find the parameters of a tree structure, we need to compute correlations for all pairs of variables, $O\left(NM^2\right)$ computational task. Selecting the maximum spanning tree can be done in the same manner as for categorical data, by Kruskal's algorithm, $O\left(M^2\right)$ task (Cormen et al., 1990). Pelleg and Moore (2003) suggest accelerating the algorithm by approximating the correlations for low-correlation pairs while reducing computation by ignoring some of the data entries. Once the parameters are computed, the computation of $p_{\mathcal{N}CL}(\boldsymbol{x})$ can be performed in only $O(M)$ (as compared to $O\left(M^2\right)$ for the saturated model.) A generation of a single sample from $p_{\mathcal{N}CL}(\boldsymbol{x})$ would also require $O(M)$ as we do not need to compute the Cholesky decomposition of $\mathbf{\Sigma}_{CL}$ explicitly. Instead, we can convert $\mathcal{G}_T$ into a directed tree ($O(M)$), and then generate a sample for each variable given its predecessor using univariate conditional distributions.

5.3 Summary

In this chapter, we presented real-valued counterparts to several models for multivariate categorical data. We described conditionally independent mixtures of delta and exponential distributions, applicable to modeling of rainfall amounts. For contrast with full bivariate exponential distribution, we described commonly known normal distribution with full covariance and listed some of its properties. Finally, we reintroduced a multivariate normal distribution with tree-structured covariance matrix and described its advantages over the full covariance model. We also derived a closed form expression for the covariance matrix of the tree-structured normal distribution.

Tree-structured normal distributions were first described by Zarutskij (1979, 1980). While the model is very efficient, does not require large amount of data to train, and is easy to implement, it is rarely mentioned in the literature (e.g., Raudys, 2001). The closed form expression for the tree-structured covariance matrix has not been noted before although Whittaker (1990) considers the functional form of a correlation matrix with conditional independence with three variables.

Chapter 6

Putting It All Together: HMMs with Multivariate Emission Models

In Chapter 3, we described how HMMs can be thought of as a combination of models for the transition layer $P(\mathbf{S}|\mathbf{X})$ and the emission layer $P(\mathbf{R}|\mathbf{S})$ (e.g., Equation 3.5).

As discussed in Chapter 3, the set of HMM parameters $\boldsymbol{\Theta}$ consists of transition parameters $\boldsymbol{\Omega}$ and emission parameters $\boldsymbol{\Upsilon}$. Further, $\boldsymbol{\Upsilon}$ consists of parameters corresponding to each value of the hidden state variable S with $P(\boldsymbol{r}|S=i,\boldsymbol{r}_{previous},\boldsymbol{\Upsilon}) = P(\boldsymbol{r}|S=i,\boldsymbol{r}_{previous},\boldsymbol{\phi}_i,\boldsymbol{\upsilon}_i)$ where $\boldsymbol{\phi}_i$ are the parameters specifying the conditional independence structure for hidden state i, and $\boldsymbol{\upsilon}_i$ are the parameters of the probability distribution specified by $\boldsymbol{\phi}_i$. Structures $\boldsymbol{\phi}_i$ are fixed for some of the emission models (e.g., independence, full bivariate MaxEnt), and are learned for others (e.g., Chow-Liu trees, PUC-MaxEnt). Further, if the structure is learned, it can either be learned independently for different hidden states (i.e., it is allowed $\boldsymbol{\phi}_i \neq \boldsymbol{\phi}_j$ for $i \neq j$)

or constrained to be shared across all hidden states ($\boldsymbol{\phi}_i = \boldsymbol{\phi}_j$ for all i, j).[1] For the rest of the thesis, whenever the conditional independence structure for the states is learned, we assume that it is learned independently for all hidden states.

Parameter estimation is performed by the Baum-Welch algorithm as described in Section 3.3. Each iteration of the algorithm updates the parameters $\mathbf{\Theta}^r$ to $\mathbf{\Theta}^{r+1}$ maximizing the data log-likelihood from iteration r to $r+1$. Iterations consist of two steps: the E-step, computing posterior distributions $P\left(S_{nt}|\boldsymbol{R}_{n,1:T}, \mathbf{\Theta}^r\right)$ and $P\left(S_{nt}, S_{n,t-1}|\boldsymbol{R}_{n,1:T}, \mathbf{\Theta}^r\right)$; the M-step, estimating new parameters $\mathbf{\Theta}^{r+1}$. For the E-step of HMM parameter learning, the computational complexity is $O\left(NTK\left(K + R_{time}\right)\right)$, and the space complexity is $O\left(NTK^2 + R_{space}\right)$. Computation of $P\left(\boldsymbol{R}_{nt}|\boldsymbol{R}_{n,t-1}, \boldsymbol{S}_{nt}, \mathbf{\Theta}^r\right)$ has computational complexity $O\left(R_{time}\right)$ and space complexity $O\left(R_{space}\right)$, and is determined by the emission probability type. For the M-step, the estimation of the transition probabilities can be performed in $O\left(NTK^2\right)$ time and space[2]; the estimation of $\mathbf{\Upsilon}^{r+1}$ depends on the emission type.

In this chapter, we put together the hidden state models discussed in Chapter 3 with multivariate models discussed in Chapters 4 and 5, and compare their computational complexities and properties.

6.1 HMM with Conditional Independence

We will denote by HMM-CI (or NHMM-CI) a model which uses the independence model described in Section 4.1 for the emission distributions $P\left(\mathbf{r}|S\right)$.[3] The indepen-

[1] This choice can be thought of as a bias-variance trade-off (Bilmes, 2004).

[2] For NHMMs, estimation of the transition parameters depends on the dimensionality of the input variables and the optimization algorithm.

[3] Independence becomes conditional on the value of the hidden state S.

dence structures $\boldsymbol{\phi}_i$ are known (see Figure 6.1 (a) for the Bayesian network structure); a set $\boldsymbol{\upsilon}_i$ of emission parameters for state i consist of

$$\boldsymbol{\upsilon}_i = \left\{p_{imb} = P\left(R_{tm} = b | S_t = i\right) : m = 1, \ldots, M,\ b = 0, \ldots, B-1\right\}.$$

The total number of free emission parameters for an HMM-CI is $KM\left(B-1\right)$. As was shown in Section 3.4, in order to update the values of the emission parameters in the E-step of the EM-algorithm, we need to minimize the KL-divergence between $P_i\left(\mathbf{r}\right)$ as defined in Equation 3.19 and $P\left(\mathbf{r}|S=i, \boldsymbol{\upsilon}_i^{r+1}\right)$. Combining with (4.1), we get the following update rules for $\boldsymbol{\Upsilon}^{r+1}$:

$$\hat{p}_{imb} = P_i\left(r_m = b\right) = \frac{\sum_{r_{ntm}=b} W_{nt}\left(i\right)}{\sum_{n=1}^{N}\sum_{t=1}^{T} W_{nt}\left(i\right)} = \frac{\sum_{r_{ntm}=b} A_{nt}\left(i\right)}{\sum_{n=1}^{N}\sum_{t=1}^{T} A_{nt}\left(i\right)}$$

where $A_{nt}\left(i\right) = P\left(S_{nt} = i | \boldsymbol{r}_{n,1:T}, \boldsymbol{x}_{n,1:T}, \boldsymbol{\Theta}^r\right)$.

This basic model has two advantages over more complicated models: it has few parameters, and the M-step update of the emission parameters can be performed quickly, in $O\left(NTKMB\right)$. In addition, it often has a straightforward interpretation in the application domain. While this model does not explicitly model multivariate interactions between variables of $\boldsymbol{r}$ and between the variables at different time steps, it does so indirectly via hidden states. As the results in Chapter 8 show, this simple model can sometimes be as effective as more complex models while requiring only a fraction of parameters and training time. On the other hand, models that model variables interactions directly can often achieve better prediction accuracy and often obtain so with fewer hidden states (smaller K value) .

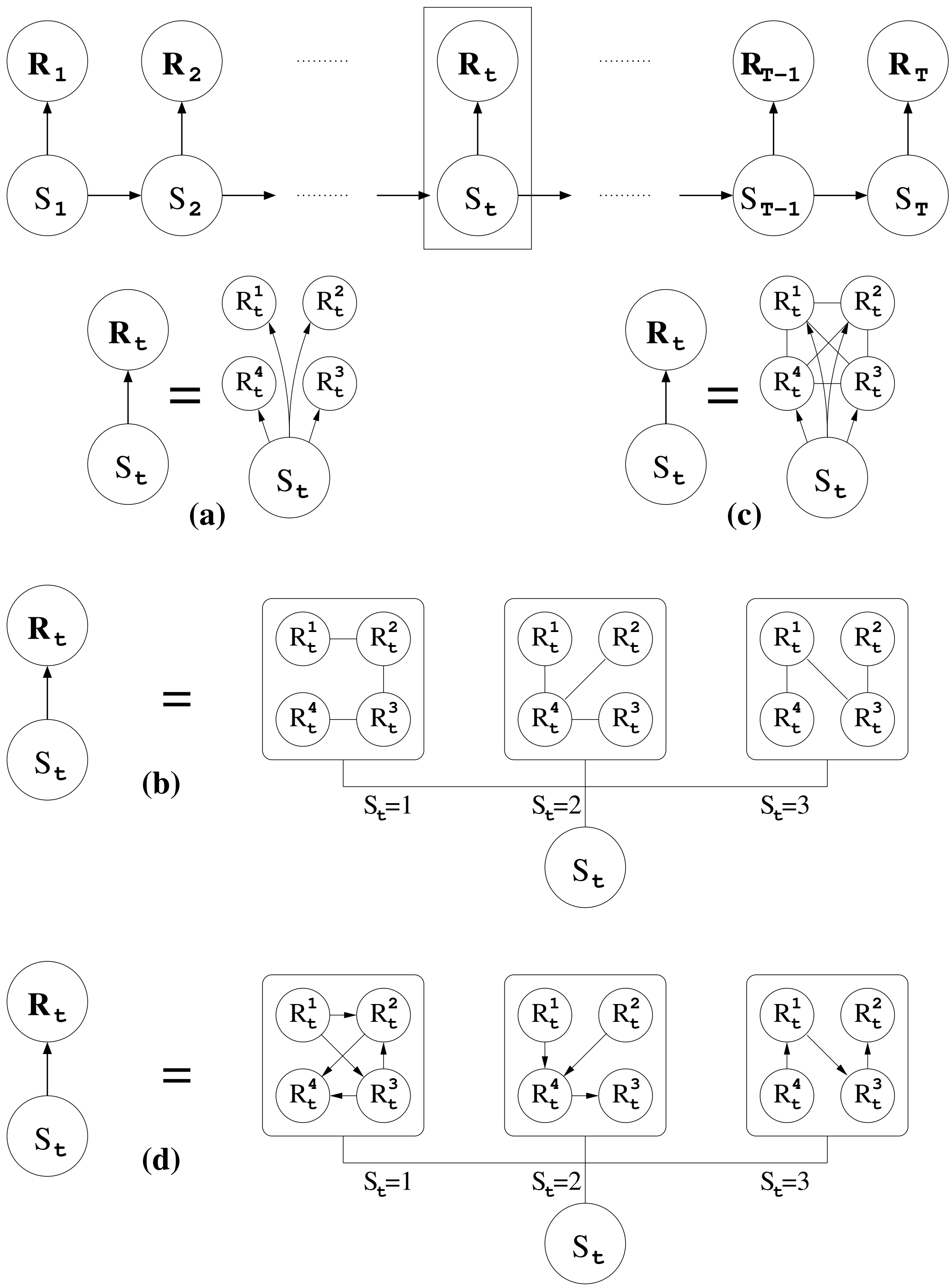

Figure 6.1: Bayesian network for a hypothetical (a) HMM-CI; (b) HMM-CL or HMM-CL-Normal; (c) HMM-Full-MaxEnt or HMM-Full-Normal; (d) HMM-PUC-MaxEnt

6.1.1 AR-HMM with Conditional Independence

By combining an autoregressive HMM with the model described in Equation 4.2 (and Figure 4.1, right) we obtain a model for which the Bayesian network is described by Figure 6.2 (a). We will denote this model by HMM-Chains. Just as with the HMM-CI, the conditional independence structures $\boldsymbol{\phi}_i$ are known. The set of emission parameters

$$\begin{aligned}\boldsymbol{\upsilon}_i = &\left\{p_{imbb'} = P\left(R_{tm} = b|S_t = i, R_{t-1,m} = b'\right) : m = 1,\ldots,M,\ b,b' = 0,\ldots,B-1\right\}\\ &\cup\left\{p_{imb} = P\left(R_{1m} = b|S_1 = i\right) : m = 1,\ldots,M,\ b = 0,\ldots,B-1\right\}\end{aligned}$$

has $M\left(B^2-1\right)$ free parameters for a total $KM\left(B^2-1\right)$ of free emission parameters. The E-step update of the parameters is as straightforward as for HMM-CI:

$$\begin{aligned}\hat{p}_{imbb'} = P_i\left(r_m = b|r_{prev,m}\right) &= \frac{\sum_{\substack{r_{ntm}=b\\ r_{n,t-1,m}=b'}} W_{nt}\left(i\right)}{\sum_{r_{n,t-1,m}=b'} W_{nt}\left(i\right)} = \frac{\sum_{\substack{r_{ntm}=b\\ r_{n,t-1,m}=b'}} A_{nt}\left(i\right)}{\sum_{r_{n,t-1,m}=b'} A_{nt}\left(i\right)}\\ \text{and } \hat{p}_{imb} &= \frac{\sum_{r_{n1m}=b} A_{n1}\left(i\right)}{\sum_{n=1}^{N} A_{n1}\left(i\right)}.\end{aligned}$$

If there are only a small number of series available, the probabilities of the first entry can be estimated from the whole data set:

$$\hat{p}_{imb} = \frac{\sum_{r_{ntm}=b} A_{nt}\left(i\right)}{\sum_{n=1}^{N}\sum_{t=1}^{T} A_{nt}\left(i\right)}.$$

The E-step update of the emission parameters can be performed in $O\left(NTKMB^2\right)$ time, almost the same as for HMM-CI for small B.

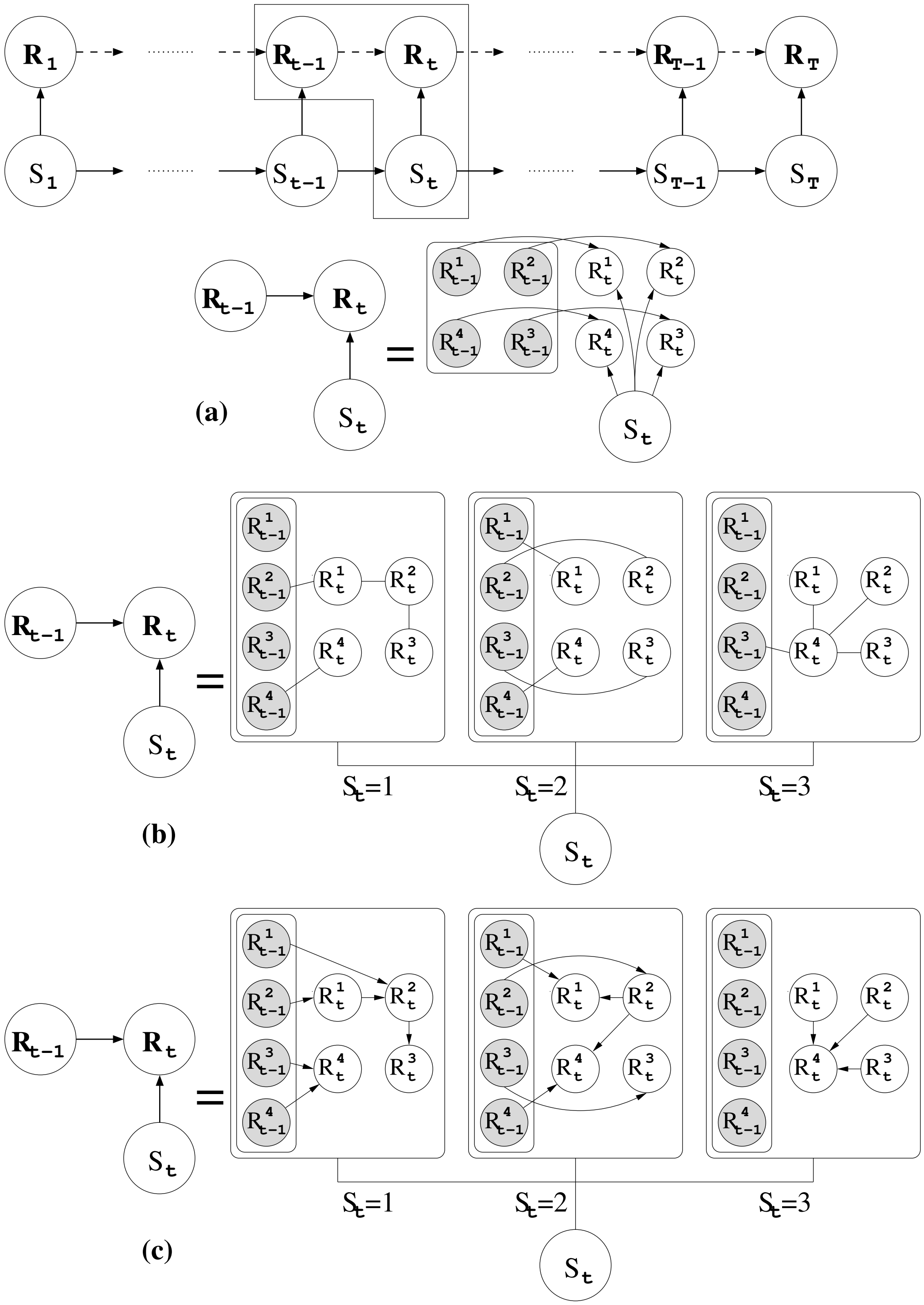

Figure 6.2: Bayesian network for a hypothetical (a) HMM-Chains; (b) HMM-CCL; (c) AR-HMM-PUC-MaxEnt

6.1.2 Application to Multi-Site Precipitation Modeling

As mentioned in Section 5.1.1, rainfall amounts for individual stations are sometimes modeled as a mixture of exponentials. A useful model is a mixture of a Dirac delta functions and C exponentials (conditionally independent given the state)

$$
\begin{aligned}
p\left(\boldsymbol{r}_{nt}|S_{nt}=i\right) &= \prod_{m=1}^{M} p\left(r_{ntm}|S_{nt}=i\right) \text{ where} \\
p\left(r_{ntm}|S_{nt}=i\right) &= \begin{cases} \pi_{im0} & r_{ntm}=0 \\ \sum_{c=1}^{C} \pi_{imc}\alpha_{imc}\exp\left(-\alpha_{imc}r_{ntm}\right) & r_{ntm}>0 \end{cases}
\end{aligned}
$$

with $\pi_{imc} \geq 0$ and $\sum_{c=0}^{C} \pi_{imc} = 1$. This model (from Section 5.1.1) can be used as the emission model with HMMs, and in this manner we create a model capable of producing realistic daily precipitation amount simulations. The Bayesian network for this model is the same as for HMM-CI (Figure 6.1 (a)). Bellone (2000) suggests using a two-component mixture with a gamma or a negative binomial distribution instead of the exponential distributions.

The E-step update of π's and α's can be performed using the methodology described in Section 4.4.2. To compute the updated probabilities in Expressions 4.24 and 4.25, we first need to compute the posterior probabilities of the mixture components $p_{ntm}^{ic} = P\left(C_{ntm} = c | S_{nt} = i, r_{ntm}\right)$:

$$\begin{aligned} p_{ntm}^{ic} &= \frac{\pi_{imc} p\left(r_{ntm} | S_{nt} = i\right)}{\sum_{c=0}^{C} \pi_{imc} p\left(r_{ntm} | S_{nt} = i\right)} \\ &= \begin{cases} 1 & c = 0,\ r_{ntm} = 0, \\ 0 & c \neq 0,\ r_{ntm} = 0 \text{ or } c = 0,\ r_{ntm} > 0, \\ \frac{\pi_{imc}\alpha_{imc}\exp(-\alpha_{imc} r_{ntm})}{\sum_{c=1}^{C} \pi_{imc}\alpha_{imc}\exp(-\alpha_{imc} r_{ntm})} & c \neq 0,\ r_{ntm} > 0, \end{cases} \end{aligned}$$

and then to update the probability distributions in Equation 4.26. Equivalently, we update the weights $W_{nt}(i)$ from Equation 3.18, setting the new weights to $W_{nt}(i) \times p_{ntm}^{ic}$. Then the $\boldsymbol{\pi}$ parameters can be updated as

$$\hat{\pi}_{imc} = \frac{\sum_{n=1}^{N} \sum_{t=1}^{T} W_{nt}(i)\, p_{ntm}^{ic}}{\sum_{n=1}^{N} \sum_{t=1}^{T} \sum_{c=0}^{C} W_{nt}(i)\, p_{ntm}^{ic}} = \frac{\sum_{n=1}^{N} \sum_{t=1}^{T} A_{nt}(i)\, p_{ntm}^{ic}(i)}{\sum_{n=1}^{N} \sum_{t=1}^{T} A_{nt}(i)}$$

with the mixing probability for zero rainfall identical to the update rule for no precipitation for the categorical case:

$$\hat{\pi}_{im0} = \frac{\sum_{\substack{n,t \\ r_{nt}=0}} A_{nt}(i)}{\sum_{n=1}^{N} \sum_{t=1}^{T} A_{nt}(i)}.$$

To update λ's, we first need to recall that to minimize $KL\left(p(x) \parallel \alpha \exp(-\alpha x)\right)$, $\alpha = \left[E_p(X)\right]^{-1}$. Then

$$\hat{\alpha}_{imc} = \left[\frac{\sum_{n=1}^{N} \sum_{t=1}^{T} W_{nt}(i)\, p_{ntm}^{ic} r_{ntm}}{\sum_{n=1}^{N} \sum_{t=1}^{T} W_{nt}(i)\, p_{ntm}^{ic}}\right]^{-1} = \frac{\sum_{n=1}^{N} \sum_{t=1}^{T} A_{nt}(i)\, p_{ntm}^{ic}}{\sum_{n=1}^{N} \sum_{t=1}^{T} A_{nt}(i)\, p_{ntm}^{ic} r_{ntm}}.$$

The computation required for the update is then $O(NTMC)$ to compute p_{ntm}^{ic}, $O(NTMC)$ to update π's and α's. If the data has relatively few precipitation occurrences, the computation can be sped up to $O\left(\#\left(r_{ntm} = 0\right) + C \times \#\left(r_{ntm} = 1\right)\right)$.

6.2 HMM with Chow-Liu Trees

The HMM-CI and HMM-Chain models can often be too simplistic in real applications since they do not capture direct interactions between the components of $\boldsymbol{R}_{nt}$. Chow-Liu trees are designed to capture some such multivariate dependencies. In this section, we described a model obtained by using Chow-Liu tree distributions as the emission probability distributions for HMMs. We can use HMMs with Chow-Liu trees or conditional Chow-Liu forests to model the output variable given the hidden state. HMMs can model the temporal structure of the data while the Chow-Liu models can capture "instantaneous" dependencies between multivariate outputs as well as additional dependence between vector components at consecutive observations over time that the state variable does not capture.

By combining HMMs with the Chow-Liu tree model and with the conditional Chow-Liu forest model we obtain HMM-CL and HMM-CCL models, respectively. The set of parameters $\boldsymbol{\Theta}$ for these models with K hidden states and B-valued M-variate vector sets consists of a $K \times K$ transition matrix $\boldsymbol{\Gamma}$, a $K \times 1$ vector $\boldsymbol{\Pi}$ of probabilities for the first hidden state in a sequence, and Chow-Liu trees or conditional forests $T_i\left(\boldsymbol{r}|\boldsymbol{r}_{previous}\right)$ with parameters $(\boldsymbol{\phi}_i, \boldsymbol{\upsilon}_i)$ for each hidden state i. Examples of graphical model structures for both the HMM-CL and HMM-CCL are shown in Figure 6.1 (b) and 6.2 (b), respectively. The likelihood of $\boldsymbol{\Theta}$ can then be computed as

$$
\begin{aligned}
L(\mathbf{\Theta}) &= P(\mathbf{R}|\mathbf{\Theta}) = \prod_{n=1}^{N} \sum_{s_{n,1:T}} P\left(S_{n,1:T}, \boldsymbol{r}_{n,1:T}|\mathbf{\Theta}\right) \\
&= \prod_{n=1}^{N} \sum_{S_{n,1:T}} P\left(s_{n1}|\mathbf{\Theta}\right) \prod_{t=2}^{T} P\left(s_{nt}|s_{n,t-1}, \mathbf{\Theta}\right) \prod_{t=1}^{T} P\left(\boldsymbol{r}_{nt}|S_{nt}, \boldsymbol{r}_{n,t-1}, \mathbf{\Theta}\right) \\
&= \sum_{n=1}^{N} \sum_{i_{n1}=1}^{K} \pi_{i_{n1}} T_{i_{n1}}\left(\boldsymbol{r}_{n1}\right) \sum_{t=2}^{T} \sum_{i_{nt}=1}^{K} \gamma_{i_{n,t-1}i_{nt}} T_{i_{nt}}\left(\boldsymbol{r}_{nt}|\boldsymbol{r}_{n,t-1}\right)
\end{aligned}
$$

with $P\left(\boldsymbol{r}_{nt}|s_{nt}, \boldsymbol{r}_{n,t-1}, \mathbf{\Theta}\right) = P\left(\boldsymbol{r}_{nt}|s_{nt}, \mathbf{\Theta}\right)$ and $T_i\left(\boldsymbol{r}_{nt}|\boldsymbol{r}_{n,t-1}\right) = T_i\left(\boldsymbol{r}_{nt}\right)$ for the HMM-CL. For hidden state S_{t-1} taking value i, the probability distribution $P\left(\mathbf{R}_t|\mathbf{\Theta}\right)$ is just a mixture of Chow-Liu trees (Meilă and Jordan, 2000) with mixture coefficients $(\gamma_{i1}, \ldots, \gamma_{iK})$ equal to the i-th row of the transition matrix $\mathbf{\Gamma}$.

Each $\boldsymbol{\upsilon}_i$ of HMM-CL has $M(B-1)$ free parameters to specify the marginals for M variables and $|\mathcal{E}_T|(B-1)^2$ free parameters to specify the pairwise probabilities; the structure $\boldsymbol{\phi}_i$ is specified by parameters specifying the edges of $\mathcal{E}_T$. For HMM-CCL, each $\boldsymbol{\upsilon}$ consists of $M(B-1) + |\mathcal{E}_x|(B-1)^2 + |\mathcal{E}_y| B(B-1)$ free parameters.

The M-step update of the emission parameters $\mathbf{\Upsilon}$ follows the Chow-Liu tree (or conditional Chow-Liu forest) algorithm as described in Figures 4.2 or 4.3. First, one needs to compute the univariate and bivariate marginals of P_i (Equation 3.19) for each $i = 1, \ldots, K$, a total of $O\left(NTKM^2B^2\right)$ computations. After that, the mutual informations for all variable pairs are computed, a $O\left(M^2B^2\right)$ computation complexity per hidden state. Finally, a maximum spanning tree is constructed with pairwise mutual informations as weights in $O\left(M^2\right)$ time per hidden state.

6.2.1 HMM with Tree-Structured Normals

Similar to distributions for categorical data, we can combine HMMs with tree-structured multivariate normal distributions, setting $p(\boldsymbol{r}_{nt}|S_{nt}=i,\boldsymbol{\Upsilon}) = p_{\mathcal{N}CL}(\boldsymbol{r}_{nt}|\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)$ as defined in Section 5.2.2. This model, HMM-CL-Normal, mirrors HMM-CL in structure (its Bayesian network structure class is the same as for HMM-CL, Figure 6.1 (b)) and in parameter estimation. $\boldsymbol{\upsilon}_i$ has $M+|E_T|$ free parameters. The M-step emission update for HMM-CL-Normal is performed following the steps of Figure 4.2. First, we estimate all univariate and bivariate marginal distributions by computing $\boldsymbol{\mu}_i$ and $\mathbf{R}_i$ for distributions P_i of Equation 3.19; this can be performed in $O(NTKM^2)$ time. Mutual information estimation does not require any computation since absolute values of correlations can be used instead. A maximum spanning tree is obtained by Kruskal's algorithm (e.g., Cormen et al., 1990) in $O(M^2)$ time per hidden state. Once the tree is obtained, the concentration matrix entries needed for computation of $p(\boldsymbol{r}_{nt}|s_{nt})$ can be obtained in time $O(M)$ per hidden state.

6.3 HMM with Emission Models of Higher Complexity

Both HMM-CL and HMM-CCL allow us to model some of the multivariate dependency directly. However, the tree structure of the dependency model for the emissions may be too restrictive. This is, perhaps, most pronounced in the HMM-CCL model where emission distributions are broken into multiple independent tree-structured components. We can use the maximum entropy models from Section 4.3 to relax the structure restrictions.

6.3.1 HMM with Full Bivariate MaxEnt

We consider modeling $P\left(\boldsymbol{r}_{nt}|S_{nt}=i,\boldsymbol{\Upsilon}\right)$ as a full bivariate maximum entropy model. For the binary case, the emission probability distributions can be expressed as

$$\begin{aligned}
P\left(\boldsymbol{r}_{nt}|S_{nt}=i,\boldsymbol{\Upsilon}\right) &= \frac{1}{C}\exp\left(\sum_{j=1}^{M}\alpha_{ij}r_{ntj}+\sum_{j=1}^{M}\sum_{k=j+1}^{M}\beta_{ijk}r_{ntj}r_{ntk}\right)\\
\text{where } C &= \sum_{\boldsymbol{R}}\exp\left(\sum_{j=1}^{M}\alpha_{ij}r_{j}+\sum_{j=1}^{M}\sum_{k=j+1}^{M}\beta_{ijk}r_{j}r_{k}\right),\\
\text{and } \boldsymbol{v}_i &= \left(\alpha_{i1},\ldots,\alpha_{iM},\beta_{i12},\ldots,\beta_{i,M-1,M}\right)\in\mathbb{R}^{M(M+1)/2}.
\end{aligned}$$

We will call this model HMM-Full-MaxEnt. An example of a resulting Bayesian network is shown in Figure 6.1 (c). This model has $M\left(M+1\right)/2$ free parameters per hidden state and captures all of the bivariate variable interactions, but at a significant computational cost. In Section 4.3, we saw that the parameters of exponential models are usually learned via iterative algorithms, and the full bivariate maximum entropy model is no exception. Each update of $\boldsymbol{v}_i$ in the M-step of each Baum-Welch iteration would, in turn, be iterative. For small values of M, one can use iterative scaling (IS) (e.g., Jelinek, 1998) to update $\boldsymbol{v}_i$. IS updates the parameters to match the bivariate distributions of $P_i\left(\boldsymbol{r}\right)$ and $P\left(\boldsymbol{r}|\boldsymbol{v}_i\right)$. Bivariate marginals of P_i can be precomputed in $O\left(NTKMB^2\right)$ time. However, the exact computation of each iteration of IS is $O\left(2^{M-1}\right)$,[4] exponential in the dimensionality of the data vectors. There are alternative techniques for parameter estimation of the full bivariate maximum entropy distributions (e.g., sampling, belief propagation), all of them computationally involved.

[4]Iterative scaling for full bivariate MaxEnt model requires computation of univariate and bivariate marginals at each iteration.

6.3.2 HMM with Saturated Normal Distributions

The real-valued Gaussian counterpart of the HMM-Full-MaxEnt is the HMM with emission distributions modeled as normals with a full covariance matrix (Section 5.2.1):

$$p\left(\boldsymbol{r}_{nt}|S_{nt}=i,\boldsymbol{\Upsilon}\right)=p_{\mathcal{N}}\left(\boldsymbol{r}_{nt}|\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i\right).$$

We denote this model HMM-Full-Normal. HMM-Full-Normal has the same graphical model interpretation as HMM-Full-MaxEnt (Figure 6.1 (c)), and the same number of free parameters, $M\left(M+1\right)/2$ per hidden state. In contrast, the update of emission parameters of HMM-Full-Normal can be done very efficiently, in only $O\left(NTKM^2\right)$:

$$\hat{\boldsymbol{\mu}}_i=E_{P_i}\left(\boldsymbol{R}\right)=\frac{\sum_{n=1}^{N}\sum_{t=1}^{T}A_{nt}\left(i\right)\boldsymbol{r}_{nt}}{\sum_{n=1}^{N}\sum_{t=1}^{T}A_{nt}\left(i\right)}$$

and

$$\hat{\boldsymbol{\Sigma}}_i=Var_{P_i}\left(\boldsymbol{R}\right)=\frac{\sum_{n=1}^{N}\sum_{t=1}^{T}\left(\boldsymbol{r}-\hat{\boldsymbol{\mu}}_i\right)\left(\boldsymbol{r}-\hat{\boldsymbol{\mu}}_i\right)'}{\sum_{n=1}^{N}\sum_{t=1}^{T}A_{nt}\left(i\right)}.$$

Once the $\hat{\boldsymbol{\Sigma}}_i$ are computed, the concentration matrices $\mathbf{K}_i$ needed for computation of $P\left(\boldsymbol{r}_{nt}|S_{nt}=i,\boldsymbol{\Upsilon}\right)$ can be obtained in $O\left(M^3\right)$ per hidden state.

6.3.3 HMM with Product of Univariate Conditional Max-Ent Distributions

As we have seen in Section 4.3.2, we can use PUC-MaxEnt models to learn multivariate dependencies from data. Let $\mathcal{V}$ be the set of vertices corresponding to vector variables at time t, and $\mathcal{V}_y$ be the set of vertices corresponding to vector variables at time $t-1$. Then

$$\begin{aligned} P\left(\boldsymbol{r}_{nt}|S_{nt}=i,\boldsymbol{\Upsilon}\right) &= P_{PUC-ME}\left(\boldsymbol{r}_{nt}|\boldsymbol{\delta}_i,\mathcal{F}_i\right) \\ \text{where } P_{PUC-ME}\left(\boldsymbol{r}\right) &= \prod_{v\in\mathcal{V}}\frac{1}{Z\left(\boldsymbol{r}_{pa(v)}\right)}\exp\left(\sum_{f\in\mathcal{F}_{iv}}\delta_{if}f\left(\boldsymbol{r}\right)\right). \end{aligned}$$

The structure parameters are the set of functions, $\boldsymbol{\phi}_i = \mathcal{F}_i$. Coefficients $\boldsymbol{\delta}_i$ are the numeric parameters defining the distribution given the structure, i.e., $\boldsymbol{\upsilon}_i = \boldsymbol{\delta}_i$. We will denote this HMM as HMM-PUC-MaxEnt. We can also similarly model conditional probability distribution as $P\left(\boldsymbol{r}_{nt}|\boldsymbol{r}_{n,t-1},S_{nt}=i,\boldsymbol{\Upsilon}_i\right)$ as a PUC-MaxEnt model, following Equations 4.15 and 4.16:

$$\begin{aligned} P\left(\boldsymbol{r}_{nt}|\boldsymbol{r}_{n,t-1},S_{nt}=i,\boldsymbol{\Upsilon}\right) &= P_{PUC-ME}\left(\boldsymbol{r}_{nt}|\boldsymbol{r}_{n,t-1},\boldsymbol{\delta}_i,\mathcal{F}_i\right) \\ \text{where } P_{PUC-ME}\left(\boldsymbol{r}|\boldsymbol{r}^\star\right) &= \prod_{v\in\mathcal{V}}\frac{1}{Z\left(\boldsymbol{r}_{pa_x(v)},\boldsymbol{r}^\star_{pa_y(v)}\right)}\exp\left(\sum_{f\in\mathcal{F}_{iv}}\delta_{if}f\left(\boldsymbol{r},\boldsymbol{r}^\star\right)\right). \end{aligned}$$

We will denote HMM with PUC-MaxEnt emissions by HMM-PUC-MaxEnt (Figure 6.1 (d)), and AR-HMM with PUC-MaxEnt emissions as AR-HMM-PUC-MaxEnt (Figure 6.2 (c)).

We will concentrate on univariate and bivariate functions, as described in Section 4.3.5. Algorithm StructureLearningPUC-MaxEnt(P_i) (Figure 4.10) can be used to update the parameters of the emission parameters at each M-step. To run the algorithm, one does not require distributions $P_i\left(\boldsymbol{r}\right)$, just its bivariate marginals obtainable in $O\left(NTKM^2\right)$ time. Gains for candidate features can be computed in $O\left(NTM\right)$ time per iteration of Newton-Raphson, no more than $O\left(M^2\right)$ computed per iteration of the algorithm. Once a feature is added, each iteration of conjugate gradient algorithm is performed in $O\left(NTM\right)$ time as well. Since there are $O\left(M^2\right)$

possible pairwise interactions, no more than $O(M^2)$ features will be added per hidden state.

6.4 Summary

In this chapter, we introduced hidden Markov models for multivariate temporal data. The models vary (via the emission probabilities) in terms of the level of multivariate interactions they encode, from conditional independence (no direct multivariate interactions) to maximum entropy models (possibly, all direct multivariate interactions). However, the more complex the model gets, the higher (typically) the computational complexity of the parameter estimation. Table 6.1 summarizes the complexities of the models. The trade-off is especially evident for models for categorical data with structures more complex than trees. For these models, the structure and the numeric parameter learning cannot be performed simultaneously effectively requiring inner loop iterations to search for the improved dependency structure.

There are a number of novel contributions in this chapter. HMM-CL and HMM-CCL (Kirshner et al., 2004) offer efficient compromises between the quick but often simplistic HMM-CI and the computationally intensive HMM-Full-MaxEnt. HMM-PUC-MaxEnt and AR-HMM-PUC-MaxEnt allow to learn complex interactions between variables of $\boldsymbol{R}_{nt}$ given the hidden state. While both of the models are more computationally demanding than simpler (HMM-CI, HMM-CL and HMM-CCL) models, HMM-PUC-MaxEnt and AR-HMM-PUC-MaxEnt allow for richer dependency structures, and their computational demands can be reduced by restricting the complexity of the dependence structure.

In terms of related work, Zucchini and Guttorp (1991) explored the HMM-CI model for precipitation occurrence modeling. As a follow-up, Hughes and Guttorp (1994) introduced a non-homogeneous version of HMM-CI, and later described a restricted non-homogeneous HMM-Full-MaxEnt model (Hughes et al., 1999). HMM-CL and HMM-CCL models (Kirshner et al., 2004) introduce an aspect of learning the dependence structure between data variables, not commonly seen in the HMM framework. An exception is the work of Bilmes (1999, 2000) where HMMs are allowed to learn temporal dependencies between individual output variables.

Table 6.1: Comparison of complexities of HMM models for multivariate time series

Model	Complexity of $P(\boldsymbol{r}\vert\boldsymbol{r}_{prev}, s)$	Complexity of E-step	Complexity of M-step
HMM-CI	$O(M)$	$O(NTK(K+M))$	$O(NTKMB)$
HMM-Chains	$O(M)$	$O(NTK(K+M))$	$O(NTKMB^2)$
HMM-CL or HMM-CCL	$O(M)$	$O(NTK(K+M))$	$O(NTKM^2B^2)$
HMM-CL-Normal	$O(M)$	$O(NTK(K+M))$	$O(NTKM^2)$
HMM-Full-MaxEnt	$O(M^2)$	$O(NTK(K+M^2))$	$O\left(KI_1 2^{M-1} + NTKM^2\right)$
HMM-Full-Normal	$O(M^2)$	$O(NTK(K+M^2))$	$O(NTKM^2)$
HMM-PUC-MaxEnt or AR-HMM-PUC-MaxEnt	$O(M^2)$	$O(NTK(K+M^2))$	$O(NTKM^2(I_2M^3 + I_3M))$

Comment: MaxEnt models are assumed binary with univariate or bivariate features. Legend: N – the number of sequences; T – the length of each sequence; K – the number of hidden states; M – the number of variables in the data vectors; B – the number of values each of the variables take; I_1 – the number of iterations needed for IS to converge; I_2 – the number of iterations needed to compute Gain; I_3 – the number of iterations needed for conjugate gradient algorithm to converge.

Chapter 7

Experimental Results: Sensitivity of Learning HMMs for Multivariate Binary Data

In this chapter we are addressing the question of the sensitivity of learning the parameters of models from Chapter 6 for binary data with respect to the amount of available data. We will demonstrate empirically the ability to learn both the numeric and structural parameters given sufficient data, and we will also demonstrate that these models may not recover the original parameters and structure if the amount of training data is too small. We will also investigate whether a model can sufficiently approximate another model with different structural properties.

The motivation for this kind of analysis comes from limited historical data available for many applications. Assuming that we know the correct model type used to generate the data, can we accurately learn the model parameters given the amount of data that we have? For example, in the particular application of interest, are

24 sequences of 90 daily precipitation occurrence observations at 10 stations enough data to learn the parameters of a 6-state HMM-CL accurately? We will perform an empirical study in this chapter to answer this question.

7.1 Experimental Setup

The purpose of the experiment is to find out what amount of data is sufficient to learn accurately the parameters of a model. Assume for the moment that the type of the model is fixed and known, and that the set of true parameters of the model, the target distribution, is known. We will simulate data sets of different sizes from the target distribution and learn the parameters of the model from these sets. We can then measure the differences between the learned distributions and the target distribution.

We still need to define how to measure whether the learned distribution is similar enough to the target distribution. Since models for both the target and learned distributions are the same, one possibility is a direct comparison of parameters. The problem is the order in which emission distributions appear is not known, so a consideration of all $K!$ permutations of states may be necessary. Alternatively, we could use one of the information theory measures of difference between distributions, e.g., KL-divergence. The advantage is that such difference can be computed even for distributions of different types, as long as they are defined on the same domain. However, the KL-divergence between distributions with hidden variables usually cannot be computed in closed form, and HMMs are no exception,[1] although there are

[1] In order to define KL on HMMs, we need to specify the domain. We will assume that the domain of distributions in question consists of all multivariate sequences of the same length T and the same vector dimensionality M.

possible closed form approximations to KL for HMMs (e.g., Do, 2003). Instead of using closed form approximations, we will approximate the exact values of the KL by large sample approximation; we will generate a data set $\mathcal{D} = \{\boldsymbol{r}_{1,1:T}, \ldots, \boldsymbol{r}_{N_{KL},1:T}\}$ consisting of N_{KL} sequences (where N_{KL} is large) from target distribution $P(\boldsymbol{r})$, and then compute the difference in log-probability of the data under the target model and the learned model $P_{learned}$:

$$
\begin{aligned}
KL\left(P \parallel P_{learned}\right) &\approx \frac{1}{N_{KL}} \sum_{n=1}^{N_{KL}} \left(\ln P\left(\boldsymbol{r}_{n,1:T}\right) - \ln P_{learned}\left(\boldsymbol{r}_{n,1:T}\right)\right) \\
&= \frac{1}{N_{KL}} \ln P\left(\mathcal{D}\right) - \frac{1}{N_{KL}} \ln P_{learned}\left(\mathcal{D}\right). \qquad (7.1)
\end{aligned}
$$

The first term on line 7.1 is an approximation of $-H_P\left[\boldsymbol{R}\right]$, the negated entropy of the target distribution. We can divide both terms by the number of observation points in each sequence $(T \times M)$ making the terms independent of the number of observations (normalizing):

$$
KL_{norm}\left(P \parallel P_{learned}\right) \approx \frac{1}{N_{KL}TM} \ln P\left(\mathcal{D}\right) - \frac{1}{N_{KL}TM} \ln P_{learned}\left(\mathcal{D}\right). \qquad (7.2)
$$

For binary data, this normalized KL can be thought of as a relative entropy per bit of data. We will call the log-likelihood terms on line 7.2 *normalized or scaled* log-likelihoods. A model which assigns each binary observation a probability 0.5 (at random) would have the scaled log-likelihood of $-\ln 2 \approx -0.69315$; we expect all of the models trained on a data set to have scaled log-likelihoods higher than $-\ln 2$.[2] Then

$$
0 \leq KL_{norm}\left(P \parallel P_{learned}\right) \leq -H_P\left[\boldsymbol{R}\right] + \ln 2. \qquad (7.3)
$$

[2] $-\ln 2$ is a lower bound of the negative entropy for binary variables.

We perform the experiments under the following scheme. First, we choose types of models to examine (from the models in Chapter 6). For each type, we consider a range for K, the number of hidden states. For each combination of the model type and the number of hidden states, we generate multiple sets of parameters (target distributions) $\mathbf{\Theta}_1, \ldots, \mathbf{\Theta}_{N_m}$ for the model. For each target distribution, we generate 10 sets of N sequences each of length T of M-dimensional binary ($B = 2$) vectors. For each of such sets (with index $i_s = 1, \ldots, 10$), we learn back the parameters $\mathbf{\Theta}_{ni_s}$. To evaluate the models, we generate a large set from each original model with $N_{KL} = 10000$ and compute scaled log-likelihoods (and normalized KL-divergences) of that set under the corresponding target model and all of the models approximating that target model.

We perform two sets of experiments. With the first set, we investigate what factors influence how well one can learn the parameters of an HMM given the correct model type. We consider HMM-CI and HMM-CL with $K = 2, \ldots, 8$ hidden states. First, we generate $N_m = 20$ multiple sets of parameters per model type. We then generate $N = 10, 15, 20, 35, 50, 100, 200$ sequences of binary vectors of dimensionality $M = 15$ for each parameter set, each sequence of length $T = 100$ ($7 \times 7 \times 20 = 980$ models and $7 \times 7 \times 20 \times 10 = 9800$ sets of data for each type of the model). For each simulated set, we learn back the parameters of the model using the same model type used to generate that data set. We then evaluate the KL-divergence for each learned set of parameters[3]. Since normalized KL-divergence depends on the entropy of the target distribution (as the results in the next section suggest), we repeat these experiments for three sets of parameters of target distributions with different entropies. The average values of the entropies for the models used in this set of

[3]The settings of the EM algorithm used to estimate the parameters are discussed in Appendix C.

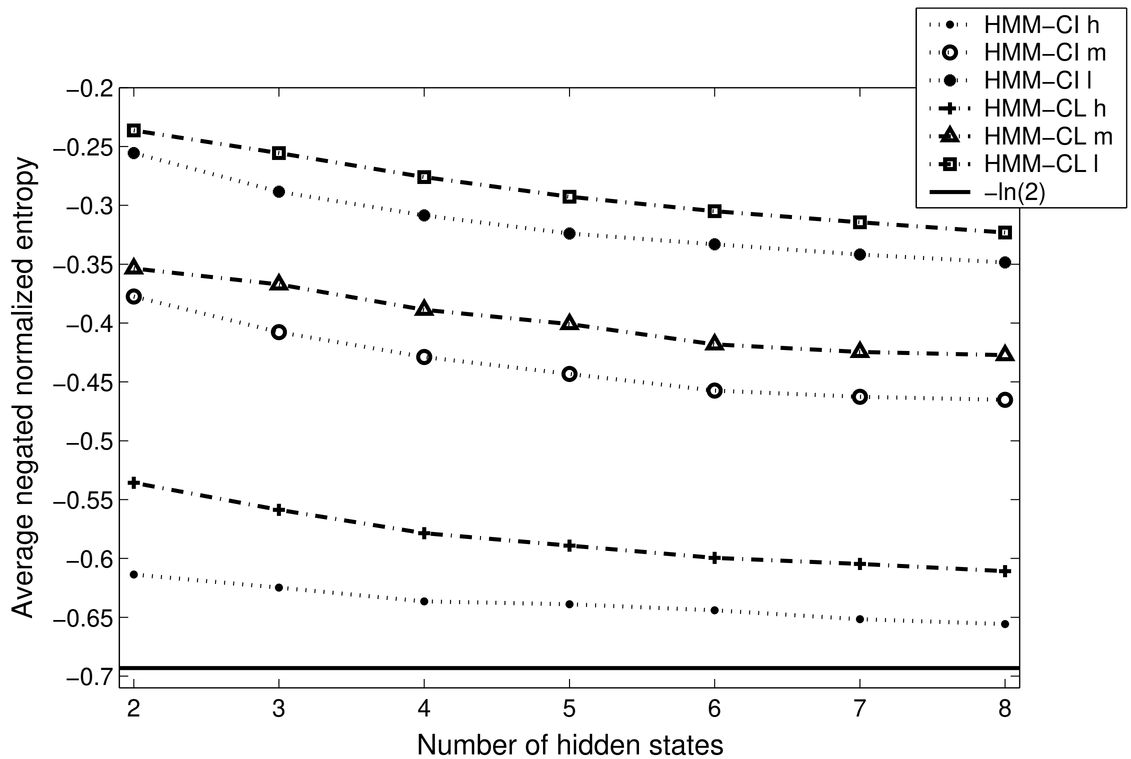

Figure 7.1: Estimated average negated normalized entropy $\frac{\ln P(\mathcal{D})}{N_{KL}TM}$ of the test models as a function of the number of hidden states K for the first experimental setup. The lines correspond to models HMM-CI and HMM-CL with high, moderate, and low average estimated entropy (denoted by h,m, and l, respectively). Also plotted the lower bound on the negated normalized entropy, $-\ln 2$.

experiments (crudely classified as low, medium, and high) are shown in Figure 7.1.

For the second set of experiments, we employ smaller test sets to study how effectively one can use models of type other than the true model type for learning target distributions. We consider six types of models, HMM-CI, HMM-Chains, HMM-CL, HMM-CCL, HMM-PUC-MaxEnt, and AR-HMM-PUC-MaxEnt. We use $K = 2, \ldots, 6$, $N_m = 20$, $N = 12, 16, 24, 36, 60, 100$, $T = 90$, and $M = 10$. Thus for each type of the model, we simulate $5 \times 20 \times 6 \times 10 = 6000$ data sets. For each of 1200 data sets corresponding to one six model types, we learn the parameters[3] for all six model types with the same K as the true model.

7.2 Results

7.2.1 First Set of Experiments

First, we look at how close the scaled log-likelihoods of learned models approach the scaled log-likelihoods of target distributions. Figure 7.2 summarizes the KL-divergence for HMM-CI and HMM-CL models with target distributions having high, moderate, and low entropies. Each of these plots show averaged scaled log-likelihood for each number of hidden states $K = 2, \ldots, 8$. From the figures it is apparent that the learning algorithm can recover the target distribution given a sufficient amount of data. Also, as expected, the accuracy of the learned model improves with larger data sets. For the same amount of data, HMM-CI can recover the target distribution with higher precision than HMM-CL; this is due to HMM-CL having a higher number of free parameters. It appears that models with higher entropy of the target distribution are estimated more accurately than same model types with parameters yielding models with lower entropies. One possible naive explanation is that if the original model has higher entropy, the learned models have a smaller range of values for scaled log-likelihood and thus less room to be worse by a fixed margin. For example, if the target distribution has scaled entropy of 0.65, the range of the scaled KL-divergence is only $[0, 0.0431]$ while a distribution with scaled entropy of 0.3 has its KL-divergence in the range of $[0, 0.3931]$.

To quantify whether a data set of a certain size is sufficient for learning the parameters of the underlying (known) distribution for HMMs, we compute the empirical probability that $KL_{norm}(P \parallel P_{learned}) \leq \tau$. We chose the value for the threshold

$\tau = 0.005$.[4] The probability can be computed based on 10×20 values of KL_{norm} for each value of K and N. If the probability is sufficiently high, we would consider the amount of data sufficient. Figure 7.3 contains contour plots of the probabilities that $KL_{norm} \leq \tau$ for HMM-CI (left) and HMM-CCL (right) with high, moderate, and low entropies. The x-axes are the number of free numeric parameters for the model rather than the number K of hidden states. The y-axes are the total number of binary points (in this case, $15 \times 100 = 1500$ times the number of sequences). Most of the level curves are roughly linear on the log-scale. The implication is that we need the log of the amount of data to be proportional to the number of free parameters to be able to learn the parameters of the model with accuracy. Also, as the entropy of the target distribution decreases, more data is needed to learn the model accurately according to the same threshold τ.

[4] The value is decided on from looking at the parameter values for the models, but is subjective and somewhat arbitrary.

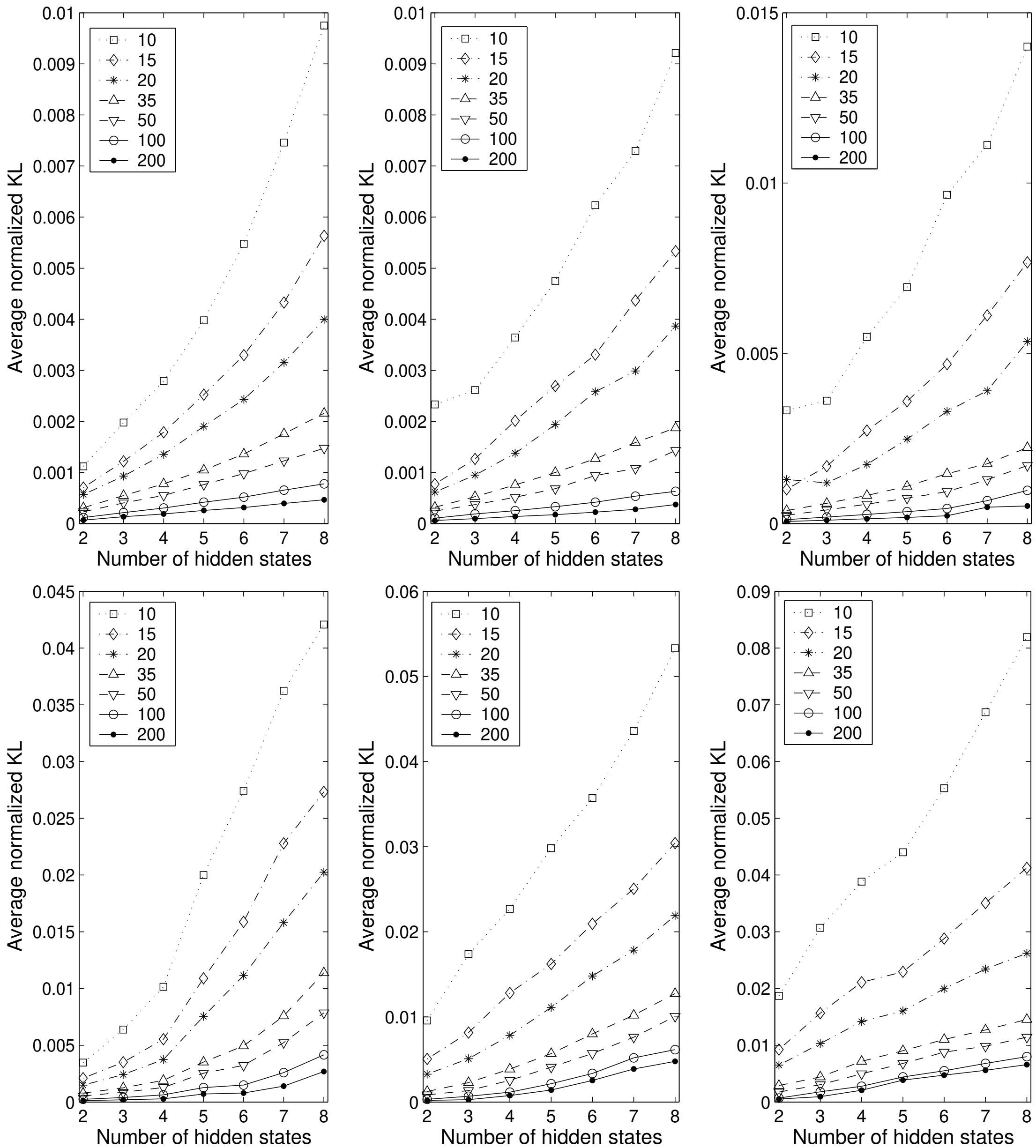

Figure 7.2: Estimated normalized KL-divergences for HMM-CI (top) and HMM-CL (bottom) with high (left), moderate (center), and low (right) average estimated entropy as a function of the number of hidden states. Curves on each plot correspond to different values of the number of sequences N.

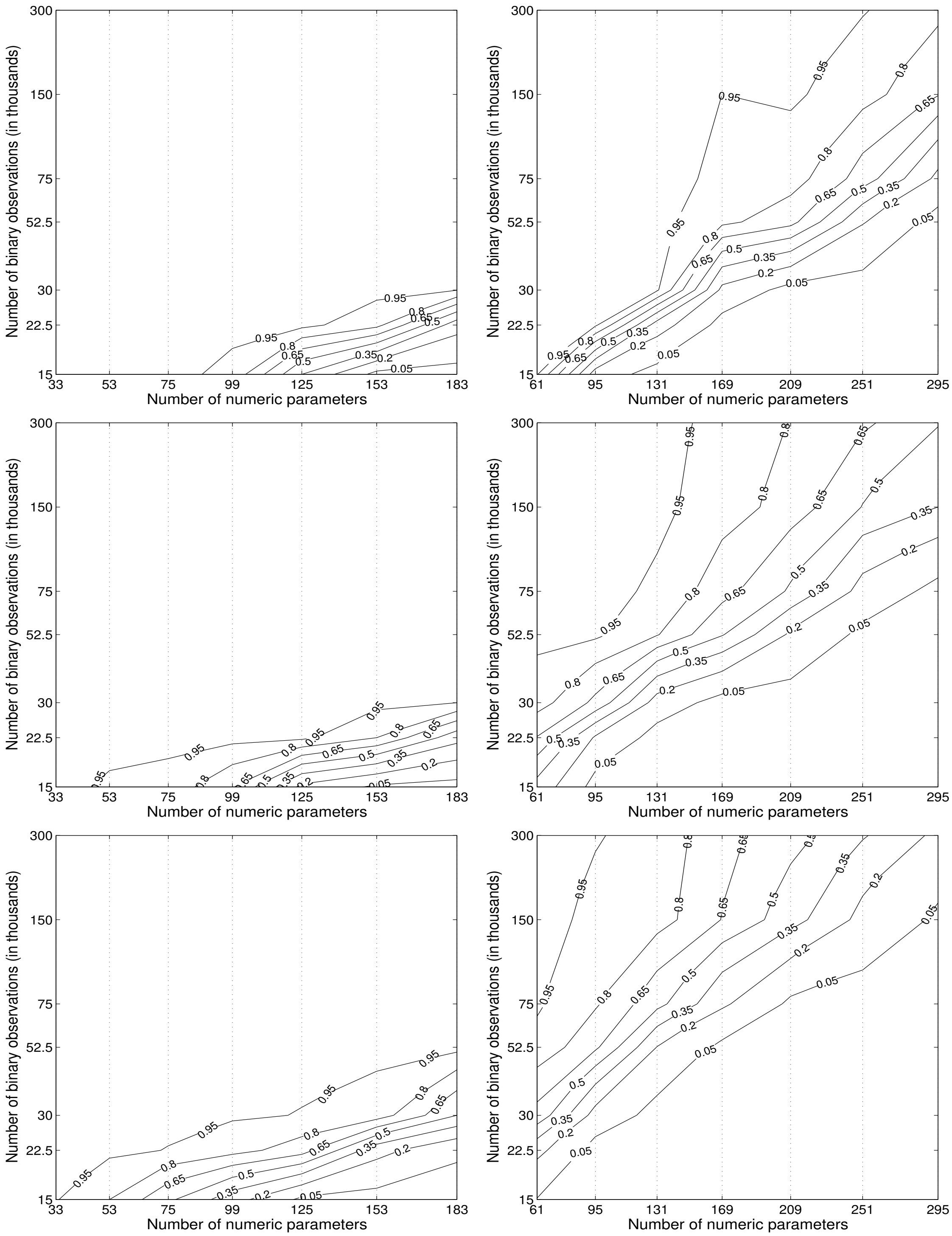

Figure 7.3: Contour plots of the probability that $KL_{norm} \leq \tau$ ($\tau = 0.005$) for HMM-CI (left) and HMM-CL (right) with high entropy (top), moderate entropy (center), and low entropy (bottom).

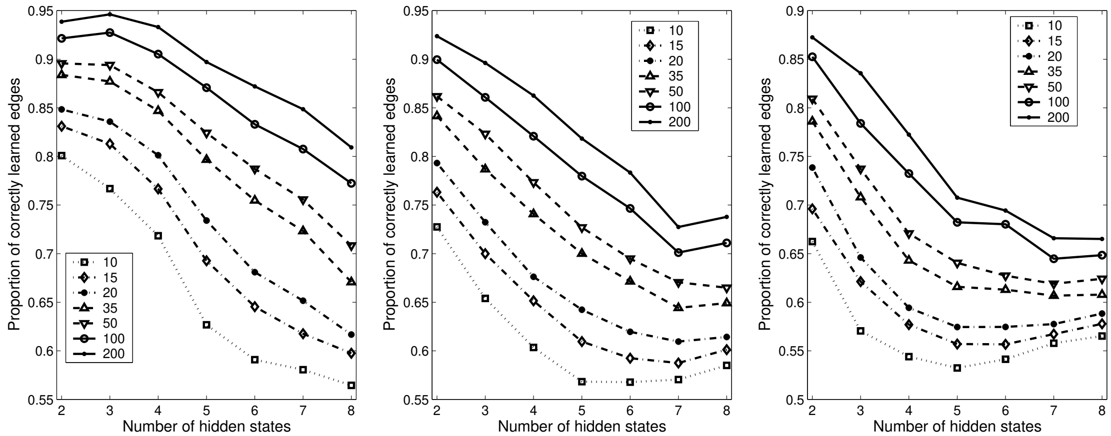

Figure 7.4: Upper bounds on the proportion of correctly learned edges for HMM-CL with high (left), moderate (center), and low (right) entropies as a function of the number of hidden states. Curves on each plot correspond to different values of the number of sequences N.

Finally, we examine whether HMM-CL can learn back the correct tree structure. We wish to estimate what proportion of the tree edges in the original distribution is recovered. To compute this number exactly, we need to match up the states of the original and the learned models, an unrealistic task. Instead, we can compute an upper bound to the proportion of recovered edges by lumping all of the edges in each model together, and then computing the proportion of edges appearing in both the original and the learned model. This upper bound on the proportion of correctly learned edges is shown in Figure 7.4. As expected, the structure is learned better when more data is available. Also, similar to observations from the estimated KL-divergence, given the same amount of data, the structure of the distribution with higher entropy can be learned more accurately than the one with the lower entropy.

7.2.2 Second Set of Experiments

First, we use the second experimental setup to extend the results in Figure 7.2 to other model types. The estimated values of KL-divergence for learned models with the same type as the true model are shown Figure 7.5. All plots are consistent with the intuition that more data will result in more accurate models. The plots corresponding to HMM-CI (top left) and HMM-CL[5] (top center) are consistent with plots from the first experiment (Figure 7.2). Somewhat surprisingly, HMM-CCL (bottom center) requires a little fewer data sequences than HMM-CL to learn back the model parameters. Figure 7.6 further suggests this may be due in part to HMM-CCL recovering the structure better than HMM-CL. Both HMM-PUC-MaxEnt (top right) and AR-HMM-PUC-MaxEnt appear to hit a plateau in KL-divergence. This may be due to the stopping criterion in the algorithm used to learn the feature sets (Figure 4.10) as no feature adding less than certain improvement in log-likelihood is added.

Finally, we consider how various models can approximate models of different types of structure (no structure, multivariate, and temporal). We consider three types for the true model: HMM-CI (no temporal or multivariate structure for hidden states), HMM-CL (multivariate structure only), HMM-CCL (both multivariate and temporal structure). For HMM-CI (Figure 7.7), any of the six HMM types can do a reasonable job approximating it. However, models other than the true type (HMM-CI) tend to overfit, especially the models that learn features. For HMM-CL (Figure 7.8), models learning only the spatial structure, HMM-CL and HMM-PUC-MaxEnt, perform the

[5]This plot can provide an answer to the question in the beginning of this chapter. The plot suggests that average KL-divergence for learning back the parameters of a 4-state HMM-CL is close to 0.007, an acceptable number. However, as we have seen in the previous subsection, the entropy of the true model plays a significant part in a possible answer.

best. Models also learning the temporal structure perform reasonably but not as well as HMM-CL or HMM-PUC-MaxEnt. HMM-CI and HMM-Chains are at a clear disadvantage since they have no explicit mechanism to learn spatial dependencies. For HMM-CCL (Figure 7.9), predictably, AR-HMM-PUC-MaxEnt does the best job (after HMM-CCL) as it can learn both spatial and temporal features. The rest of the models perform poorly since they cannot model additional temporal or spatial information.

7.3 Summary

In this chapter, we have shown empirically that HMM-CI, HMM-CL, and HMM-CCL (both parameters and structures) can be learned accurately given sufficient data. We defined a measure of accuracy in learning the parameters, and studied how accurately can the parameters of a multivariate HMM be learned as a function of the data set size. The empirical plots suggest that the size of the data set needed for accurate parameter estimation depends not only on the number of free parameters, but also on the entropy of the target distribution. The experiments also suggest that models capable of learning the same types of features, spatial and/or temporal, as the true model can be used as a good approximation although with a risk of overfitting.

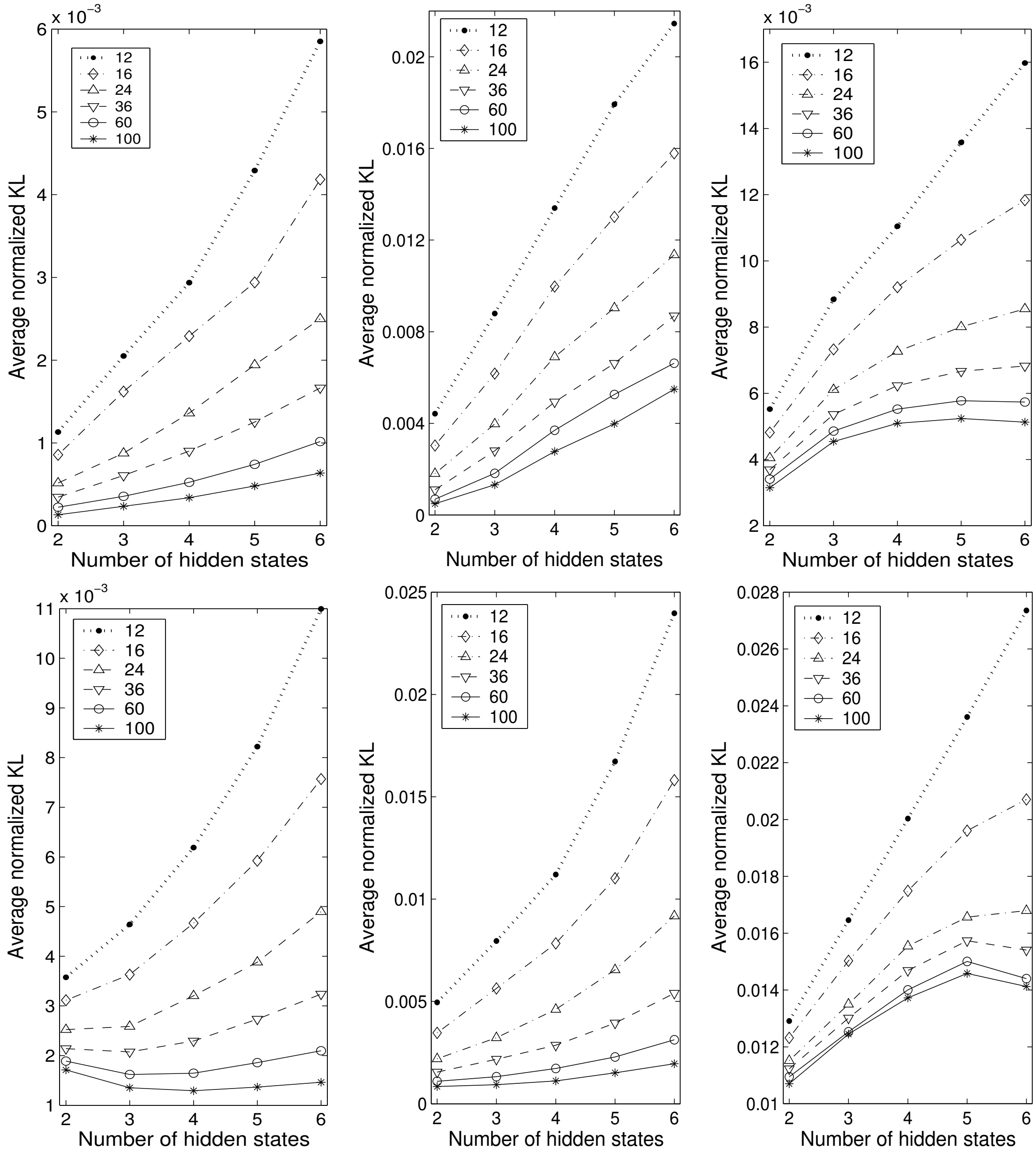

Figure 7.5: Estimated normalized KL-divergences for HMM-CI (top left), HMM-CL(top center), HMM-PUC-MaxEnt (top right), HMM-Chains (bottom left), HMM-CCL (bottom center), and AR-HMM-PUC-MaxEnt (bottom right) as a function of the number of hidden states. Curves on each plot correspond to different values of the number of sequences N.

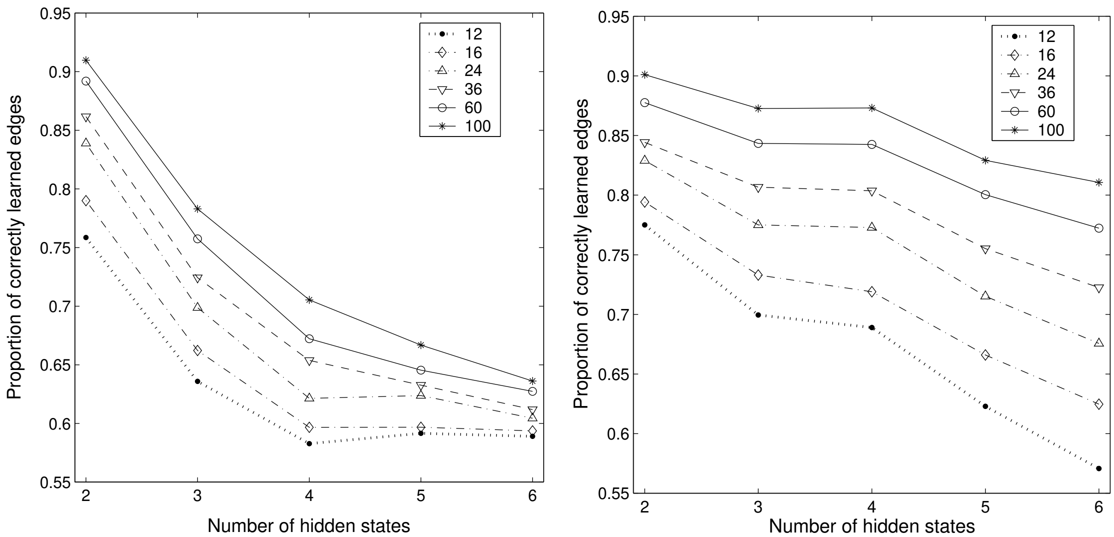

Figure 7.6: Upper bounds on the proportion of correctly learned edges for HMM-CL (left) and HMM-CCL (right) as a function of the number of hidden states. Curves on each plot correspond to different values of the number of sequences N.

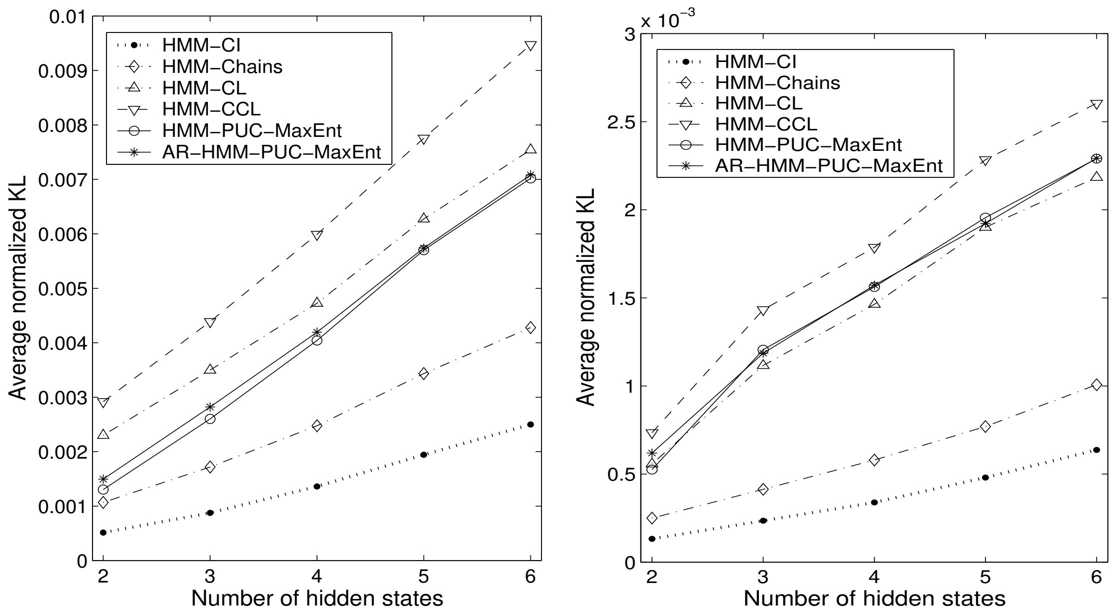

Figure 7.7: Estimated normalized KL-divergences for different models for estimation the parameters of a HMM-CI based on 24 sequences (left) and 100 sequences (right). Curves on each plot correspond to different models.

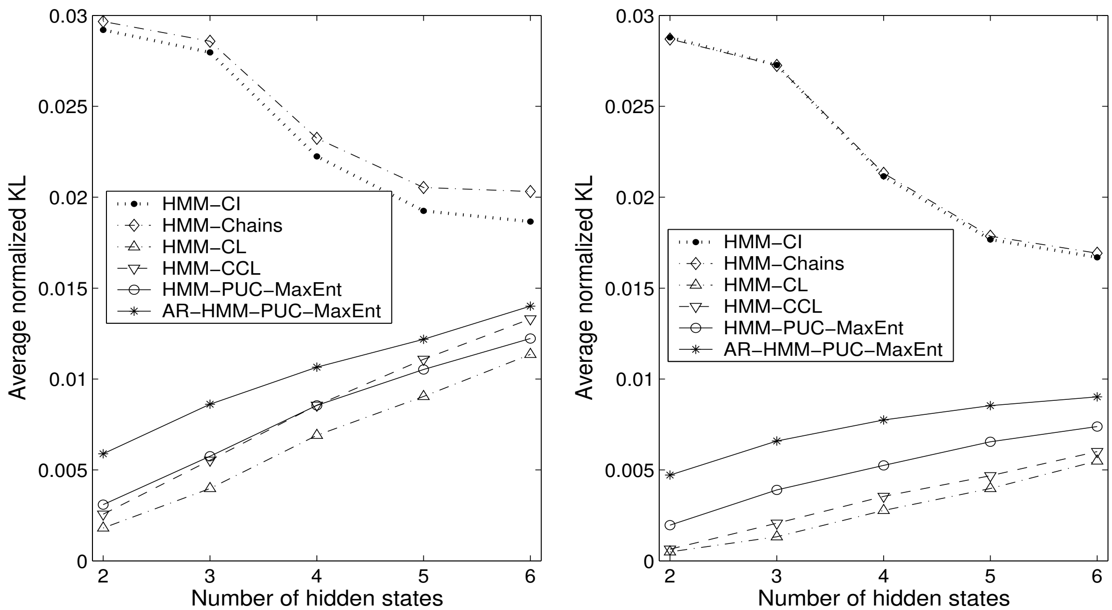

Figure 7.8: Estimated normalized KL-divergences for different models for estimation the parameters of a HMM-CL based on 24 sequences (left) and 100 sequences (right). Curves on each plot correspond to different models.

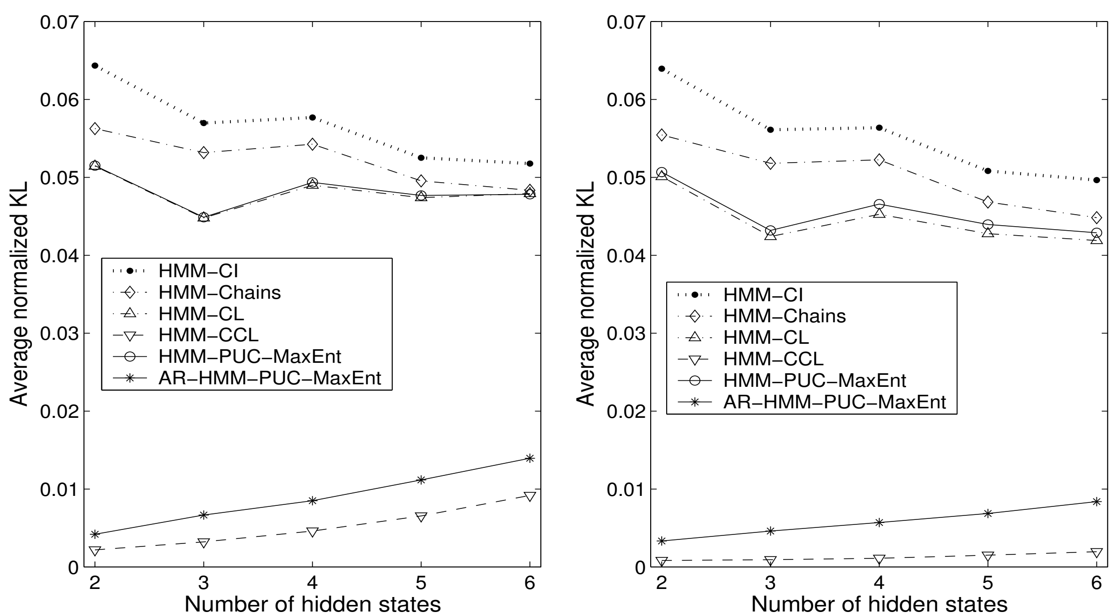

Figure 7.9: Estimated normalized KL-divergences for different models for estimation the parameters of a HMM-CCL based on 24 sequences (left) and 100 sequences (right). Curves on each plot correspond to different models.

Chapter 8

Experimental Results: Large Scale Study of Precipitation Modeling for Different Geographic Regions

In this chapter, we apply the methodology developed in Chapters 3-6 to the problem of modeling daily precipitation occurrence simultaneously at multiple rain stations for different geographic regions. First, we briefly summarize prior statistical work for this application (Section 8.1). Then, we describe the analysis methodology (Section 8.2). After that, we consider four different geographic regions with different amounts of data (Section 8.3). For each region, we summarize the data, compare the performance of the models on the data, and consider possible interpretation of the weather states.

8.1 Related Work on Multi-site Precipitation Occurrence Modeling

Statistical approaches to precipitation occurrence modeling date back to the 1850's (Katz, 1985). However, until recently, most of the work concentrated on modeling data collected from a single weather station. Wilks and Wilby (1999) provide an overview of the literature related to early stochastic precipitation modeling. Most of the described methods model rainfall occurrence as a Markov chain with two states, wet and dry, sometimes using higher order Markov chains. The run-lengths of wet and dry spells commonly exhibit near-geometric distributions. Thus, 2-state Markov chains can often be used to simulate data with run-lengths matching that of the historical data. However, these single-station models cannot be easily extended to multiple sites as that requires modeling multivariate (or spatial) dependencies between the stations. Using an unconstrained Markov chain on the vector of occurrences is impractical since the number of parameters required is exponential in the number of stations (Guttorp, 1995). Wilks (1998) used multivariate probit models to match the pairwise correlations of daily observations, and later extended the method to capture some of the climate variability by allowing seasonal atmospheric variables to influence the parameters of the model (Wilks, 2002). Alternatively, a mathematical model of the spatial physical process can be used to simulate precipitation at multiple locations (e.g., Waymire et al., 1984; Goodall and Phelan, 1991; Cox and Isham, 1988). Yet another alternative is to model multi-site precipitation dependent only on the finite number of evolving weather states with states either being explicitly classified based on the atmospheric conditions (e.g., Hay et al., 1991; Wilson et al., 1992; Bardossy and Plate, 1992) or considered unobserved, the latter resulting in an HMM. Other alternatives include non-parametric methods, for example, k-nearest neighbor algorithm (Rajagopalan and Lall, 1999), or modeling station dependence via a Bayesian network (Cofiño et al., 2002).[1] In some cases, the weather states are not assumed to be discrete (e.g., Sansó and Guenni, 1999), and can even be viewed as an evolving random field (Allcroft and Glasbey, 2003).

Hidden Markov models have been successfully applied to multi-site precipitation modeling since they were introduced by Zucchini and Guttorp (1991). Hughes and Guttorp (1994) extended the HMM framework to allow the weather states to evolve dependent on the atmospheric variables. Initially, the occurrences were modeled conditionally independent given the states, but later models remove this restriction (Hughes et al., 1999). It is worth noting that the HMM framework can be easily extended to model multi-site rainfall *amounts* as well (e.g., Bellone, 2000; Bellone et al., 2000).

Precipitation modeling has a number of potential uses in water management and forecasting. (See, for example, Wilks and Wilby, 1999, for more information.) Simulated precipitation data sequences can be used to study interannual climate variability and possibly be used in forecasting. These simulated sequences can also be used as inputs in crop models. Historical records for a number of geographic regions (e.g., Kenya, India) contain a large quantity of missing data presenting a serious problem for analysis and forecasting. Stochastic precipitation models could alleviate some of the problems by estimating or simulating these missing values.

[1]Bellone (2000) contains an excellent review of the development of statistical methods for precipitation modeling.

8.2 Model Selection and Evaluation of the Results

As was mentioned in Chapter 1, we want to develop generative models (based on the historical data) that can produce realistic simulated daily rainfall occurrence sequences for networks of stations. Before analyzing possible models, we need to decide how to evaluate the models and what criteria to use to choose between models. We employ leave-k-out cross-validation for model selection. Assume we have a set of N ordered data sequences (typically these correspond to daily sequences for N different years or seasons). Under leave-k-out cross-validation, we will train N/k models, one for each set obtained by leaving-out k non-overlapping consecutive sequences. Each model is then evaluated on the corresponding left-out k sequences. For evaluation criteria, we use a mix of statistics capturing temporal, spatial, and interannual variability properties of the data, as well as the fit of the model to the data, and the ability of the model to fill-in missing observations. Possible criteria include the difference in linear correlation for observed and simulated data, the difference in probability of precipitation at a given station for observed and simulated data, the difference in precipitation persistence (as explained later) for observed and simulated data, the scaled log-likelihood of the left-out sequences (Chapter 7), and the average classification error in predicting randomly-selected observations that are deliberately removed from the left-out data and then predicted by the model. Persistence is defined as the probability of a precipitation occurrence for a particular station given precipitation event at that station at previous observation. Persistence can be thought of as a proxy for wet spell run-length distributions (Chapter 1) as these spells can often be modeled by a geometric distribution with one free parameter. Average classification error helps in assessing whether the model can remedy the common situation of missing station readings in real precipitation records.

The parameters for individual hidden states of fitted HMMs can often be visualized (as will be shown in the next section), and these sometimes called "weather states" are often of direct scientific interest from a meteorological viewpoint. Thus, in evaluating these models, models that can explain the data with fewer states are generally preferable.

8.3 Performance Analysis on Geographic Regions

We use the models described earlier in the thesis to analyze four data sets, each for a distinct geographic region: Ceará region of Northeastern Brazil, Southwestern Australia, Western U.S., and Queensland region of Northeastern Australia. The exact setup for the experiments can be found in Appendix C. Some of the analysis from Subsections 8.3.2 and 8.3.3 appeared in Kirshner et al. (2004).

8.3.1 Ceará Region of Northeastern Brazil

First, we model the data for the Ceará region. (The set was described in detail in Section 1.1.) We will use leave-one-out and leave-six-out cross-validations for the Ceará set. By leaving out only one data sequence at a time, we evaluate models trained on sets almost identical to the whole set, and get an accurate estimate for the average out-of-sample log-likelihood, an indicator of the predictive power of the model. However, this method is computationally intensive (as it requires almost N times the computation of an algorithm run on the whole data). Also, one-sequence validation sets are too small to evaluate the correlation and the persistence as these statistics have high variance for individual years (as illustrated for persistence in Figure 8.1), so larger validation sets are desirable. However, since the data set is already small, leaving out large number of seasons will leave too little data to train

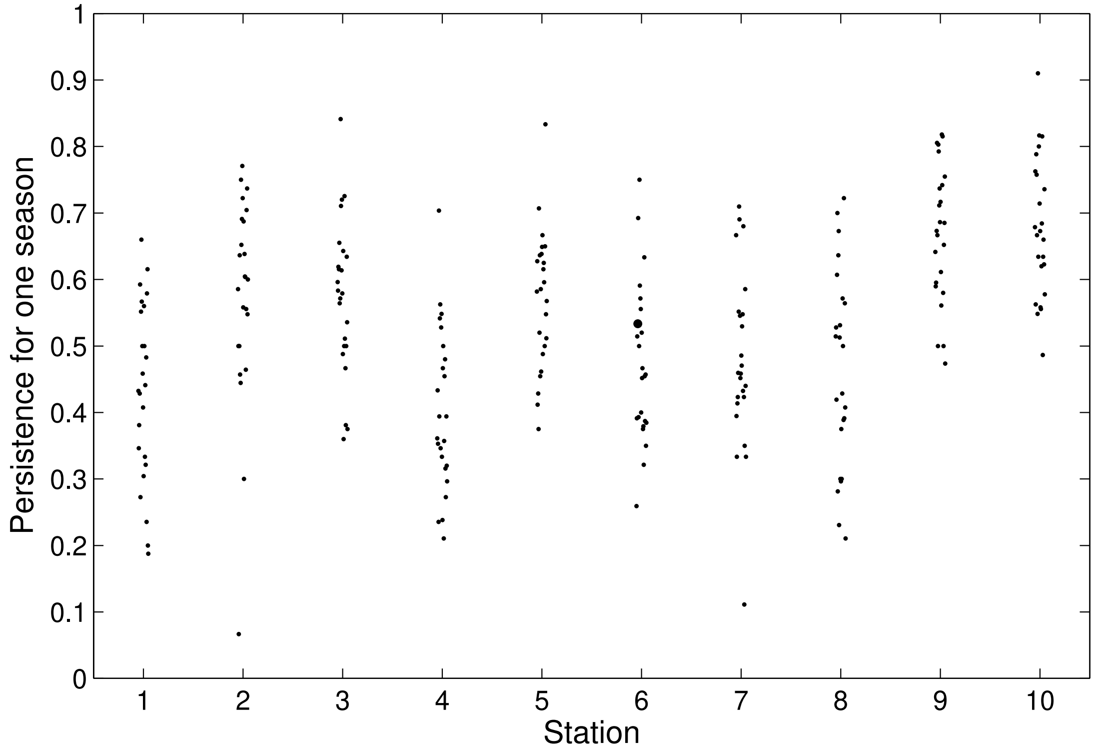

Figure 8.1: Precipitation persistence for Ceará set. Each point corresponds to a value for one season.

the models on. Leave-six-out cross-validation provides a reasonable compromise.

For now, we will concern us only with the HMM-CI[2]. The results of the leave-six-out cross-validation are shown in Table 8.1. (Robertson et al. (2004) contains a slightly different version of this table.) To obtain the numbers in Table 8.1, one needs to follow these steps:

1. Train a model on each of the cross-validation training sets.

2. Simulate a large number of sequences from each of the learned models, and then compute statistics (the correlation for each station pair and the persistence for

[2]For a detailed analysis of HMM-CIs and NHMM-CIs applied to modeling of the Ceará set, see Robertson et al. (2004). We present only a small part of that analysis in this section.

each station) for each simulated set.

3. For each of the left-out sets (as determined by cross-validation):
   - Compute the correlation and the persistence; compute the absolute difference between these statistics and the same statistics of the corresponding simulated data.
   - Compute the scaled log-likelihood (of the corresponding model) and the average classification error of prediction by leaving out a single observation and then predicting it with the corresponding model.
4. Average the results across the data sets.

A K-state M-variate binary HMM-CI has $K^2 - 1 + M * K$ free parameters, $K^2 + 10K - 1$ for this set. There is no clear improvement in numbers for $K > 4$, so the numbers suggest training the model with $K = 4$. The numbers for the out-of-sample evaluation for the leave-one-out cross-validation (Table 8.2) are a little better than their counterparts for the leave-six-out cross-validation case. More accurate models are the result of training on the 23 sequences (versus 18 with leave-six-out cross-validation). Again, however, there is no significant improvement past $K = 4$, so both cross-validations point to the same conclusion for K.

By overimposing the probabilities of precipitation for each station in each hidden state over the map of the region, we can visualize the resulting hidden (weather) states. Figure 8.4 shows these states for 4-state HMM-CI trained on all 24 data sequences. The transition probabilities for this 4-state model are provided in Table 8.3. The transition matrix suggests that both the "wet" state (state 1) and the "dry" state (state 2) are somewhat persistent as indicated by values of corresponding self-transition parameters. The transitions between these two states occur mostly through

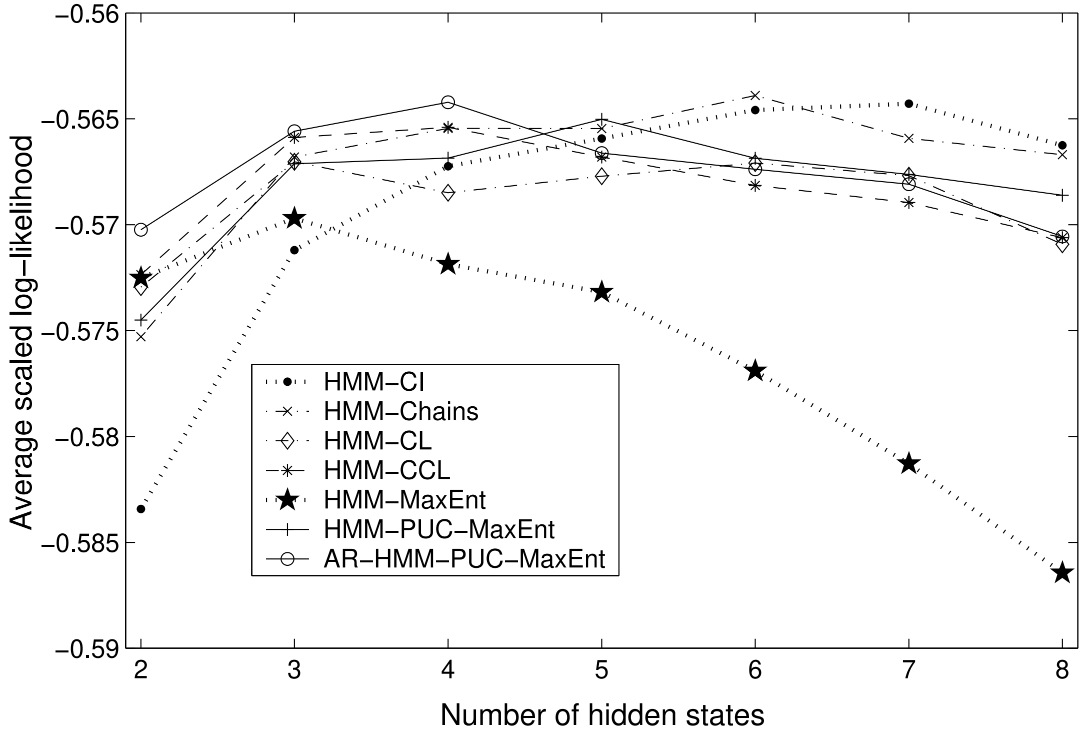

Figure 8.2: Ceará data: average out-of-sample log-likelihood obtained by a leave-one-out cross-validation for various models across a number of hidden states.

the other two "transitional" states with state 3 occurring more frequently than state 4 (Robertson et al., 2004).

Figure 8.2 compares the average out-of-sample log-likelihoods for different models under leave-one-out cross-validation. Figure 8.3 compares the expected accuracy of correlation (top) and persistence (bottom) according to the leave-six-out cross-validation. The measurements are quite noisy as training sets are small, especially for models with large numbers of parameters (e.g., AR-HMM-PUC-MaxEnt, models with large numbers of hidden states). For all models, except for HMM-CI, no significant improvement is noticeable for $K > 3$. According to the log-likelihood, HMM-MaxEnt clearly overfits[3] while the rest of the models have comparable performances, with HMM-CI requiring larger number of hidden states to match the performance of other models. As expected, the models with spatial structure perform the best according to the difference in correlation while the models with temporal structure perform the best according to the difference in persistence. Thus, HMM-CCL or AR-HMM-PUC-MaxEnt with $K = 3$ hidden states would provide a solution with good performance according to the statistics of interest. However, even without temporal links between the stations, HMMs can capture statistics of the data; the data simulated from a 3-state HMM-CL and a 4-state HMM-CI models trained on all 24 years of the data has statistics resembling the original data set. Probabilities of precipitation for individual stations obtained from the data simulated from either of 4-state HMM-CI or 3-state HMM-CL match the corresponding probabilities for the historical data exactly (Figure 8.6 left). Between-station correlations are also quite closely matched, especially for 3-state HMM-CL (Figure 8.5). However, the persistence numbers are somewhat underestimated (Figure 8.6 right) leading to an underestimation of lengths of the wet spells.

[3]This is not surprising since HMM-MaxEnt requires $M\left(M-1\right)/2$ free parameters per hidden state. Hughes et al. (1999) suggest reducing the number of free parameters by parametrizing them using the geo-locations of the stations.

Table 8.1: Performance of HMM-CIs evaluated by leave-six-out cross-validation.

Number of hidden states	Average out-of-sample scaled log-likelihood	Average out-of-sample absolute difference in correlation (over all stations pairs)	Average out-of-sample absolute difference in persistence	Average out-of-sample classification accuracy
$K = 2$	-0.5855	0.0622	0.0796	69.22
$K = 3$	-0.5742	0.0516	0.0682	71.26
$K = 4$	-0.5709	0.0464	0.0643	71.53
$K = 5$	-0.5683	0.0461	0.0650	71.75
$K = 6$	-0.5685	0.0453	0.0645	71.96
$K = 7$	-0.5697	0.0447	0.0644	71.71
$K = 8$	-0.5699	0.0438	0.0632	71.85

Table 8.2: Performance of HMM-CIs evaluated by leave-one-out cross-validation.

Number of hidden states	Number of free parameters	Average out-of-sample scaled log-likelihood	Average out-of-sample classification accuracy
$K = 2$	23	-0.5834	69.15
$K = 3$	38	-0.5712	71.71
$K = 4$	55	-0.5672	71.96
$K = 5$	74	-0.5659	72.09
$K = 6$	95	-0.5646	72.38
$K = 7$	118	-0.5643	72.46
$K = 8$	143	-0.5662	71.94

Table 8.3: Transition parameters for 4-state HMM-CI trained on Ceará data.

From	To			
	State 1	State 2	State 3	State 4
State 1	0.6988	0.0115	0.1652	0.1245
State 2	0.0238	0.6824	0.1583	0.1355
State 3	0.1570	0.1451	0.5884	0.1095
State 4	0.2352	0.1918	0.1602	0.4128

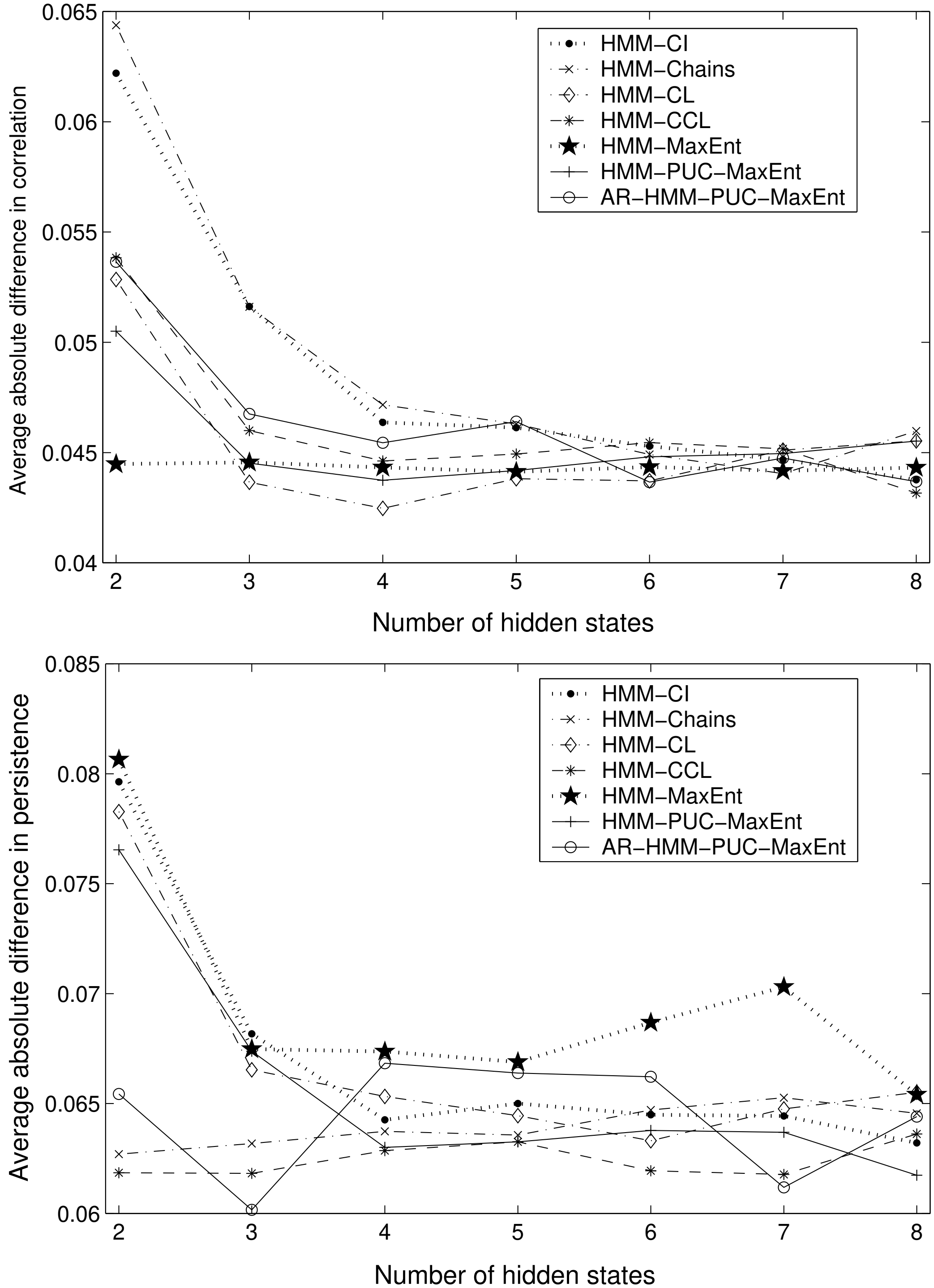

Figure 8.3: Ceará data: average out-of-sample absolute difference between the statistics of the simulated data and the left-out data as determined by leave-six-out cross-validation for correlation (top) and persistence (bottom).

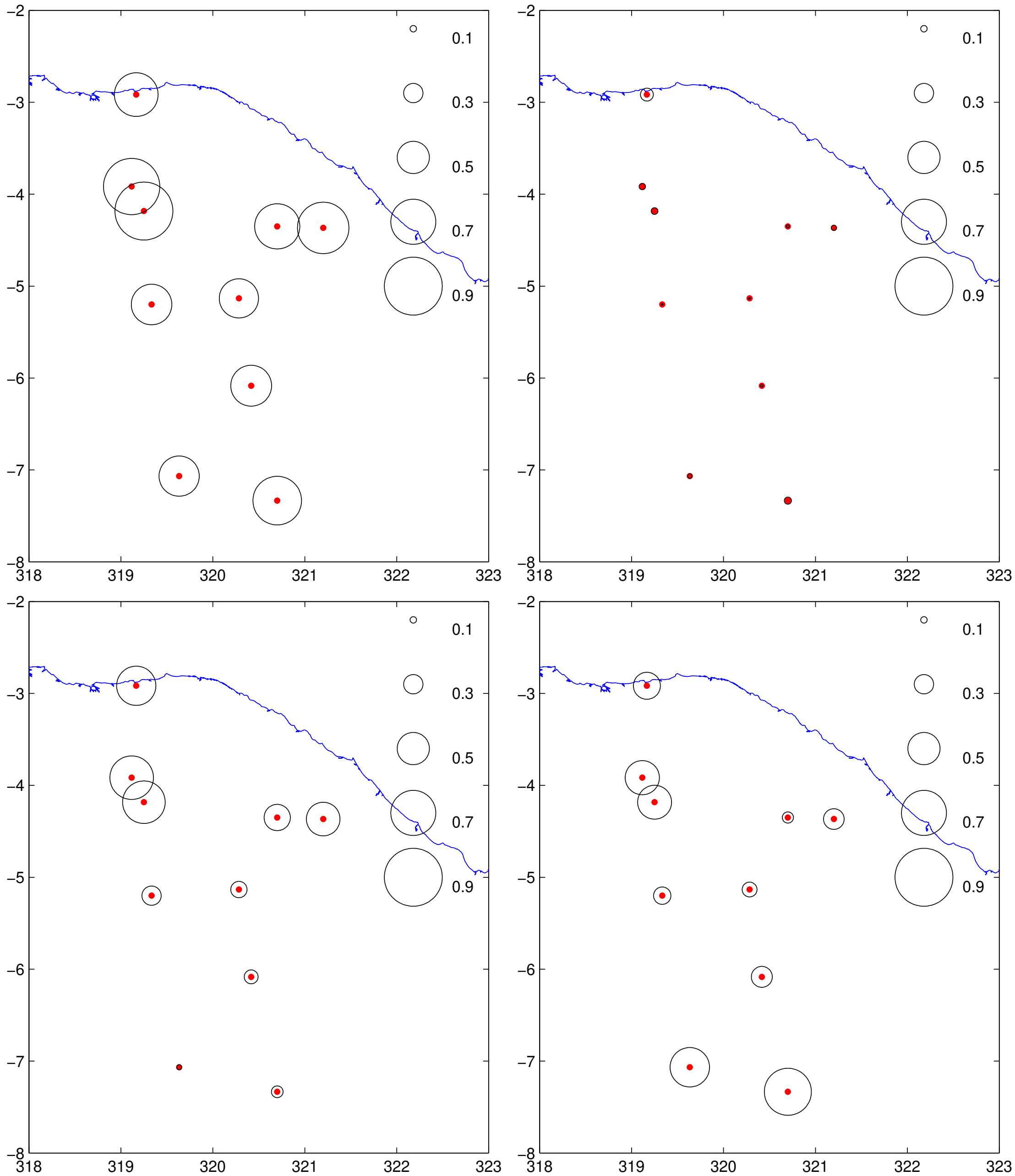

Figure 8.4: Weather states obtained by training 4-state HMM-CI on the Ceará set. Circle radii indicate the precipitation probability for each station given the state.

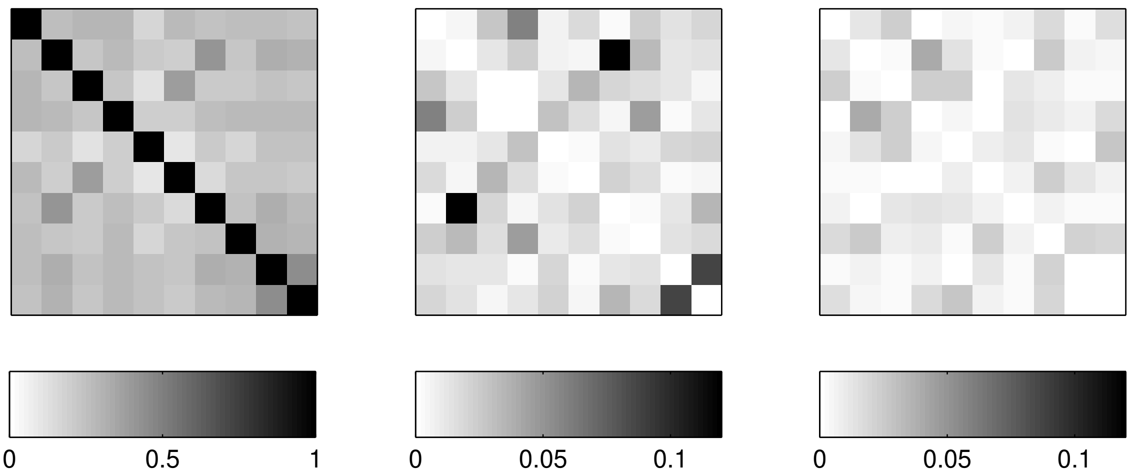

Figure 8.5: Ceará data: correlation matrix of rainfall daily occurrence (left), absolute difference between the correlation matrix of the data and the correlation of the data simulated from a 4-state HMM-CI (middle) and a 3-state HMM-CL (right). The average absolute difference for non-diagonal entries is 0.020 for HMM-CI and 0.010 for HMM-CL.

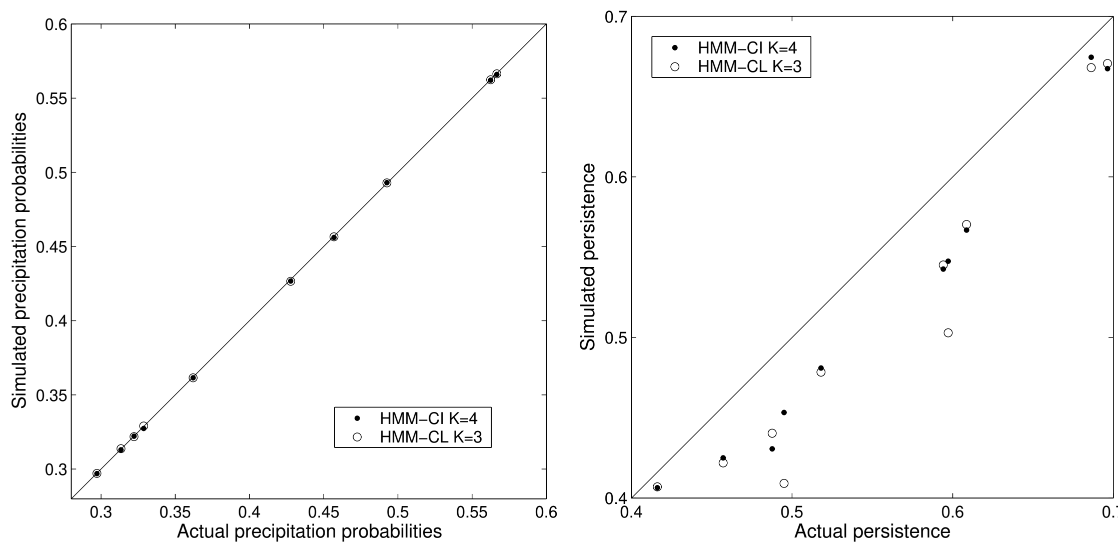

Figure 8.6: Ceará data: scatter plot of the precipitation probabilities (left) and the persistence (right) for each station of the actual data versus the simulated data from various models. Straight lines correspond to $y = x$.

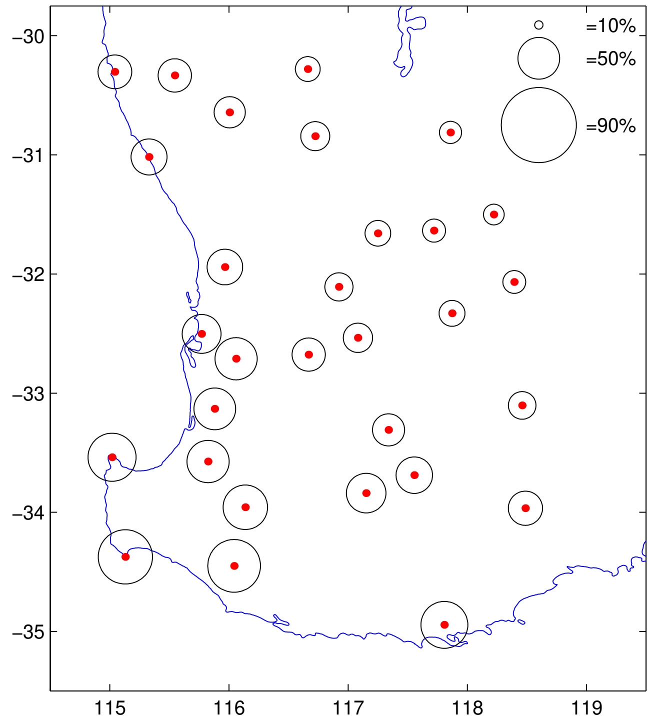

Figure 8.7: Stations in the Southwestern Australia region. Circle radii indicate marginal probabilities of rainfall ($> 0.3mm$) at each location.

8.3.2 Southwestern Australia

In the experiments below we use data collected from 30 stations over 15 184-day winter seasons (May–October, 1978–1992).[4] Figure 8.7 shows the network of stations.

We use leave-one-out cross-validation to evaluate the predictive accuracy of the models on this data set. For evaluation, we use two different criteria. We compute the scaled log-likelihood and the average classification error for seasons not in the training data. The models considered are the HMM with conditional independence (HMM-CI), the HMM with dependence on the previous observation for the same station

[4]This data has been collected by the Commonwealth Bureau of Meteorology of Australia and provided to us by Dr. Stephen P. Charles of CSIRO (Commonwealth Scientific and Industrial Research Organisation) Land and Water, Australia.

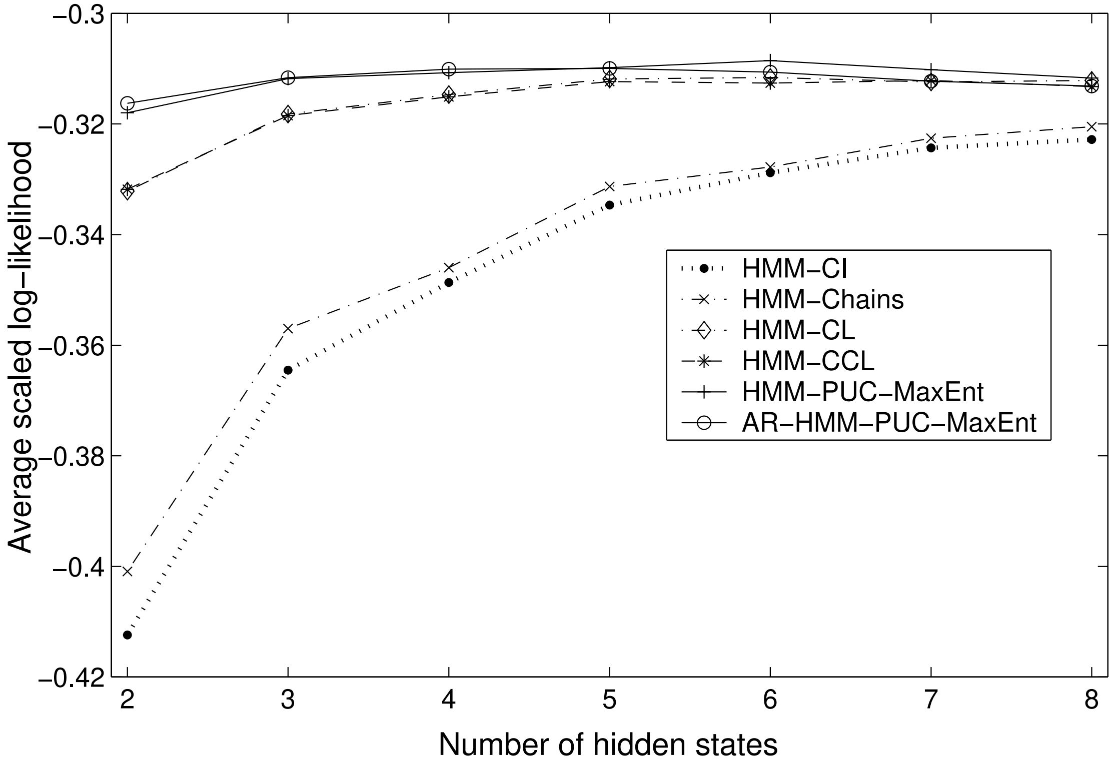

Figure 8.8: Southwestern Australia data: average out-of-sample log-likelihood obtained by a leave-one-out cross-validation for various models across a number of hidden states.

(HMM-Chains), the HMM with Chow-Liu tree emissions (HMM-CL), the HMM with conditional Chow-Liu forest emissions (HMM-CCL), the HMM with product of univariate conditional exponentials (HMM-PUC-MaxEnt), and the HMM with product of univariate conditional exponentials allowing dependencies on the previous observation (AR-HMM-PUC-MaxEnt). The comparison plot of scaled out-of-sample log-likelihood for these models across different K is shown in Figure 8.8.[5] The plot clearly shows that models allowing additional temporal links are performing the same as their counterparts without the temporal links. The plot also suggests that the data contains a large number of strong spatial dependencies as the HMM-CI model requires

[5]As was discussed in Chapter 7, scaled log-likelihood falls within $[-\ln 2, 0)$.

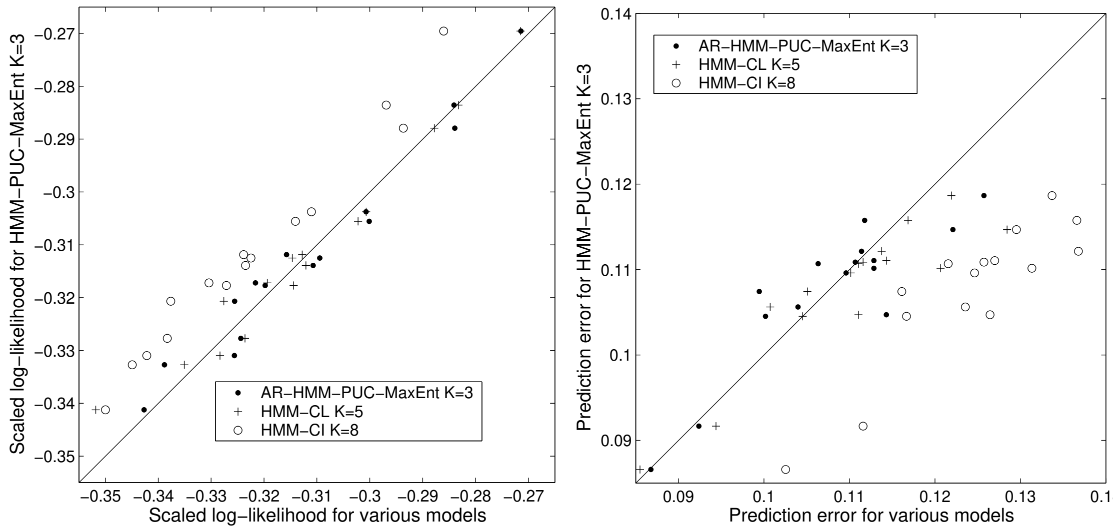

Figure 8.9: Southwestern Australia data: scatter plot of scaled log-likelihoods (left) and average prediction error (right) obtained by leaving one-winter-out cross-validation. Straight lines correspond to $y = x$.

a large number of states to capture the data. This is explained by the fact that the Australian stations are very close spatially. Since temporal dependencies are not as strong, they do not add much to the model. HMM-CL and HMM-CCL have very similar performance on this data (Kirshner et al., 2004). Since HMM-PUC-MaxEnt slightly outperforms HMM-CL, we can infer that the tree structure of the states of the HMM-CL may be oversimplifying the dependency between the observations for different stations. Note that HMM-PUC-MaxEnt requires more parameters than HMM-CL for the same K. However, we can use HMM-PUC-MaxEnt with smaller K (e.g., $K = 3$) and still obtain a model with a comparable fit to the data.

The scatter plots in Figure 8.9 show the scaled log-likelihoods and average classification errors for the models on the left-out sets. The classification errors are computed as follows: a single observation for one station is withheld, and then its value, given the rest of the left-out set, is classified (predicted) by the model learned on the training test; the error is then averaged over all observations in the hold-out set. K is chosen subjectively according to the scaled log-likelihood for each model as we want to select a model providing a good fit with the smallest number of hidden states. The y-axis is the performance of the HMM-PUC-MaxEnt model, and the x-axis represents the performance of the other models (shown with different symbols). Higher implies better performance for log-likelihood (on the left) and worse for error (on the right). A 3-state HMM-PUC-MaxEnt (249 free real-valued parameters[6,7]) is performing about as well as both a 3-state AR-HMM-PUC-MaxEnt (279 free real-valued parameters[8,7]) and a 5-state HMM-CL (319 free real-valued parameters[6]), and noticeably better than an 8-state HMM-CI (303 free parameters). The statistics of the data simulated from such model (3-state HMM-PUC-MaxEnt) trained on all 15 seasons of data resemble that of the actual data (Figures 8.10 and 8.11), with exact match for precipitation occurrence probabilities, good match for correlations, and a slight underestimation of rainfall persistence.

Examples of the Chow-Liu tree structures learned by the model are shown in Figure 8.12 for the 5-state HMM-CL model trained on all 15 years of Southwestern Australia data. The states learned by the model correspond to a variety of wet and dry spatial patterns. The tree structures are consistent with the meteorology and topography of the region (Hughes et al., 1999). Winter rainfall over SW Australia is large-scale and frontal, impacting the southwest corner of the domain first and foremost. Hence, the tendency for correlations between stations along the coast during moderately wet weather states. Interesting correlation structures are also identified

[6] This number does not include the parameters need to specify the features.

[7] Number of parameters is for the model with the highest log-likelihood trained on the set with all data sequences.

[8] This number does not include the parameters needed to specify the features and the parameters needed to specify the probabilities for the first observation in each sequence.

in the north of the domain even during dry conditions. For comparison, we consider other HMMs not modeling additional time dependencies, HMM-CI and HMM-PUC-MaxEnt, with weather states shown in Figures 8.13 and 8.14, respectively. All three models have easily identifiable wet (top left) and dry (bottom) states with the same rainfall probabilities for individual stations across the models in these states. HMM-CL and HMM-PUC-MaxEnt have very similar stations' precipitation probabilities for the other three states; the edge structures also exhibit similarities. On the other hand, HMM-CI's other weather states do not match well to states of either HMM-CL or HMM-PUC-MaxEnt. HMM-CCL's weather states (Figure 8.15) differ very little from states of HMM-CL in stations' precipitation and mildly in spatial edges with very few temporal edges present in HMM-CCL; this indicates that the dependence in the data is mostly spatial.

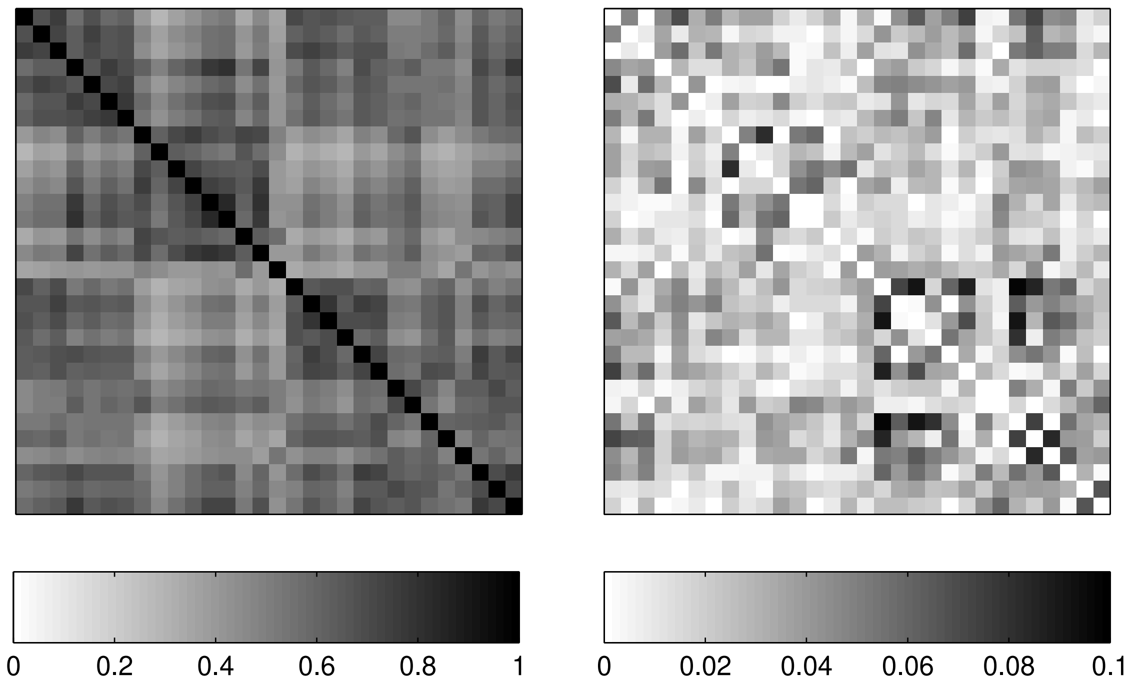

Figure 8.10: Southwestern Australia data: correlation matrix of rainfall daily occurrence (left), absolute difference between the correlation matrix of the data and the correlation of the data simulated from a 3-state HMM-PUC-MaxEnt (right). The average absolute difference for non-diagonal entries is 0.025.

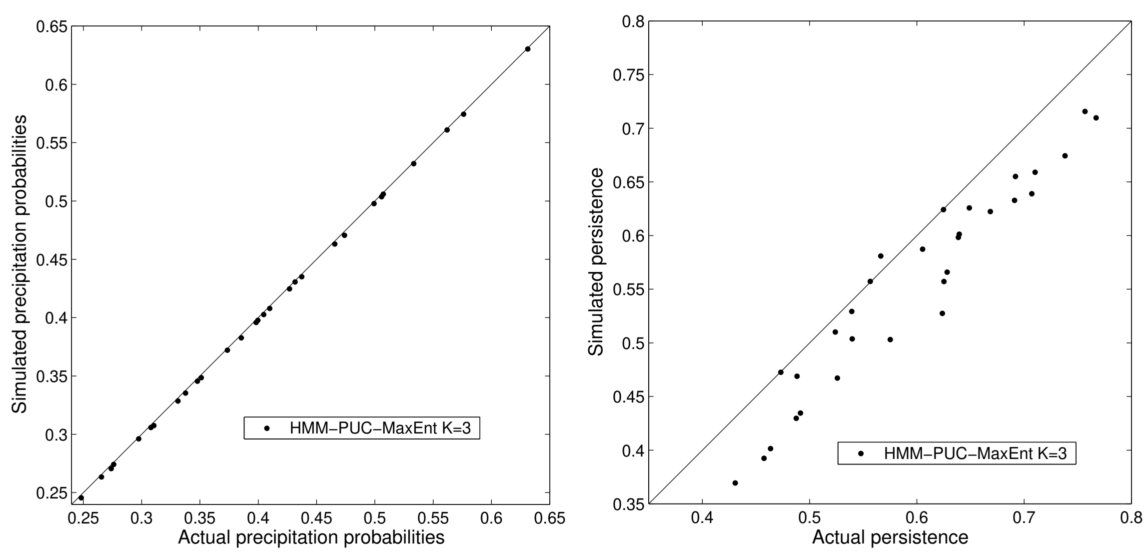

Figure 8.11: Southwestern Australia data: scatter plot of the precipitation probabilities (left) and the persistence (right) for each station of the actual data versus the simulated data from various models. Straight lines correspond to $y = x$.

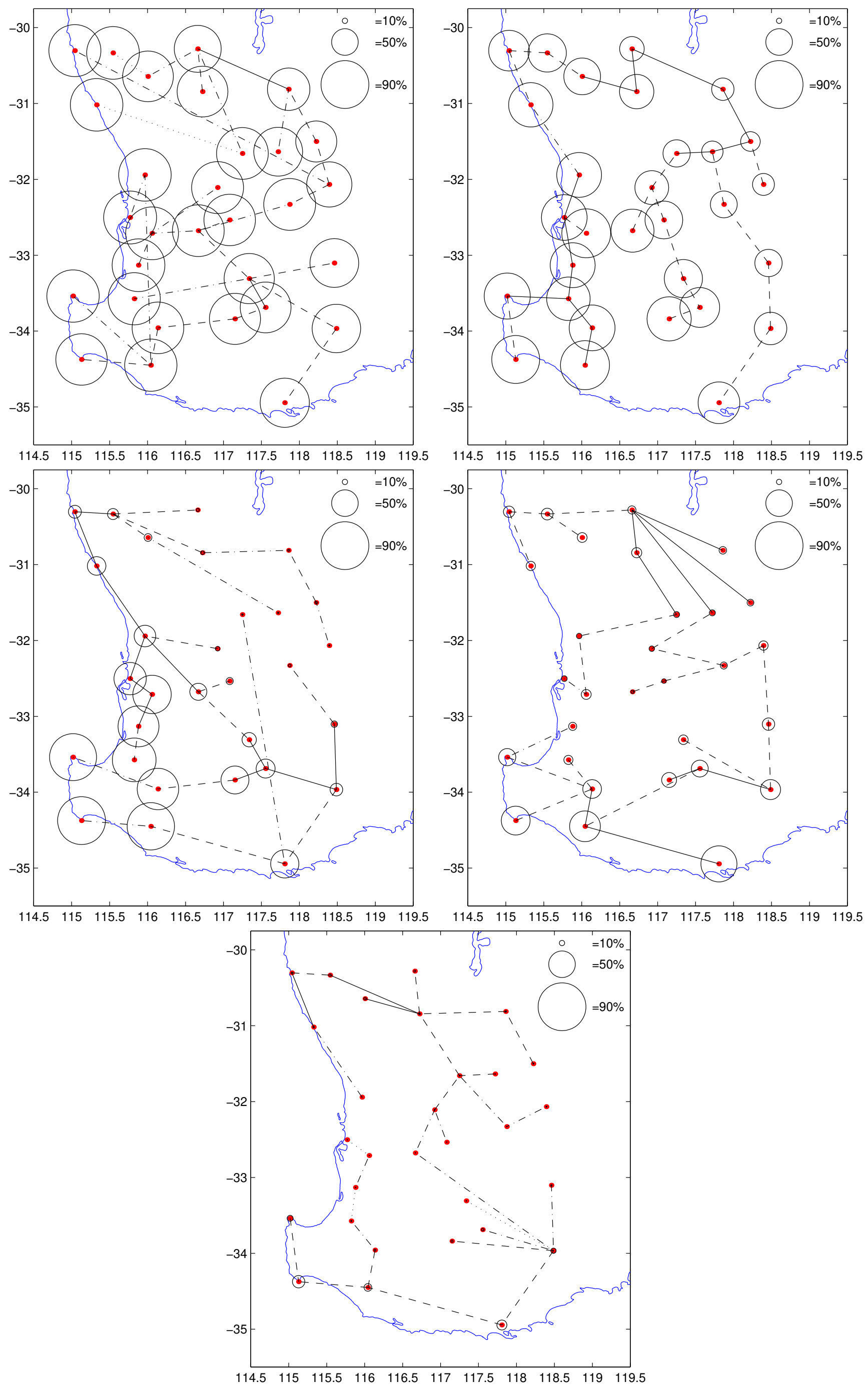

Figure 8.12: Graphical interpretation of the hidden states for a 5-state HMM-CL trained on Southwestern Australia data. Circle radii indicate the precipitation probability for each station given the state. Lines between the stations indicate the edges in the graph while different types of lines indicate the strength of mutual information of the edges.

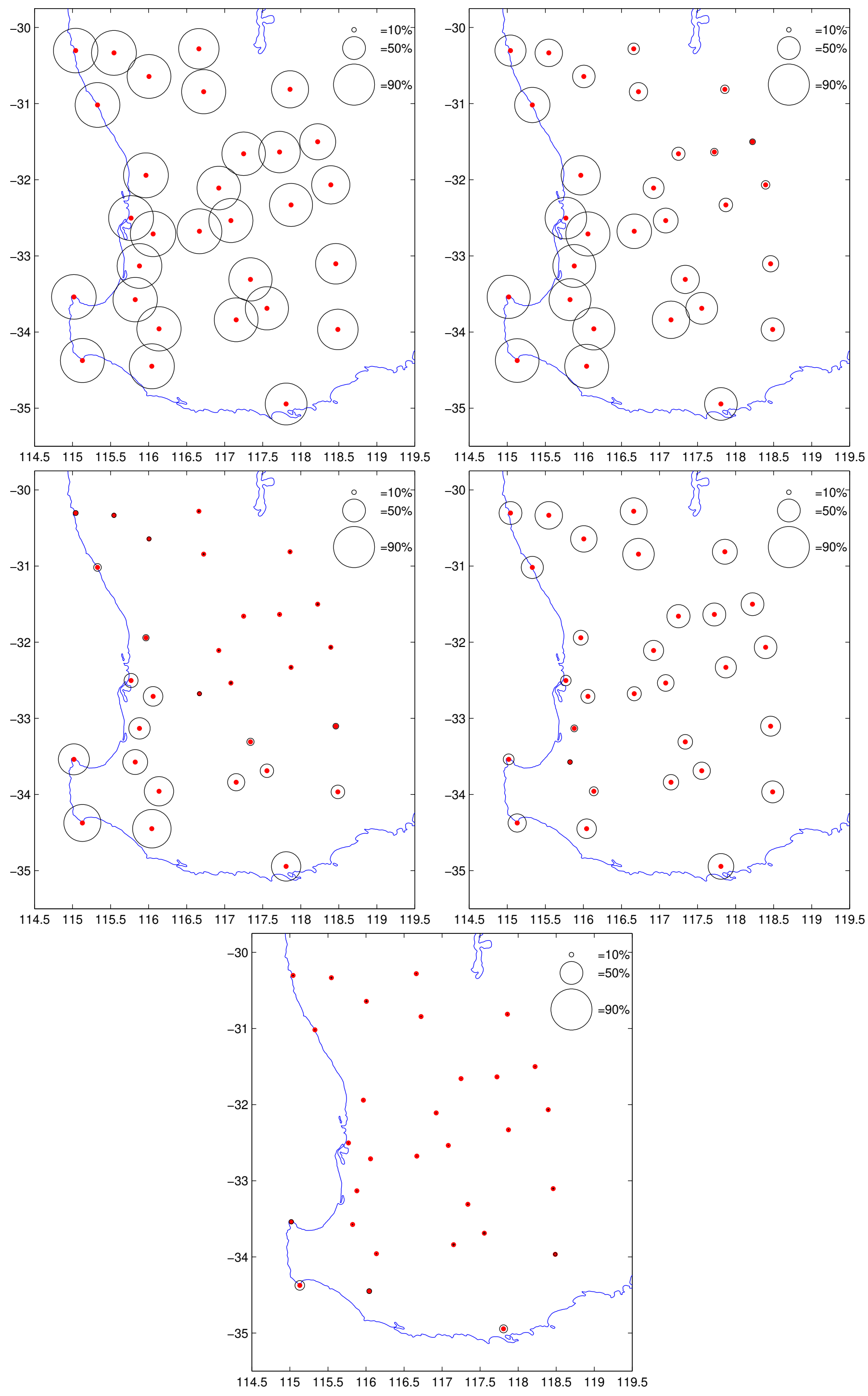

Figure 8.13: Graphical interpretation of the hidden states for a 5-state HMM-CI model trained on Southwestern Australia data. Circle radii indicate the precipitation probability for each station given the state.

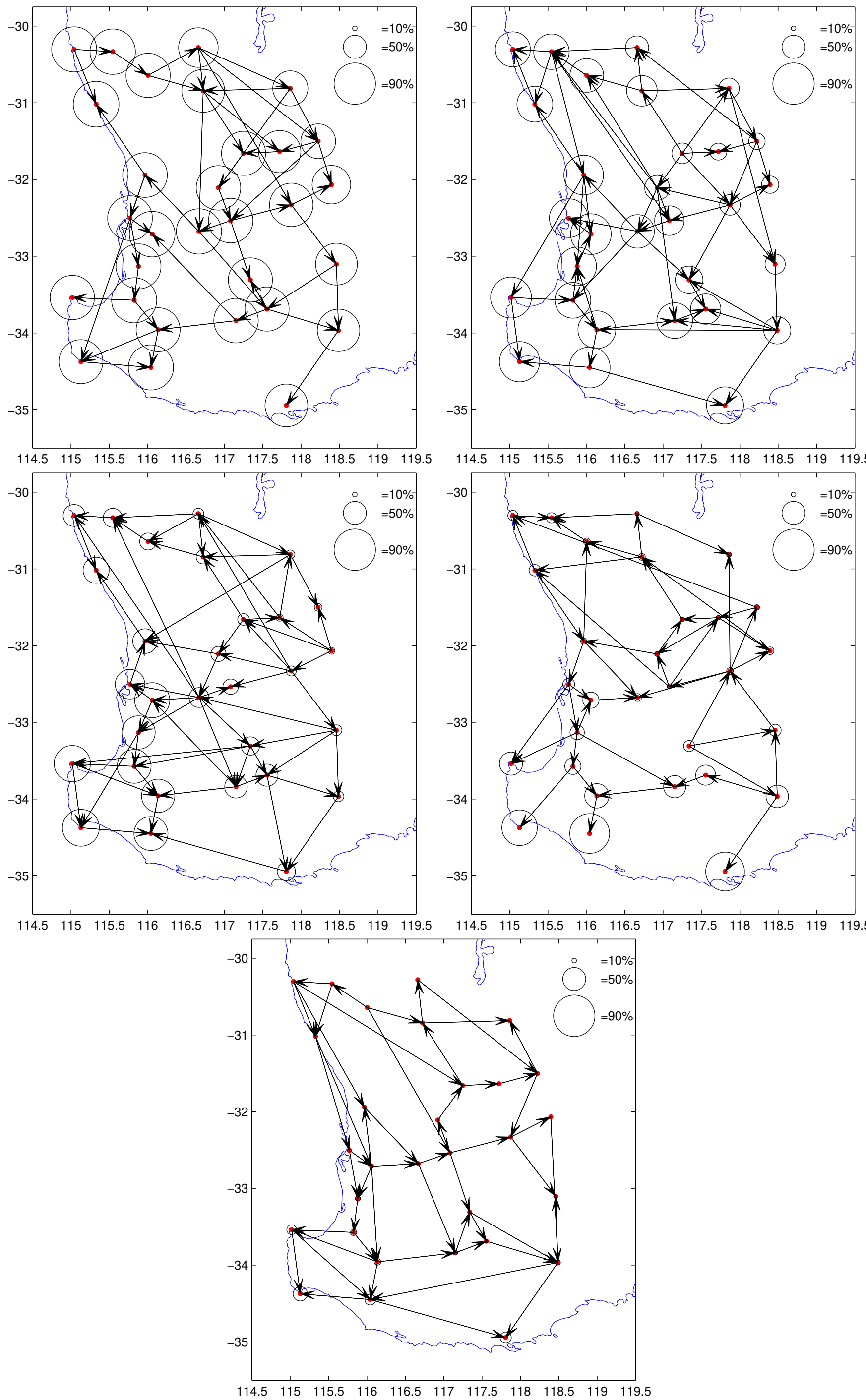

Figure 8.14: Graphical interpretation of the hidden states for a 5-state HMM-PUC-MaxEnt model trained on Southwestern Australia data. Circle radii indicate the precipitation probability for each station given the state. Links between the stations indicate the edges in the graph.

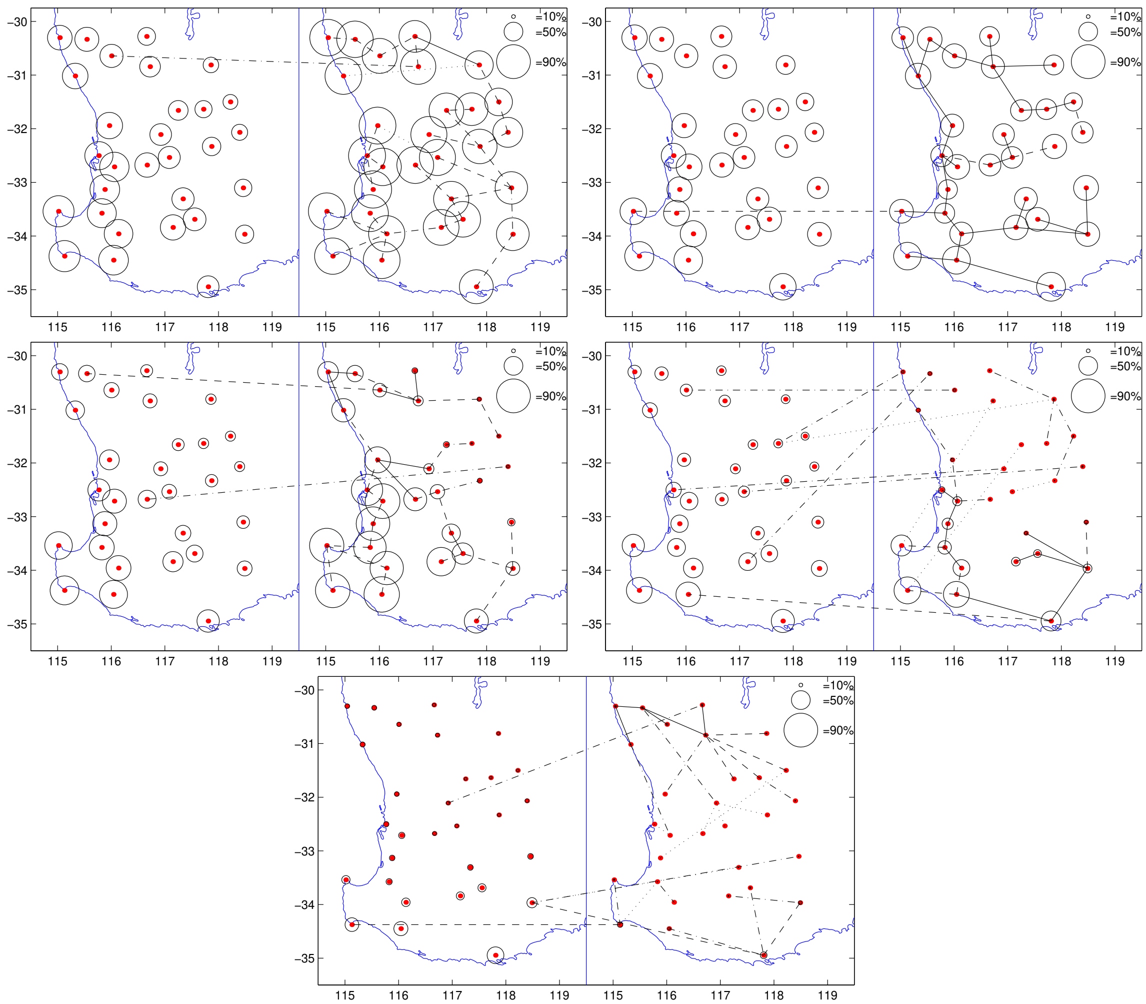

Figure 8.15: Graphical interpretation of the hidden states for a 5-state HMM-CCL trained on Southwestern Australia data. Circle radii indicate the precipitation probability for each station given the state. Lines between the stations indicate the edges in the graph while different types of lines indicate the strength of mutual information of the edges. The left side of the plot corresponds to observations $\boldsymbol{R}_{t-1}$ while the right side to $\boldsymbol{R}_t$.

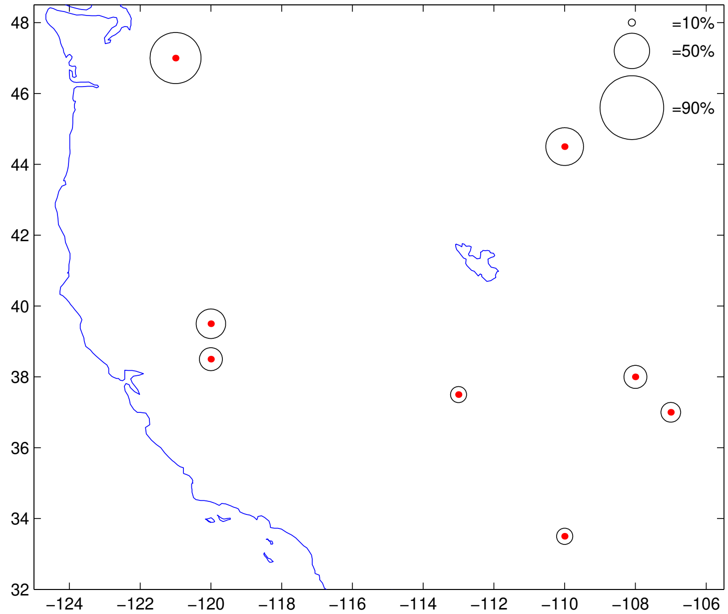

Figure 8.16: Stations in the Western U.S. region. Circle radii indicate marginal probabilities of rainfall ($> 0mm$) at each location.

8.3.3 Western United States

The Western U.S. data was collected from 8 sparsely located stations over 39 90-day winter seasons (December–February, 1951–1990). Figure 8.16 shows the network of stations.

The analysis is performed similarly to that of the Southwestern Australia data. As before, we use leave-one-out cross-validation and the same two different criteria: the scaled log-likelihood and the average classification error for seasons not in the training data. Since the stations are sparsely located, we do not expect strong spatial dependencies; thus, models with additional temporal links are expected to perform better than the models without them. In addition to the models considered in Section 8.3.2 (HMM-CI, HMM-Chains, HMM-CL, HMM-CCL, HMM-PUC-MaxEnt, and AR-HMM-PUC-MaxEnt), we consider several stateless models with temporal dependencies. These are degenerative cases ($K = 1$) of the HMM-Chains, HMM-CCL, and AR-HMM-PUC-MaxEnt; these models are essentially dynamic Bayesian networks (DBNs) with transitions defined as CI, conditional Chow-Liu forest, and PUC-MaxEnt distributions, respectively. We will denote these models as DBN Chains, DBN CCL, and DBN AR-PUC-MaxEnt.[9] The comparison plot of scaled log-likelihood for these models across different K is shown in Figure 8.17. In contrast to Southwestern Australia data, HMMs with additional temporal links significantly outperform HMMs without them. It is also worth noting that AR-HMM-PUC-MaxEnt achieves high out-of-sample log-likelihood even for small ($K = 2, 3$) number of hidden states.

The scatter plots in Figure 8.18 show the scaled log-likelihoods and classification errors for the models on the left-out sets. K is chosen subjectively according to the scaled log-likelihood for each model as we want to select a model providing a good fit with the smallest number of hidden states. The y-axis is the performance of the AR-HMM-PUC-MaxEnt model, and the x-axis represents the performance of the other HMMs (shown with different symbols). Higher implies better performance for log-likelihood (on the left) and worse for error (on the right). The number of hidden states for the models is chosen to correspond to high average values of out-of-sample log-likelihood and so that the number of free parameters for different types of models is about the same. These plots lead us to the same conclusion as Figure 8.17, that models with temporal links fit this data better as all three models with explicitly model temporal dependencies outperform the two models without them.

[9] DBN Chains is just a model with independent Markov chains. Kirshner et al. (2004) used DBN CCL under the name of chain Chow-Liu forests (CCLF).

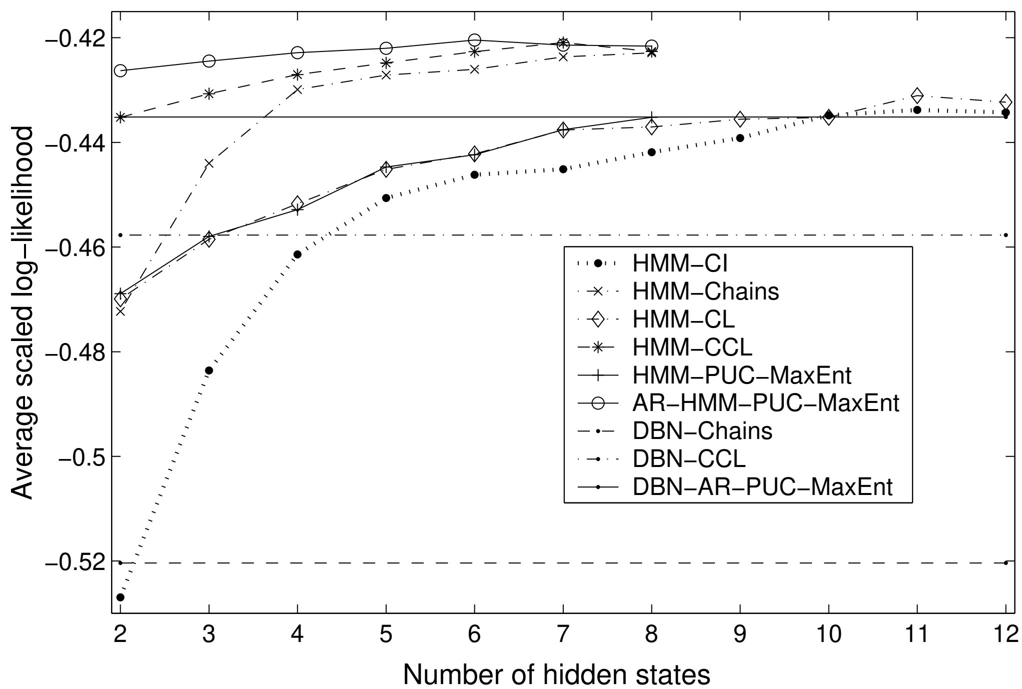

Figure 8.17: Western U.S. data: average out-of-sample log-likelihood for various models across a number of hidden states. Straight lines correspond to scaled log-likelihoods of DBNs with transition determined by the corresponding conditional distribution.

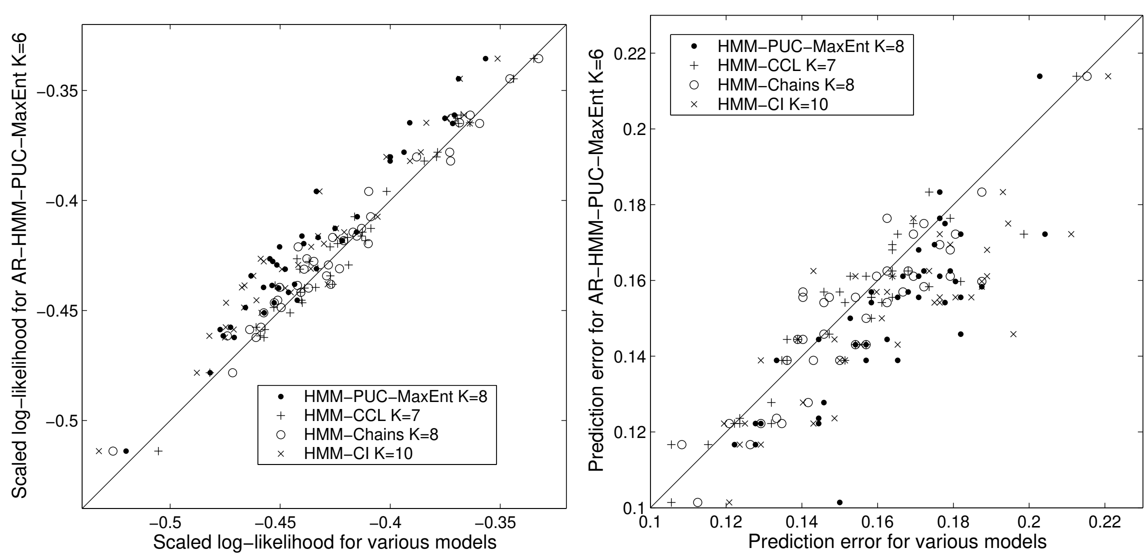

Figure 8.18: Western U.S. data: scatter plot of scaled log-likelihoods (left) and average prediction error (right) obtained by leaving one-winter-out cross-validation. Lines correspond to $y = x$.

Examples of the conditional Chow-Liu forest structures learned by the model are shown in Figure 8.19 for the 7-state HMM-CCL model trained on all 39 years of data. The states learned by the model correspond to a variety of wet and dry spatial and temporal patterns. Corresponding plots of the AR-HMM-PUC-MaxEnt with $K = 6$ are shown in Figure 8.20. The weather states match up remarkably well except for the third row states of the HMM-CCL being merged into one (third row left) state of the 6-state AR-HMM-PUC-MaxEnt. Both models generate data with statistics very similar to the historical data, matching the precipitation probabilities exactly (Figure 8.22 left), while closely matching both persistence (Figure 8.22 right) and correlation (Figure 8.21).

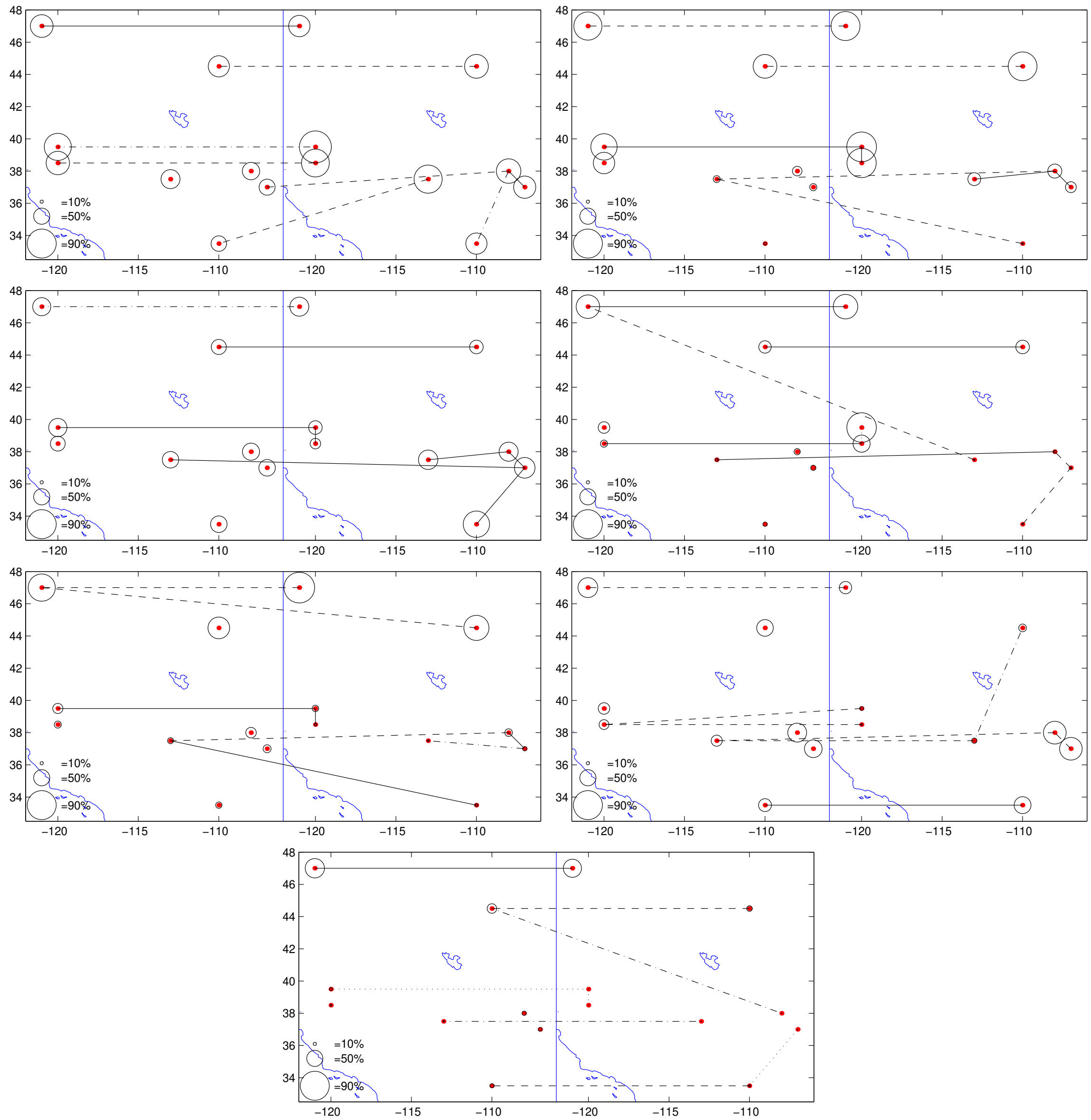

Figure 8.19: Graphical interpretation of the hidden states for a 7-state HMM-CCL trained on Western U.S. data. Circle radii indicate the precipitation probability for each station given the state. Lines between the stations indicate the edges in the graph while different types of lines indicate the strength of mutual information of the edges. The left side of the plot corresponds to observations $\boldsymbol{R}_{t-1}$ while the right side to $\boldsymbol{R}_t$.

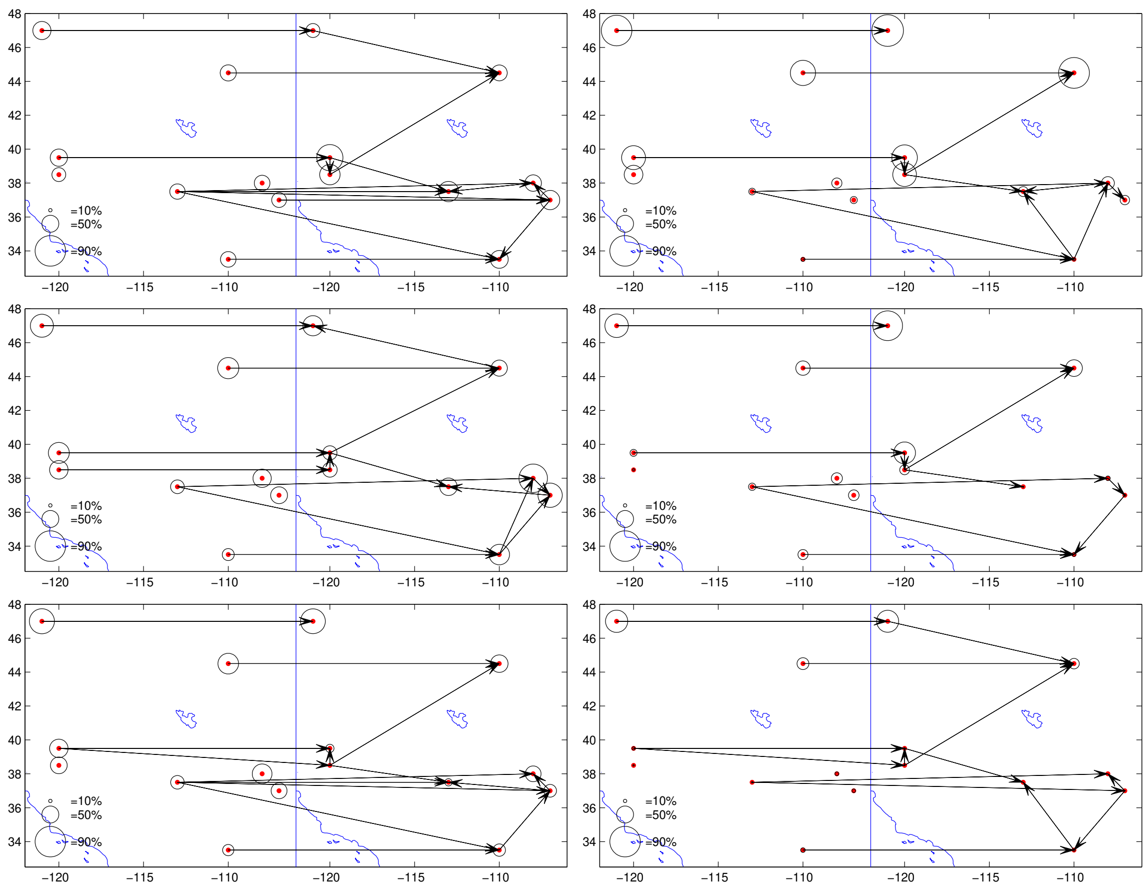

Figure 8.20: Graphical interpretation of the hidden states for a 6-state AR-HMM-PUC-MaxEnt trained on Western U.S. data. Circle radii indicate the precipitation probability for each station given the state. Edges between the stations indicate the edges in the graph. The left side of the plot corresponds to observations $\boldsymbol{R}_{t-1}$ while the right side to $\boldsymbol{R}_t$.

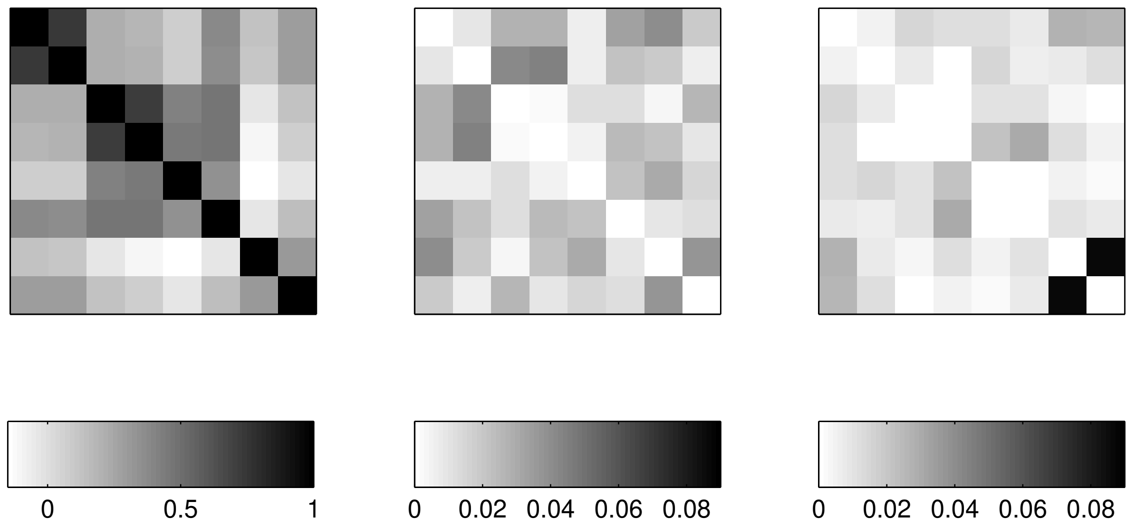

Figure 8.21: Western U.S. data: correlation matrix of rainfall daily occurrence (left), absolute difference between the correlation matrix of the data and the correlation of the data simulated from a 6-state AR-HMM-PUC-MaxEnt (middle) and a 7-state HMM-CCL (right). The average absolute difference for non-diagonal entries is 0.019 for AR-HMM-PUC-MaxEnt and 0.013 for HMM-CCL.

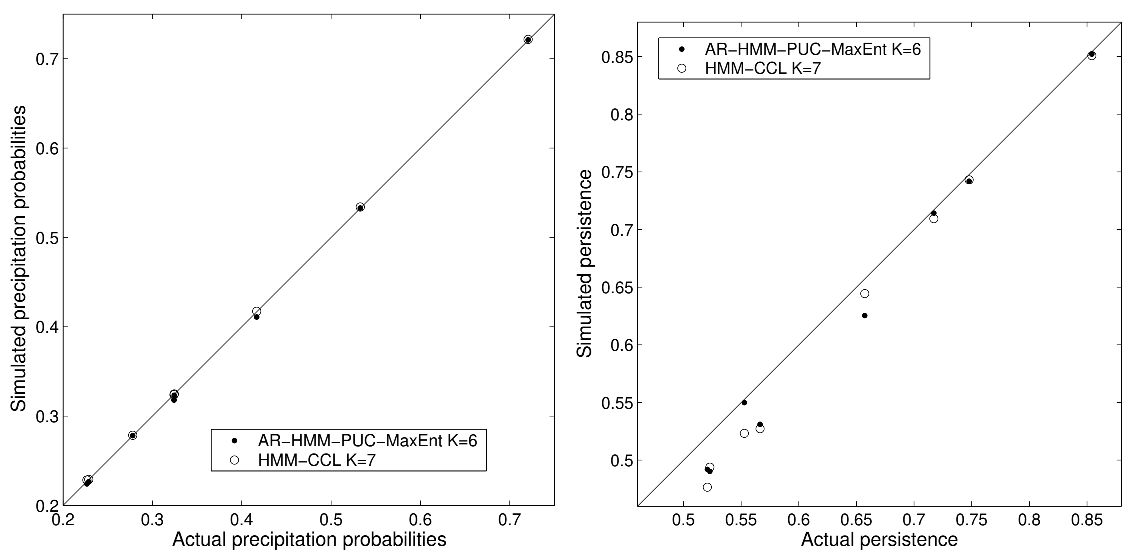

Figure 8.22: Western U.S. data: scatter plot of the precipitation probabilities (left) and the persistence (right) for each station of the actual data versus the simulated data from various models. Straight lines correspond to $y = x$.

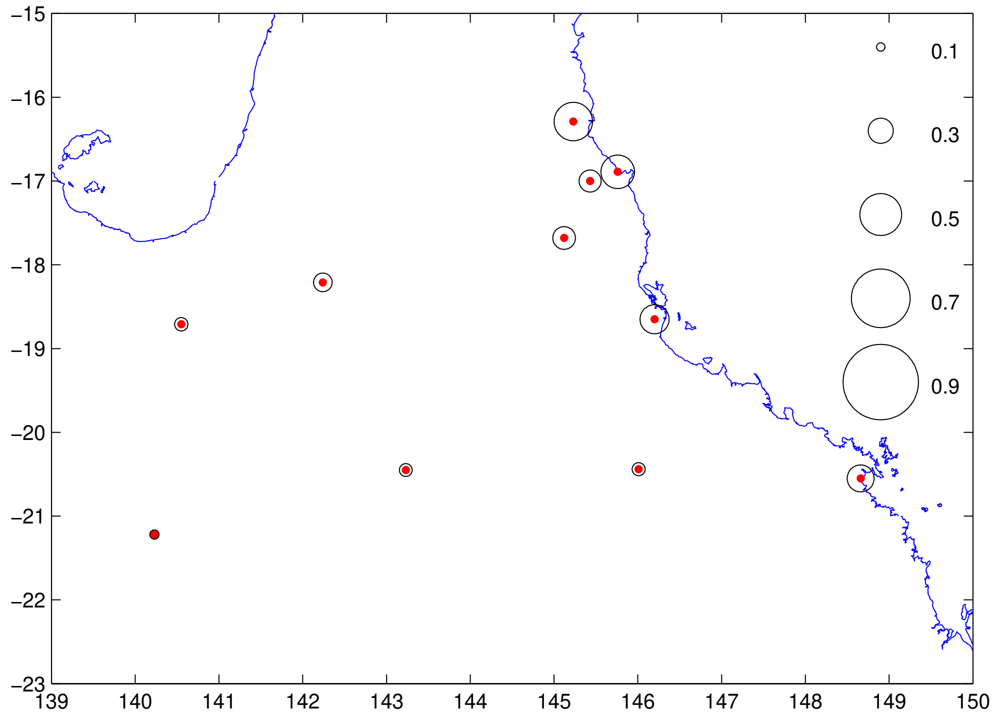

Figure 8.23: Stations in the Queensland (Northeastern Australia) region. Circle radii indicate marginal probabilities of rainfall ($\geq 1mm$) at each location.

8.3.4 Queensland (Northeastern Australia)

This data was collected from 11 stations over 40 197-day austral summer periods (October 1–April, 1958–1998). This data set is derived from the Patched Point Dataset (PPD) (Jeffrey et al., 2001). Figure 8.23 shows the network of stations. Judging just by the locations of the stations, the data set should be fit best by models allowing for both temporal and spatial dependencies (HMM-CCL and AR-HMM-PUC-MaxEnt) as the stations are sparsely located except for the cluster in the north.

For this set, we use leave-eight-out (five-fold) cross-validation to evaluate the models. In addition to the scaled log-likelihood, we would also consider out-of-sample

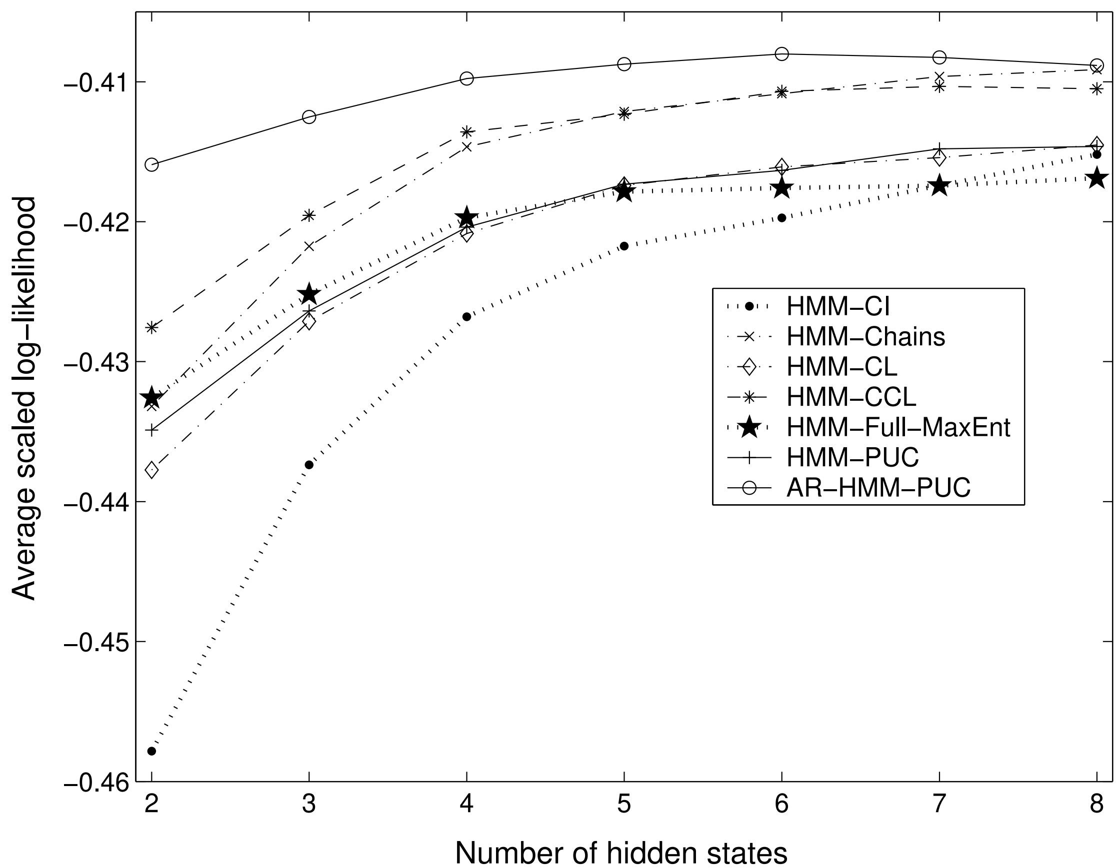

Figure 8.24: Queensland data: average out-of-sample log-likelihood for various models across different number of hidden states.

average absolute difference in spatial (pairwise linear) correlation and out-of-sample average absolute difference in persistence. In addition to six HMM models considered for previous regions, we will also consider HMM-Full-MaxEnt. The comparison plot of scaled log-likelihood for these models across different K is shown in Figure 8.24. Models with additional temporal links perform noticeably better than the models without them with AR-HMM-PUC-MaxEnt having the highest average out-of-sample scaled log-likelihood. HMM-Full-MaxEnt appears to overfit as the models with many fewer parameters outperform it for larger values of K.

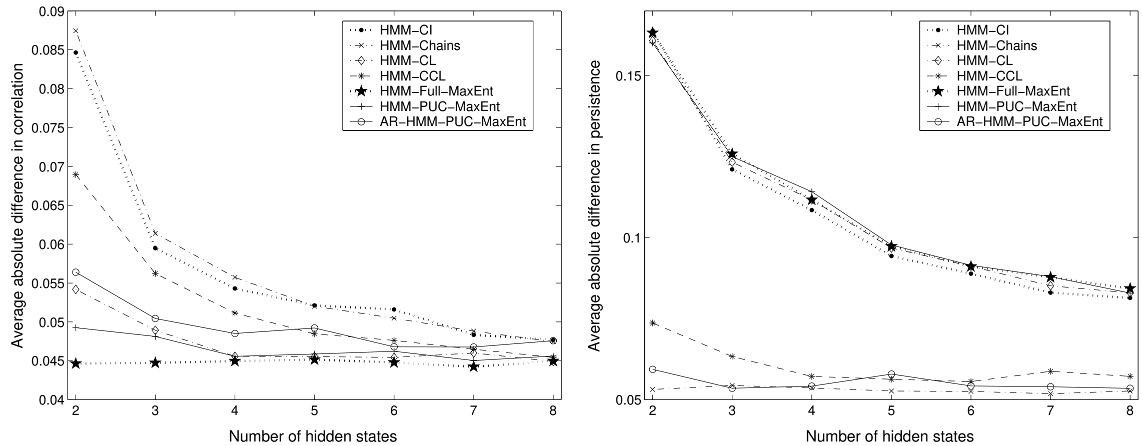

Figure 8.25: Queensland data: average out-of-sample absolute difference of correlation (left) and persistence (right) of the simulated data with the left-out data statistics.

Figure 8.25 compares the correlation (left) and the persistence (right) statistics of the data simulated from the models and the data. The correlation measures the multivariate dependencies while the persistence measures temporal dependencies. As expected HMM-Full-MaxEnt matches the correlation the best as it explicitly models all pairwise dependencies; also, HMM-Chains matches persistence the best for a similar reason. AR-HMM-PUC-MaxEnt appears to be the best of the set as it performs almost as good as HMM-Chains in terms of persistence while matching most of the spatial correlation.

Structures of the hidden states of a 4-state AR-HMM-PUC-MaxEnt learned on all 40 years of data are shown in Figure 8.26. As was hypothesized earlier, the states contain a large number of both temporal and spatial edges. The data simulated from this model matches the precipitation occurrence probabilities of historical data for all stations (Figure 8.28 left), correlations between most pairs of stations (Figure 8.27), while underestimating slightly the persistence (Figure 8.28 right).

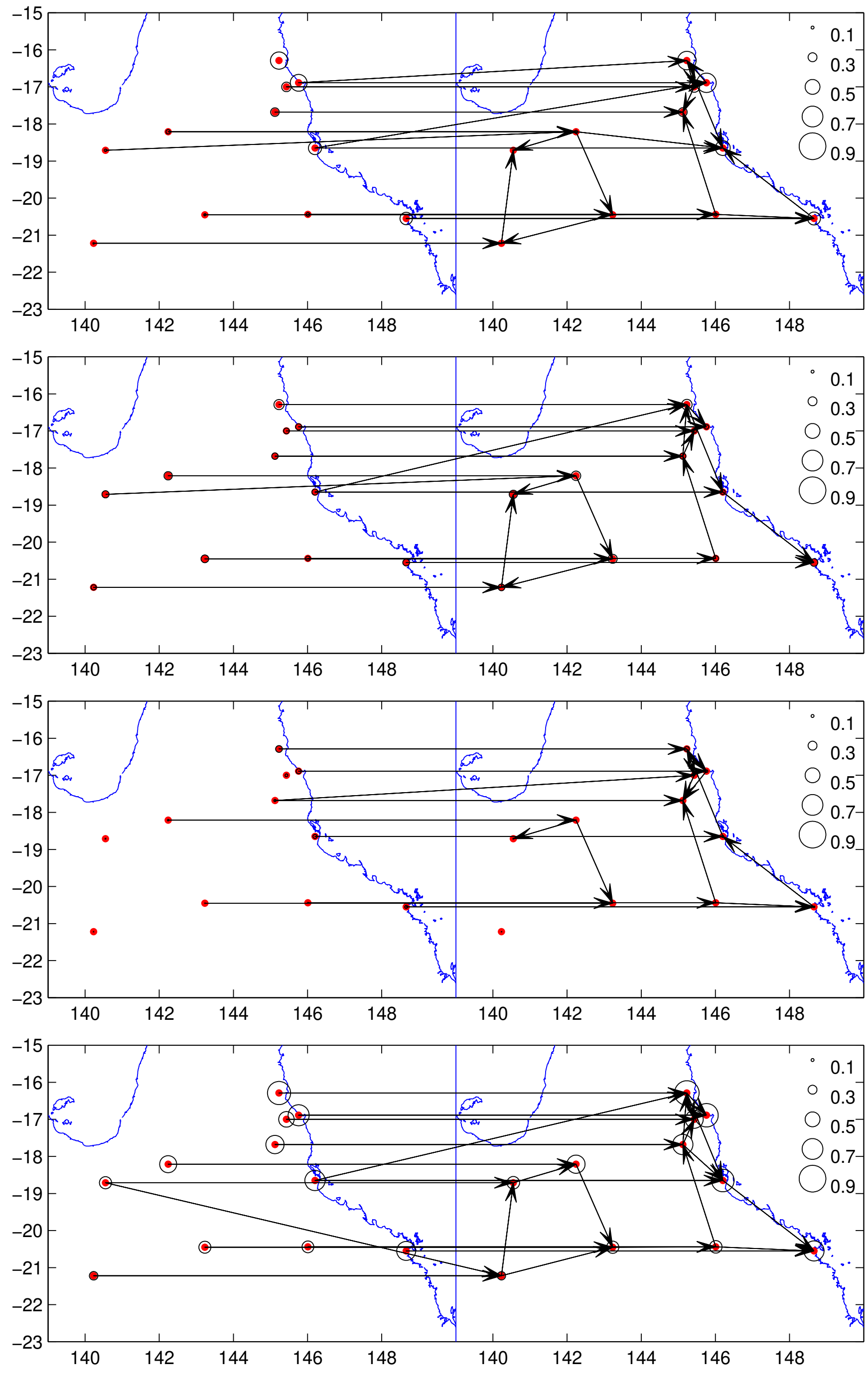

Figure 8.26: Graphical interpretation of the hidden states for a 4-state AR-HMM-PUC-MaxEnt trained on Queensland data. Circle radii indicate the precipitation probability for each station given the state. Edges between the stations indicate the edges in the graph. The left side of the plot corresponds to observations $\boldsymbol{R}_{t-1}$ while the right side to $\boldsymbol{R}_t$.

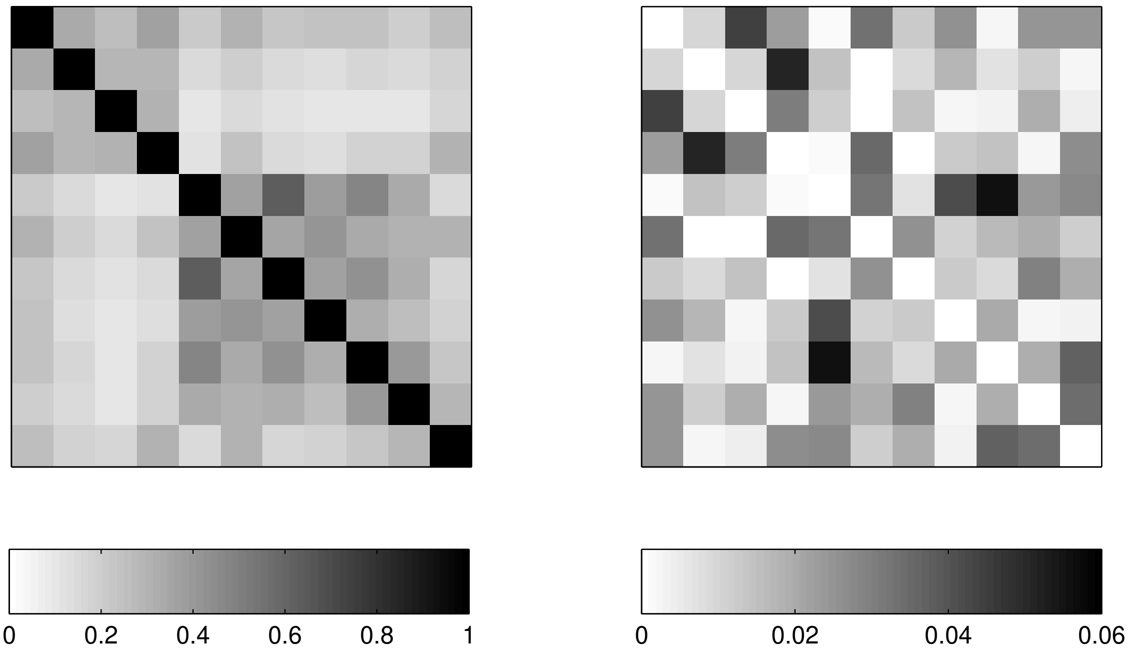

Figure 8.27: Queensland data: correlation matrix of rainfall daily occurrence (left), absolute difference between the correlation matrix of the data and the correlation of the data simulated from a 4-state AR-HMM-PUC-MaxEnt (right). The average absolute difference for non-diagonal entries is 0.013.

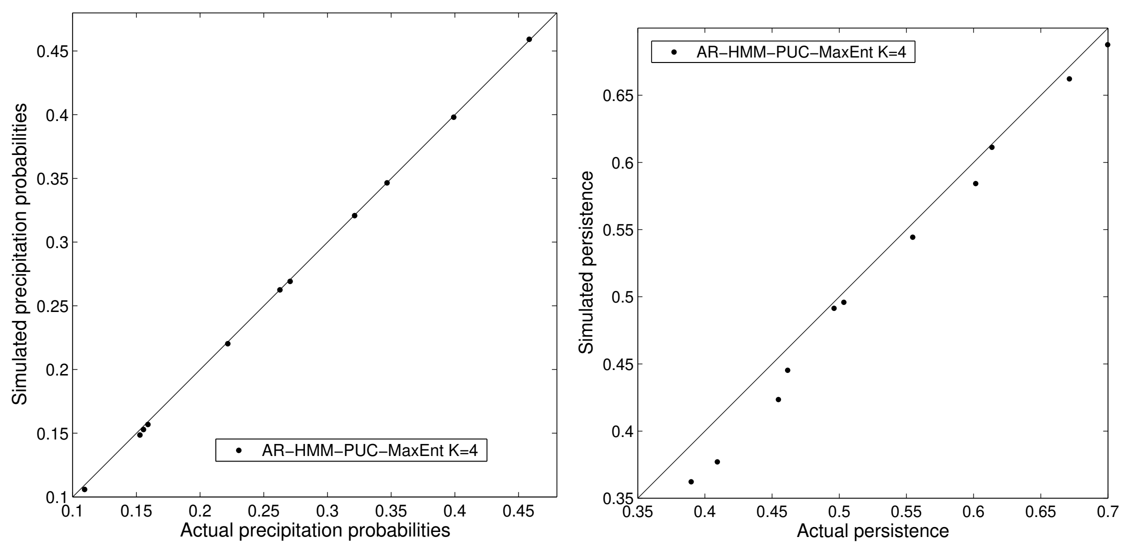

Figure 8.28: Queensland data: scatter plot of the precipitation probabilities (left) and the persistence (right) for each station of the actual data versus the simulated data from various models. Straight lines correspond to $y = x$.

8.4 Summary

We applied the models from Chapter 6 to four daily precipitation occurrence data sets of various strengths of multivariate and temporal dependencies. For each set, the best model depends on whether spatial or temporal dependence is stronger. However, the models that learn both temporal and multivariate links perform well in all cases; from these models, AR-HMM-PUC-MaxEnt fits the data a little better at the expense of larger number of parameters. However, AR-HMM-PUC-MaxEnt can be used with smaller number of hidden states cutting the number of parameters while reducing the fit only marginally.

Novel models allowing to learn multivariate and temporal links (e.g., HMM-CCL, AR-HMM-PUC-MaxEnt) show a lot of promise for modeling multivariate vector time series. We believe that their advantages over simpler models would be even more evident on data sets with vectors of higher dimensionality as these models can extract the structure corresponding to the strongest dependencies in the data.

Chapter 9

Summary and Future Directions

In this thesis we introduced a number of new models and corresponding training algorithms for modeling multivariate time series or just multivariate data. The majority of the results address categorical data although some of the models can be extended, sometimes trivially, to real-valued domains. We have empirically shown that given sufficient data, the models can correctly estimate the parameters. Finally, we applied the models to the domain of multi-site precipitation occurrence modeling and obtained superior predictive accuracy compared to similar models and better modeled persistence and spatial correlation. In this concluding chapter of the thesis, we summarize the main contributions and outline future directions and challenges.

9.1 List of Contributions

1. **Introduced Chow-Liu trees for use with HMMs and developed a corresponding parameter estimation algorithm.** The use of Chow-Liu trees in conjunction with hidden Markov models allows efficient preservation of some of the multivariate dependencies in the multivariate time series data. Due to

the efficiency of the Chow-Liu tree algorithm, the Chow-Liu model can be used as a probability distribution within each hidden state with only a reasonable increase in computational complexity. The amount of data need to train the model is not overly large as only pairwise variable interactions are considered in each hidden state.

2. **Extended Chow-Liu trees to the conditional distribution case and developed a corresponding AR-HMM together with their parameter estimation algorithms.** The conditional Chow-Liu (CCL) forests allow us to learn the tree-structure of the conditional distribution with Chow-Liu trees on the joint distributions as its special case. In most cases, the asymptotic computational complexity of the conditional Chow-Liu model is the same as that of the original Chow-Liu model, so the extended model is very efficient. This computational efficiency allows the use of CCL forests with AR-HMMs; the main advantage is that the new hybrid HMM-CCL model can select automatically the strongest dependencies from both temporal and multivariate dependencies thus improving the quality of the model. As with HMMs with Chow-Liu trees, the combined model can often accurately estimate the parameters with only a limited amount of data available.

3. **Introduced a model allowing joint and conditional distributions with structures less restrictive than trees; proposed a structure-learning algorithm for the model.** The product of univariate conditional exponentials (PUC-MaxEnt) model allows us to learn a distribution with an arbitrary Bayesian network on categorical valued data. This model can be thought of as a generalization of CCL forests although the structure learning and parameter estimation for this model can be significantly more involved than that in the

CCL forests case. One of the most attractive features of this model (as compared to other multivariate exponential models) is the clarity of the parameter estimation and the ease of its implementation. Since the model is represented as a product of univariate conditional distributions, the parameter estimation can be broken down into independent updates of univariate conditional exponential distributions for each variable and can likely be parallelized. This model also avoids summing over large number of variables in the parameter estimation. To estimate the structure of the model, we proposed a greedy algorithm that uses an approximation to the log-likelihood improvement achieved by adding each candidate edge.

4. **Incorporated PUC-MaxEnt with HMMs.** The new model can be used to model time series with dense multivariate or temporal dependence structures. This model, however, requires a substantial amount of data and is computationally quite involved.

5. **Analyzed the Gaussian counterpart of the Chow-Liu tree model.** Since pairwise variable dependencies in multivariate normal distributions are encoded in its covariance matrix, we studied the structure of this matrix for tree-structured multivariate Gaussian distributions. We have derived a closed form expression for the covariance matrix, detailed the computational complexities of the operations for this tree-structured matrix, outlining several possible advantages of this model over the saturated model. Finally, we incorporated this new model into an HMM for modeling of multivariate real-valued time series.

6. **Introduced a new model for efficient modeling of multi-site precipitation amounts.** This HMM-CI based model allows us to model multi-site

precipitation amounts without learning a categorical data model first. Initial experimental results suggest that the model produces realistic precipitation simulation (Robertson et al., in preparation).

7. **Applied the model to several historical precipitation occurrence data sets.** The study compared how well the models fit the data based on log-likelihood and spatial and temporal statistics of the data. The data sets varied in the amount of data, and strengths of bivariate temporal and spatial dependencies.

8. **Conducted an empirical simulation study of how well some of the described models can estimate the true parameters.** The experiments tested how well can a model recover the parameters given various amounts of simulated data. The experiment was repeated for models with different entropies.

9. **Developed a software suite for parameter estimation and analysis for the models in the thesis.** I developed Unix-based C++ software for data modeling using HMMs with multivariate distributions in hidden states. The software is publicly available, and most of the routines and features are being used by our scientific collaborators, e.g., at IRI and CSIRO.

9.2 Future Directions

1. **Models for multivariate real-valued non-Gaussian time series.** The vast majority of the models described in this thesis deal with multivariate categorical data. However, a large portion of the real-world data does not fit this criterion; precipitation data is by nature non-negative real-valued (non-

Gaussian as a result) and is modeled as binary data because very few reliable real-valued models are available. The desired models should preserve univariate marginals of the data while capturing multivariate dependencies. Our attempts to incorporate tree-based models for precipitation amounts were fruitless as modeling bivariate exponentials (components of marginals) turned out to be difficult. One possible future direction is the use of *copulas* (Joe, 1997; Nelsen, 1998), multivariate cumulative density functions preserving univariate marginals. These functions may have the desired characteristics to be used with HMMs in modeling multivariate non-Gaussian real-valued time series.

2. **Feature selection for inputs to be used with non-homogeneous HMMs.** The NHMMs described in Section 3.1.2 require input variables. The sets of potential input variables can be very large. For example, in precipitation modeling the input sets are usually the variables simulated from a GCM. These variables are usually computed on a grid at regular time intervals, so there may be thousands if not millions of potential variables. Since only a small number of them are potentially useful (and non-relevant variables can degrade the performance), the importance of selecting the right set of variables is significant. Variable selection from a large set of candidate input variables in a NHMM is an open problem.

3. **Handling missing data.** Most of the real-world data has missing values. For example, precipitation records have gaps, especially the ones from less developed regions. One option is just to throw the sequences with missing data away, but in the presence of only a moderate amount of complete data, this method may impact the accuracy of the models. More desirable is to change the learning algorithm to allow for missing data. For example, HMM-CI can be

modified in a straightforward manner to allow missing observations. Methods for incorporating missing observations with other models discussed in this thesis need to be developed. One possibility is to use a Bayesian approach and to infer distributions over the missing data in the Bayesian framework.

4. **Bayesian framework for HMMs for multivariate time series.** There are advantages to using a Bayesian framework for modeling multivariate time series. One of the reasons, Bayesian framework allows us to incorporate expert knowledge into the parameter estimation and model selection. For example, for multi-site precipitation application, we may use geographical and topographical information in model selection and estimation, e.g., a spatial prior on which edges could be included in the model.

Bibliography

D. J. Allcroft and C. A. Glasbey. A latent Gaussian Markov random-field model for spatiotemporal rainfall disaggregation. *Journal of the Royal Statistical Society Series C (Applied Statistics)*, 52(4):487–498, 2003.

S. Arngborg, D. G. Corneil, and A. Proskurowski. The complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic and Discrete Methods*, 8(2):277–284, April 1987.

J. R. Ashford and R. R. Sowden. Multi-variate probit analysis. *Biometrics*, 26(3): 535–546, 1970.

F. R. Bach and M. I. Jordan. Thin junction trees. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 569–576, Cambridge, MA, 2002. MIT Press.

P. Baldi and S. Brunak. *Bioinformatics*. MIT Press, 2nd edition, 2001.

A. Bardossy and E. J. Plate. Space-time model for daily rainfall using atmospheric circulation patterns. *Water Resources Research*, 28:1247–1259, 1992.

L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Annals of Mathematical Statistics*, 37(6):1554–1563, 1966.

L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in statistical analysis of probabilistic functions of Markov chains. *Annals of Mathematical Statistics*, 41(1):164–171, February 1970.

E. Bellone. *Nonhomogeneous Hidden Markov Models for Downscaling Synoptic Atmospheric Patterns to Precipitation Amounts*. PhD thesis, Department of Statistics, University of Washington, Seattle, Washington, 2000.

E. Bellone, J. P. Hughes, and P. Guttorp. A hidden Markov model for downscaling synoptic atmospheric patterns to precipitation amounts. *Climate Research*, 15(1): 1–12, May 15 2000.

S. Bengio and Y. Bengio. An EM algorithm for asynchronous input/output hidden Markov models. In L. Xu, editor, *International Conference On Neural Information Processing*, pages 328–334, Hong-Kong, 1996.

Y. Bengio. Markovian models for sequential data. *Neural Computing Surveys*, 2: 129–162, 1999.

Y. Bengio and P. Frasconi. An input/output HMM architecture. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 427–434. MIT Press, Cambridge, MA, 1995.

A. Berchtold. The double chain Markov model. *Communications in Statistics – Theory and Methods*, 28(11):2569–2589, 1999.

A. L. Berger, V. J. Della Pietra, and S. A. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, March 1996.

R. Bhar and S. Hamori. *Hidden Markov Models: Applications to Financial Economics*. Advanced Studies in Theoretical and Applied Econometrics. Kluwer, 2004.

J. Bilmes. Dynamic Bayesian multinets. In *Proceedings of the 16th Annual Conference on Uncertainty in Artificial Intelligence (UAI-00)*, pages 38–45, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

J. Bilmes, 2004. Personal communications.

J. A. Bilmes. Burried Markov models for speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '99)*, volume 2, pages 713–716, 1999.

M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999, 1997.

P. J. Brockwell and R. A. Davis. *Introduction to Time Series and Forecasting*. Springer Texts in Statistics. Springer, second edition, 2002.

D. Brown. A note on approximations to discrete probability distributions. *Information and Control*, 2:386–392, 1959.

S. Chib. Calculating posterior distributions and modal estimates in Markov mixture models. *Journal of Econometrics*, 75(1):79–97, November 1996.

S. Chib and E. Greenberg. Analysis of multivariate probit models. *Biometrika*, 85 (2):347–361, June 1998.

D. Chickering, D. Heckerman, and C. Meek. A Bayesian approach to learning Bayesian networks with local structure. In *Proceedings of Thirteenth Conference* *on Uncertainty in Artificial Intelligence*, pages 80–89. Morgan Kaufmann, August 1997.

D. M. Chickering. Learning Bayesian networks is NP-complete. In D. Fisher and H.-J. Lenz, editors, *Learning from data: AI and statistics V*, pages 121–130. Springer-Verlag, New York, 1996.

C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, IT-14(3):462–467, May 1968.

A. S. Cofiño, R. Cano, C. Sordo, and J. M. Gutiérrez. Bayesian networks for probabilistic weather prediction. In *Proceedings of the 15th Eureopean Conference on Artificial Intelligence, ECAI'2002*, pages 695–699, 2002.

T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* The MIT Electrical Engineering and Computer Science Series. MIT Press/McGraw Hill, 1990.

T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley, New York, 1991.

R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems.* Statistics for Engineering and Information Science. Springer-Verlag, New York, 1st edition, 1999.

D. Cox and N. Wermuth. *Multivariate Dependencies – Models, analysis and interpretation*, volume 67 of *Monographs on Statistics and Applies Probability.* Chapman & Hall, 1996.

D. R. Cox and V. Isham. A simple spatial-temporal model of rainfall. *Proceedings of the Royal Society of London Series A – Mathematical, Physical, and Engineering Sciences*, 415(1849):317–328, February 1988.

I. Csiszár. *I*-divergence geometry of probability distributions and minimization problems. *Annals of Probability*, 3(1):146–158, 1975.

J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480, October 1972.

S. Della Pietra, V. Della Pietra, and J. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, April 1997.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via EM algorithm. *Journal of the Royal Statistical Society Series B-Methodological*, 39(1):1–38, 1977.

M. N. Do. Fast approximation of Kullback-Leibler distance for dependence trees and hidden Markov models. *IEEE Signal Processing Letters*, 10(4):115–118, April 2003.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, New York, second edition, 2000.

N. Friedman. Learning belief networks in the presence of missing values and hidden variables. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML-97)*, 1997.

N. Friedman and M. Goldszmidt. Learning Bayesian networks with local structure. In E. Horvitz and F. V. Jensen, editors, *Proceedings of the Twelfth Conference on*

Uncertainty in Artificial Intelligence, pages 252–262. Morgan Kaufmann, August 1996.

D. Geiger and D. Heckerman. Knowledge representation and inference in similarity networks and Bayesian multinets. *Artificial Intelligence*, 82(1-2):45–74, April 1996.

A. Genz. Comparison of methods for the computation of multivariate normal probabilities. *Computing Science and Statistics*, 25:400–405, 1993.

A. Genz. Numerical computation of rectangular bivariate and trivariate normal and t probabilities. *Statistics and Computing*, 14:151–160, 2004.

Z. Ghahramani. An introduction to hidden Markov models and Bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(1):9–42, February 2001.

C. R. Goodall and M. J. Phelan. Edge-preserving smoothing and the assessment of point process models for Gate rainfall fields. In *Statistical Inference in Stochastic Processes*, pages 35–66. Marce Dekker, Inc., 1991.

J. Goodman. Sequential conditional generalized iterative scaling. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 9–16, July 2002.

J. Grim. On structural approximating multivariate discrete probability-distributions. *Kybernetika*, 20(1):1–17, 1984.

P. Guttorp. *Stochastic Modeling of Scientific Data.* Chapman & Hall, 1995.

J. D. Hamilton. *Time Series Analysis.* Princeton University Press, Princeton, New Jersey, 1994.

J. M. Hammersley and P. E. Clifford. Markov fields on finite graphs and lattices. Unpublished manuscript, 1971.

L. E. Hay, G. McCabe, D. M. Wolock, and M. A. Ayers. Simulation of precipitation by weather type analysis. *Water Resources Research*, 27:493–501, 1991.

J. P. Hughes and P. Guttorp. Incorporating spatial dependence and atmospheric data in a model of precipitation. *Journal of Applied Meteorology*, 33(12):1503–1515, December 1994.

J. P. Hughes, P. Guttorp, and S. P. Charles. A non-homogeneous hidden Markov model for precipitation occurrence. *Journal of the Royal Statistical Society Series C Applied Statistics*, 48(1):15–30, 1999.

S. J. Jeffrey, J. O. Carter, K. B. Moodie, and A. R. Beswick. Using spatial interpolation to construct a comprehensive archive of Australian climate data. *Environmental Modelling & Software*, 16(4):309–330, 2001.

F. Jelinek. *Statistical Methods for Speech Recognition*. The MIT Press, 1998.

F. V. Jensen and F. Jensen. Optimal junction trees. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 360–366, 1994.

R. Jiroušek and S. Přeučil. On the effective implementation of the iterative proportional fitting procedure. *Computational Statistics & Data Analysis*, 19:177–189, 1995.

H. Joe. *Multivariate Models and Dependence Concepts*, volume 73 of *Monographs on Statistics and Applied Probability*. Chapman & Hall/CRC, 1997.

M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

R. W. Katz. Probabilistic models. In *Probability, statistics and decision making in the atmospheric sciences*, pages 261–288. Westview Press, Boulder, CO, 1985.

S. Kirshner, S. Parise, and P. Smyth. Unsupervised learning with permuted data. In T. Fawcett and N. Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, pages 345–352. AAAI Press, 2003.

S. Kirshner, P. Smyth, and A. W. Robertson. Conditional Chow-Liu tree structures for modeling discrete-valued vector time series. In M. Chickering and J. Halpern, editors, *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence (UAI-04)*, pages 317–324. AUAI Press, 2004.

U. Kjærulff. Triangulation of graphs – algorithms giving small total state space. Technical report, Aalborg University, 1990.

S. Kotz, N. Balakrishnan, and N. L. Johnson. *Continuous Multivariate Distributions, Volume 1: Models and Applications*. Wiley Series in Probability and Statistics. John Wiley & Sons, second edition, 2000.

A. Krogh, M. Brown, I. S. Mian, K. Sj olander, and D. Haussler. Hidden Markov models in computational biology – applications to protein modeling. *Journal of Molecular Biology*, 235(5):1501–1531, February 1994.

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

J. Kwon and K. P. Murphy. Modeling freeway traffic using coupled HMMs. Technical report, UC Berkeley, 2000.

J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In C. E. Brodley and A. P. Danyluk, editors, *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 01)*, pages 282–289. Morgan Kaufmann, 2001.

S. L. Lauritzen. *Graphical Models*. Oxford Statistical Science Series. Oxford University Press, 1996.

E. Lesaffre and H. Kaufmann. Existence and uniqueness of the maximum-likelihood estimator for a multivariate probit model. *Journal of the American Statistical Association*, 87(419):805–811, September 1992.

I. L. MacDonald and W. Zucchini. *Hidden Markov and Other Models for Discrete-valued Time Series*. Number 70 in Monographs on Statistics and Applied Probability. Chapman & Hall, 1997.

R. Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Conference on Natural Language Learning*, pages 49–55, 2002.

P. McCullagh and J. A. Nelder. *Generalized Linear Models*, volume 37 of *Monographs on Statistics and Applied Probability*. Chapman & Hall/CRC, second edition, 1989.

G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., 2000.

M. Meilă. An accelerated Chow and Liu algorithm: Fitting tree distributions to high-dimensional sparse data. In I. Bratko and S. Dzeroski, editors, *Proceedings of the Sixteenth International Conference on Machine Learning (ICML'99)*, pages 249–57. Morgan Kaufmann, 1999.

M. Meilă and M. I. Jordan. Markov mixture of experts. In R. Murray-Smith and T. A. Johanssen, editors, *Multiple Model Approaches to Nonlinear Modelling and Control*. Taylor and Francis, 1996.

M. Meilă and M. I. Jordan. Learning with mixtures of trees. *Journal of Machine Learning Research*, 1(1):1–48, October 2000.

T. P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, January 2001.

K. P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, Berkeley, CA, July 2002.

R. B. Nelsen. *Introduction to Copulas*, volume 139 of *Lecture Notes in Statistics*. Springer, 1998.

D. Pavlov, A. Popescul, D. M. Pennock, and L. H. Ungar. Mixtures of conditional maximum entropy models. In T. Fawcett and N. Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003)*, pages 584–591. AAAI Press, 2003.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., San Francisco, California, 1988.

D. Pelleg and A. Moore. Using Tarjan's red rule for fast dependency tree construction. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 801–808. MIT Press, Cambridge, MA, 2003.

A. B. Poritz. Linear predictive hidden Markov models and the speech signal. In *Proceedings of the International Conferences on Acoustics, Speech and Signal Processing (ICASSP 82)*, volume 2, pages 1291–1294. IEEE, 1982.

W. H. Press, S. A. Teukolsky, W. V. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edition, 1992.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of IEEE*, 77(2):257–286, February 1989.

B. Rajagopalan and U. Lall. A k-nearest-neighbor simulator for daily precipitation and other weather variables. *Water Resources Research*, 35(10):3089–3101, October 1999.

D. A. Randall, editor. *General Circulation Model Development: Past, Present, and Future*, volume 70 of *International Geophysics Series*. Academic Press, 2000.

C. Raphael. Automatic segmentation of acoustic musical signals using hidden Markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4): 360–370, April 1999.

Š. Raudys. *Statistical and Neural Classifiers: An Integrated Approach to Design*. Advances in Pattern Recognition. Springer, 2001.

A. W. Robertson, S. Kirshner, and P. Smyth. Hidden Markov models for modeling daily rainfall occurrence over Brazil. Technical Report 03-27, School of Information and Computer Science, University of California, Irvine, 2003.

A. W. Robertson, S. Kirshner, and P. Smyth. Downscaling of daily rainfall occurrence over Northeast Brazil using a hidden Markov model. *Journal of Climate*, 17(22): 4407–4424, November 2004.

A. W. Robertson, S. Kirshner, P. Smyth, B. Bates, and S. Charles. Subseasonalto-interdecadal variability of the Australian monsoon over north Queensland, in preparation.

R. Rosenfeld. *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, April 1994.

B. Sansó and L. Guenni. Venezuelan rainfall data analysed by using a Bayesian space-time model. *Journal of the Royal Statistical Society Series C (Applied Statistics)*, 48:345–362, 1999.

S. L. Scott. Bayesian methods for hidden Markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*, 97(457):337–351, March 2002.

M. A. Semenov and J. R. Porter. Climatic variability and the modeling of crop yields. *Agricultural and Forest Meteorology*, 73(3-4):265–283, March 1995.

R. D. Shachter. Bayes-Ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams). In G. F. Cooper and S. Moral, editors, *UAI '98: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 480–487. Morgan Kaufmann, 1998.

R. H. Shumway and D. S. Stoffer. *Time Series Analysis and Its Applications*. Springer Texts in Statistics. Springer, 2000.

P. Smyth, D. Heckerman, and M. I. Jordan. Probabilistic independence networks for hidden Markov probability models. *Neural Computation*, 9(2):227–269, February 1997.

T. Speed and H. Kiiveri. Gaussian Markov distributions over finite graphs. *Annals of Statistics*, 14(1):138–150, March 1986.

N. Srebro. Maximum likelihood bounded tree-width Markov networks. *Artificial Intelligence*, 143(1):123–138, January 2003.

R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13(3):566–579, 1984.

B. Thiesson, C. Meek, D. Chickering, and D. Heckerman. Computationally efficient methods for selecting among mixtures of graphical models. In J. Bernardo, J. Berger, A. Dawid, and A. Smith, editors, *Bayesian Statistics 6*, pages 631–656. Oxford University Press, 1999.

A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

E. Waymire, V. K. Gupta, and I. Rodriguez-Iturbe. A spectral theory of rainfall at the meso-β scale. *Water Resources Research*, 20:1453–1465, 1984.

M. West and J. Harrison. *Bayesian Forecasting and Dynamic Models*. Springer Series in Statistics. Springer, second edition, 1997.

J. Whittaker. *Graphical Models in Applied Multivariate Statistics*. John Wiley & Sons, 1990.

M. J. Wichura. Algorithm AS 241: The percentage points of the normal distribution. *Statistical Algorithms*, 37(3):477–484, 1988.

D. S. Wilks. Multisite generalization of a daily stochastic precipitation generation model. *Journal of Hydrology*, 210:178–191, 1998.

D. S. Wilks. Realizations of daily weather in forecast seasonal climate. *Journal of Hydrometeorology*, 3(2):195–207, April 2002.

D. S. Wilks and R. L. Wilby. The weather generation game: a review of stochastic weather models. *Progress in Physical Geography*, 23(3):329–357, 1999.

L. L. Wilson, D. P. Lettenmaier, and E. Skyllingstad. A hierarchical stochastic model of large-scale atmospheric circulation patterns and multiple-station daily precipitation. *Journal of Geophysical Research*, 97:2791–2809, 1992.

V. Zarutskij. Ispol'zovanie modelej prostoj zavisimosti v zadachah diskriminatsii. *Statisticheskie Problemy Upravleniya*, 38:53–75, 1979. In Russian.

V. Zarutskij. O vydelenii nekotoryh grafov svyazej dlya normal'nyh vektorov v prostranstve bol'shoj razmernosti. In T. Ryabyshkin, editor, *Algoritmicheskoe i programmnoe obespechenie prikladnogo statisticheskogo analiza*, pages 189–208. Nauka, 1980. In Russian.

W. Zucchini and P. Guttorp. A hidden Markov model for space-time precipitation. *Water Resources Research*, 27(8):1917–1923, 1991.

Appendices

A Conjugate Gradient Algorithm for Optimization

The conjugate gradient method iteratively identifies directions in the space of the parameters and maximizes the objective function along each of the directions. The directions are chosen such that each successive direction is orthogonal to the gradient of the previous estimate of the parameters and conjugate to all previous directions, thus, reducing the overlap in the optimization space from iteration to iteration. In order, to apply the conjugate gradient algorithm, we need to be able to find a gradient vector for a set of parameters and to optimize the objective function along an arbitrary vector, i.e., to solve a linear search problem.

A.1 Optimization of Transition Parameters for NHMM

As defined in Section 3.1.2, let $\mathbf{\Omega} = (\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_K)$ be the set of transition parameters. A new set of parameters $\mathbf{\Omega}^{r+1}$ will be obtained from the old $\mathbf{\Omega}^r$ in an iterative manner. Starting with $\mathbf{\Omega}_0 = \mathbf{\Omega}^r$, each successive $\mathbf{\Omega}_{l+1}$ is a set of parameters derived from the previous set of parameters $\mathbf{\Omega}_l$ by optimizing Q_S in a direction $\mathbf{\Phi}_l$, i.e.,

$$\mathbf{\Omega}_{l+1} = \mathbf{\Omega}_l + \nu_l \mathbf{\Phi}_l \text{ where } \nu_l = \arg\max_{\nu} Q_S\left(\mathbf{\Omega}_l + \nu \mathbf{\Phi}_l\right). \tag{A.1}$$

Note that for all $l \geq 0$, $Q_S\left(\mathbf{\Omega}_{l+1}\right) \geq Q_S\left(\mathbf{\Omega}_l\right) \geq Q_S\left(\mathbf{\Omega}_0\right) = Q_S\left(\mathbf{\Omega}^r\right)$; thus by choosing $\mathbf{\Theta}_S^{r+1} = \mathbf{\Theta}_l$ for any positive l would yield an improvement in log-likelihood. Usually, the iterations on l continue until the difference $Q_S\left(\mathbf{\Omega}_l\right) - Q_S\left(\mathbf{\Omega}_{l-1}\right)$ is below a predefined threshold.

We need to specify how to choose directions $\mathbf{\Phi}_l$. The standard gradient descent algorithm uses gradients $\mathbf{\Phi}_l = \nabla\left(Q_S\left(\mathbf{\Omega}_l\right)\right)$; it is, however, usually slow to converge since each of the new directions could have significant overlap with previously chosen direction vectors. The conjugate gradient method reduces the overlap between the optimization directions and can, in turn, speed up convergence to a solution. We used the Polak-Ribiere variation of the conjugate gradient method (e.g., Press et al., 1992, Chapter 10.6):

$$\begin{aligned}
\mathbf{\Phi}_0 &= -\nabla\left(Q_S\left(\mathbf{\Omega}_0\right)\right); \\
\mathbf{\Phi}_{l+1} &= \nabla\left(Q_S\left(\mathbf{\Omega}_l\right)\right) - \gamma_l \mathbf{\Phi}_l; \\
\gamma_l &= \frac{\left(\nabla\left(Q_S\left(\mathbf{\Omega}_{l+1}\right)\right) - \nabla\left(Q_S\left(\mathbf{\Omega}_l\right)\right)\right) \cdot \nabla\left(Q_S\left(\mathbf{\Omega}_{l+1}\right)\right)}{\nabla\left(Q_S\left(\mathbf{\Omega}_l\right)\right) \cdot \nabla\left(Q_S\left(\mathbf{\Omega}_l\right)\right)}.
\end{aligned}$$

All that remains is to perform the line search to find ν_l in Equation A.1. This can be accomplished by any line optimization algorithm. We solve it via Newton-Raphson by finding the zero of the derivative of $Q_S\left(\mathbf{\Omega}_l + \nu_l \mathbf{\Phi}_l\right)$ with respect to ν_l. To do this, we need to compute the first and the second derivatives of $Q_S\left(\mathbf{\Omega}_l + \nu_l \mathbf{\Phi}_l\right)$ with respect to ν_l. Assuming $\mathbf{\Phi}_l = (\lambda_{1\phi}, \ldots, \lambda_{K\phi}, \sigma_{11\phi}, \ldots, \sigma_{KK\phi}, \rho_{1\phi}, \ldots, \rho_{K\phi})$,

$$\begin{aligned}
\frac{\mathrm{d}Q_S\left(\mathbf{\Omega}_l + \nu_l \mathbf{\Phi}_l\right)}{\mathrm{d}\nu_l} &= \sum_{n=1}^{N}\sum_{i=1}^{K} A_{n1}(i) \times \\
&\times \left(\lambda_{i\phi} + \boldsymbol{\rho}'_{i\phi}\mathbf{x}_{n1}\right)\left(1 - P\left(S_{n1} = i|\boldsymbol{x}_{n1}, \mathbf{\Omega}_l + \nu_l\mathbf{\Phi}_l\right)\right) \\
&+ \sum_{n=1}^{N}\sum_{t=2}^{T}\sum_{j=1}^{K}\sum_{i=1}^{K} B_{nt}(i,j)\left(\sigma_{ji\phi} + \boldsymbol{\rho}'_{i\phi}\boldsymbol{x}_{n1}\right) \times \\
&\times \left(1 - P\left(S_{nt} = i|S_{n,t-1} = j, \boldsymbol{x}_{nt}, \mathbf{\Omega}_l + \nu_l\mathbf{\Phi}\right)\right); \\
\frac{\mathrm{d}^2Q_S\left(\mathbf{\Omega}_l + \nu_l \mathbf{\Phi}_l\right)}{\mathrm{d}\nu_l^2} &= -\sum_{n=1}^{N}\sum_{i=1}^{K} A_{n1}(i) \times \\
&\times \left(\lambda_{i\phi} + \boldsymbol{\rho}'_{i\phi}\boldsymbol{x}_{n1}\right)^2 P\left(S_{n1} = i|\boldsymbol{x}_{n1}, \mathbf{\Omega}_l + \nu_l\mathbf{\Phi}_l\right) \times \\
&\times \left(1 - P\left(S_{n1} = i|\boldsymbol{x}_{n1}, \mathbf{\Omega}_l + \nu_l\mathbf{\Phi}_l\right)\right) \\
&- \sum_{n=1}^{N}\sum_{t=2}^{T}\sum_{j=1}^{K}\sum_{i=1}^{K} B_{nt}(i,j)\left(\sigma_{ji\phi} + \boldsymbol{\rho}'_{i\phi}\boldsymbol{x}_{n1}\right)^2 \times \\
&\times P\left(S_{nt} = i|S_{n,t-1} = j, \boldsymbol{x}_{nt}, \mathbf{\Omega}_l + \nu_l\mathbf{\Phi}_l\right) \times \\
&\times \left(1 - P\left(S_{nt} = i|S_{n,t-1} = j, \boldsymbol{x}_{nt}, \mathbf{\Omega}_l + \nu_l\mathbf{\Phi}_l\right)\right).
\end{aligned}$$

This completes the information required to implement conjugate gradient algorithm for the M-step of an EM algorithm for an NHMM model.

A.2 Optimization for Univariate Conditional MaxEnt Models

In this subsection, we describe a conjugate gradient algorithm for finding parameters $\boldsymbol{\delta}$ of a univariate conditional MaxEnt distribution

$$P_{ME}\left(x|\boldsymbol{y}, \boldsymbol{\delta}, \mathcal{F}\right) = \frac{\exp\left(\sum_{f\in\mathcal{F}} \delta_f f\left(x, \boldsymbol{y}\right)\right)}{\sum_X \exp\left(\sum_{f\in\mathcal{F}} \delta_f f\left(x, \boldsymbol{y}\right)\right)}$$

minimizing

$$KL\left(P\left(x|\boldsymbol{y}\right) \parallel P_{ME}\left(x|\boldsymbol{y},\boldsymbol{\delta},\mathcal{F}\right)\right) = -H_P\left[X|\boldsymbol{Y}\right] - l\left(\boldsymbol{\delta}\right).$$

Alternatively we can maximize the log-likelihood

$$\begin{aligned} l\left(\boldsymbol{\delta}\right) &= \sum_X \sum_{\boldsymbol{Y}} P\left(x,\boldsymbol{y}\right) \ln P_{ME}\left(x|\boldsymbol{y},\boldsymbol{\delta},\mathcal{F}\right) \\ &= \sum_{f\in\mathcal{F}} \delta_f \sum_X \sum_{\boldsymbol{Y}} P\left(x,\boldsymbol{y}\right) f\left(x,\boldsymbol{y}\right) - \sum_{\boldsymbol{Y}} P\left(\boldsymbol{y}\right) \ln \sum_X \exp\left(\sum_{f\in\mathcal{F}} \delta_f f\left(x,\boldsymbol{y}\right)\right); \end{aligned}$$

We update $\boldsymbol{\delta}$ iteratively and will denote $\boldsymbol{\delta}$ at iteration l by $\boldsymbol{\delta}_l$. We start with a random initialization $\boldsymbol{\delta}_0$ of the parameters; alternatively, we can start with $\boldsymbol{\delta}_0 = \mathbf{0}$. Just as in Section A.1, we will increase $l\left(\boldsymbol{\delta}\right)$ at each iteration by choosing a direction $\boldsymbol{\phi}_l$ and maximizing $l\left(\boldsymbol{\delta}\right)$ optimizing the parameters in the direction $\boldsymbol{\phi}_l$:

$$\boldsymbol{\delta}_{l+1} = \boldsymbol{\delta}_l + \nu_l \boldsymbol{\phi}_l \text{ where } \nu_l = \arg\max_{\nu} l\left(\boldsymbol{\delta}_l + \nu\boldsymbol{\phi}_l\right).$$

We will denote by $\nabla\left(l\left(\boldsymbol{\delta}\right)\right)$ the gradient of log-likelihood. Individual partial derivatives can be computed as

$$\frac{\partial l\left(\boldsymbol{\delta}\right)}{\partial \delta_f} = \sum_X \sum_{\boldsymbol{Y}} P\left(x,\boldsymbol{y}\right) f\left(x,\boldsymbol{y}\right) - \sum_X \sum_{\boldsymbol{Y}} P\left(\boldsymbol{y}\right) P_{ME}\left(x|\boldsymbol{y},\boldsymbol{\delta},\mathcal{F}\right) f\left(x,\boldsymbol{y}\right).$$

The directions $\boldsymbol{\phi}_l$ can be computed using Polak-Ribiere algorithm by setting

$$\begin{aligned} \boldsymbol{\phi}_0 &= -\nabla\left(l\left(\boldsymbol{\delta}_0\right)\right), \qquad \boldsymbol{\phi}_l = \nabla\left(l\left(\boldsymbol{\delta}_l\right)\right) - \gamma_l \boldsymbol{\phi}_l \text{ where} \\ \gamma_l &= \frac{\left(\nabla\left(l\left(\boldsymbol{\delta}_{l+1}\right)\right) - \nabla\left(l\left(\boldsymbol{\delta}_l\right)\right)\right)\cdot\nabla\left(l\left(\boldsymbol{\delta}_l\right)\right)}{\nabla\left(l\left(\boldsymbol{\delta}_l\right)\right)\cdot\nabla\left(l\left(\boldsymbol{\delta}_l\right)\right)}. \end{aligned}$$

Linear optimization of $\boldsymbol{\delta}$ in the direction $\boldsymbol{\phi}$ can be performed by Newton-Raphson algorithm, iteratively updating ν by setting

$$\nu_{next} = \nu - \left(\frac{\mathrm{d}l\left(\boldsymbol{\delta}+\nu\boldsymbol{\phi}\right)}{\mathrm{d}\nu}\right) \times \left(\frac{\mathrm{d}^2 l\left(\boldsymbol{\delta}+\nu\boldsymbol{\phi}\right)}{\mathrm{d}\nu^2}\right)^{-1}$$

until $\frac{\mathrm{d}l\left(\boldsymbol{\delta}+\nu\boldsymbol{\phi}\right)}{\mathrm{d}\nu}$ is sufficiently close to 0. It requires the computation of the first and the second derivative of $l\left(\boldsymbol{\delta}+\nu\boldsymbol{\phi}\right)$:

$$\begin{aligned}
l\left(\boldsymbol{\delta}+\nu\boldsymbol{\phi}\right) &= \sum_{f\in\mathcal{F}}\left(\delta_f+\nu\phi_f\right)\sum_{X}\sum_{\boldsymbol{Y}} P\left(x,\boldsymbol{y}\right) f\left(x,\boldsymbol{y}\right) \\
&\quad -\sum_{\boldsymbol{Y}} P\left(\boldsymbol{y}\right)\ln\sum_{X}\exp\left(\sum_{f\in\mathcal{F}}\left(\delta_f+\nu\phi_f\right) f\left(x,\boldsymbol{y}\right)\right) \\
\frac{\mathrm{d}l\left(\boldsymbol{\delta}+\nu\boldsymbol{\phi}\right)}{\mathrm{d}\nu} &= \sum_{f\in\mathcal{F}}\phi_f\sum_{X}\sum_{\boldsymbol{Y}} P\left(x,\boldsymbol{y}\right) f\left(x,\boldsymbol{y}\right) \\
&\quad -\sum_{f\in\mathcal{F}}\phi_f\sum_{X}\sum_{\boldsymbol{Y}} P\left(\boldsymbol{y}\right) P_{ME}\left(x|\boldsymbol{y},\boldsymbol{\delta}+\nu\boldsymbol{\phi},\mathcal{F}\right) f\left(x,\boldsymbol{y}\right); \\
\frac{\mathrm{d}^2 l\left(\boldsymbol{\delta}+\nu\boldsymbol{\phi}\right)}{\mathrm{d}\nu^2} &= -\sum_{X}\sum_{\boldsymbol{Y}} P\left(\boldsymbol{y}\right) P_{ME}\left(x|\boldsymbol{y},\boldsymbol{\delta}+\nu\boldsymbol{\phi},\mathcal{F}\right)\left(\sum_{f\in\mathcal{F}}\phi_f f\left(x,\boldsymbol{y}\right)\right)^2 \\
&\quad -\sum_{\boldsymbol{Y}} P\left(\boldsymbol{y}\right)\left(\sum_{X} P_{ME}\left(x|\boldsymbol{y},\boldsymbol{\delta}+\nu\boldsymbol{\phi},\mathcal{F}\right)\sum_{f\in\mathcal{F}}\phi_f f\left(x,\boldsymbol{y}\right)\right)^2.
\end{aligned}$$

B Proofs of Theorems

Proof of Theorem 4.1:

$$
\begin{aligned}
KL(P \parallel T) &= \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln P(\boldsymbol{x}|\boldsymbol{y}) - \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln T(\boldsymbol{x}|\boldsymbol{y}) \\
&= \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln P(\boldsymbol{x}|\boldsymbol{y}) - \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln P_B(\boldsymbol{x}|\boldsymbol{y}) \\
&\quad + \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln P_B(\boldsymbol{x}|\boldsymbol{y}) - \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln T(\boldsymbol{x}|\boldsymbol{y}) \\
&= KL(P \parallel P_B) + \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln \prod_{v \in \mathcal{V}} P_B\left(x_v | \boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) \\
&\quad - \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln \prod_{v \in \mathcal{V}} T\left(x_v | \boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) \\
&= KL(P \parallel P_B) + \sum_{v \in \mathcal{V}} \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln P\left(x_v | \boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) \\
&\quad - \sum_{v \in \mathcal{V}} \sum_{\boldsymbol{Y}} \sum_{\boldsymbol{X}} P(\boldsymbol{x}, \boldsymbol{y}) \ln T\left(x_v | \boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) \\
&= KL(P \parallel P_B) \\
&+ \sum_{v \in \mathcal{V}} KL\left(P\left(x_v | \boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right) \parallel T\left(x_v | \boldsymbol{x}_{pa_x(v)}, \boldsymbol{y}_{pa_y(v)}\right)\right).
\end{aligned}
$$

$\square$

Proof of Theorem 4.2: First, let *depth* (v) for $v \in \mathcal{V}$ be the number of ancestors of v that are members of $\mathcal{V}$. *depth* (v) can best be defined recursively:

$$
depth(v) = \begin{cases} 0 & pa_x(v) = \emptyset, \\ depth(u) & pa_x(v) \neq \emptyset \wedge \exists! u \in pa_x(v). \end{cases}
$$

We will prove the statement of the theorem by induction on depth of $v \in \mathcal{V}$. For the case of *depth* $(v) = 0$ for $v \in \mathcal{V}$ either $pa_y(v) = \emptyset$ or $|pa_y(v)| = 1$. For the case of

$pa_y(v) = \emptyset$,

$$T(x_v) = T\left(x_v | y_{pa_y(v)}\right) = P\left(x_v | y_{pa_y(v)}\right) = P(x_v).$$

If $|pa_y(v)| = 1$, let $u \in \mathcal{Y}$ be that one element of $pa_y(v)$. Since T does not define probabilities on $\mathcal{Y}$, we assume that $\boldsymbol{Y}$ is sampled from $\mathcal{Y}$ according to P.

$$T(x_v) = \sum_{Y_u} P(y_u)\, T(x_v|y_u) = \sum_{Y_u} P(y_u)\, P(x_v|y_u) = P(x_v).$$

Assuming the statement of the theorem holds for all $v \in \mathcal{V}$ with $depth(v) \leq d$, $d \geq 0$, we will show that it holds for $v \in \mathcal{V}$ with $depth(v) = d+1$. Let v be such a vertex in $\mathcal{V}$. $pa_x(v)$ contains only one element; let $u \in \mathcal{V}$ be that element. By definition of $degree(v)$, $degree(u) = d$. Then

$$T(x_v) = \sum_{X_u} T(x_u)\, T(x_v|x_u) = \sum_{X_u} P(x_u)\, P(x_v|x_u) = P(x_u).$$

$\square$

Proof of Theorem 4.4: We will prove the statement of the theorem by induction on ordering index $i = 1, \ldots, M$. For $i = 1$, $\forall f \in \mathcal{F}_1$ $D_x(f) = \{v_1\}$. Then for all $f \in \mathcal{F}_1$

$$\begin{aligned}
&\sum_{\boldsymbol{X}} \sum_{\boldsymbol{Y}} P(\boldsymbol{y})\, P_{PUC-ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\delta}^\star, \mathcal{F}_{PUC})\, f(\boldsymbol{x}, \boldsymbol{y}) \\
&= \sum_{X_{v_1}} \sum_{\boldsymbol{Y}} P(\boldsymbol{y})\, P_{PUC-ME}(x_{v_1}|\boldsymbol{y}, \boldsymbol{\delta}_1^\star, \mathcal{F}_1)\, f(x_{v_1}, \boldsymbol{y}) = \sum_{X_{v_1}} \sum_{\boldsymbol{Y}} P(x_{v_1}, \boldsymbol{y})\, f(x_{v_1}, \boldsymbol{y}) \\
&= \sum_{\boldsymbol{X}} \sum_{\boldsymbol{Y}} P(\boldsymbol{x}, \boldsymbol{y})\, f(\boldsymbol{x}, \boldsymbol{y})
\end{aligned}$$

as $\boldsymbol{\delta}_1^\star$ is obtained by satisfying all of such constraints for features in $\mathcal{F}_1$. For the inductive step, assume that the statement of the theorem is valid for all $k = 1, \ldots, i-1$. Then $\forall f \in \mathcal{F}_i$,

$$
\begin{aligned}
&\sum_{\boldsymbol{X}} \sum_{\boldsymbol{Y}} P(\boldsymbol{y})\, P_{PUC-ME}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\delta}^\star, \mathcal{F}_{PUC})\, f(\boldsymbol{x}, \boldsymbol{y}) \\
&= \sum_{\boldsymbol{X}_{fa_x(v_i)}} \sum_{\boldsymbol{Y}} P(\boldsymbol{y})\, P_{PUC-ME}\left(\boldsymbol{x}_{fa_x(v_i)}|\boldsymbol{y}, \boldsymbol{\delta}^\star, \mathcal{F}_1, \ldots, \mathcal{F}_i\right) f\left(\boldsymbol{x}_{D_x(v_i)}, \boldsymbol{y}\right) \\
&= \sum_{\boldsymbol{X}_{fa_x(v_i)}} \sum_{\boldsymbol{Y}} P(\boldsymbol{y})\, P_{PUC-ME}\left(\boldsymbol{x}_{pa_x(v_i)}|\boldsymbol{y}, \boldsymbol{\delta}^\star, \mathcal{F}_1, \ldots, \mathcal{F}_{i-1}\right) \\
&\quad \times P_{PUC-ME}\left(x_{v_i}|\boldsymbol{x}_{pa_x(v_i)}, \boldsymbol{y}, \boldsymbol{\delta}_i^\star, \mathcal{F}_i\right) f\left(\boldsymbol{x}_{D_x(v_i)}, \boldsymbol{y}\right) \\
&\overset{(a)}{=} \sum_{\boldsymbol{X}_{fa_x(v_i)}} \sum_{\boldsymbol{Y}} P\left(\boldsymbol{x}_{pa_x(v_i)}, \boldsymbol{y}\right) P_{PUC-ME}\left(x_{v_i}|\boldsymbol{x}_{pa_x(v_i)}, \boldsymbol{y}, \boldsymbol{\delta}_i^\star, \mathcal{F}_i\right) f\left(\boldsymbol{x}_{D_x(v_i)}, \boldsymbol{y}\right) \\
&\overset{(b)}{=} \sum_{\boldsymbol{X}_{fa_x(v_i)}} \sum_{\boldsymbol{Y}} P\left(\boldsymbol{x}_{fa_x(v_i)}, \boldsymbol{y}\right) f\left(\boldsymbol{x}_{D_x(v_i)}, \boldsymbol{y}\right) = \sum_{\boldsymbol{X}} \sum_{\boldsymbol{Y}} P(\boldsymbol{x}, \boldsymbol{y})\, f(\boldsymbol{x}, \boldsymbol{y})
\end{aligned}
$$

where (a) is due to the assumption in the theorem, and (b) is due to the definition of how $\boldsymbol{\delta}_i$ is obtained. □

C Experimental Setup

In this appendix, we list the details of experimental runs of the software for Chapters 7 and 8.

The EM (Baum-Welch) algorithm was run with random restarts until the convergence in scaled log-likelihood (log-likelihood divided by the number of binary observations in the data). The parameters for the restart with the highest log-likelihood are chosen as the solution. For experiments in Chapter 7, 10 random restarts were used for each run of EM. For Chapter 8, 50 random restarts were used for each

EM run. For all of the runs of the software, the convergence threshold was set to 5×10^{-5}, i.e., the iterations of EM stopped when the difference between the scaled log-likelihoods of the successive iterations was less than 5×10^{-5}.

For the initialization of an HMM, all of the parameters of the first state $\boldsymbol{\pi}$ and the transition matrix $\boldsymbol{\Gamma}$ are drawn from the uniform distribution $U(0,1)$ and then renormalized such that all parameters in $\boldsymbol{\pi}$ and all rows of $\boldsymbol{\Gamma}$ add up to one. For HMM-CI, HMM-Chains, HMM-CL, and HMM-CCL, all of free emission parameters are initialized by being chosen independently and at random from $U(0,1)$ with CL and CCL being initialized as conditionally independent (i.e., no edges). For HMM-MaxEnt, the parameters for full bivariate MaxEnt models are initialized by sampling from $\mathcal{N}(\cdot|0,1)$. HMM-PUC-MaxEnt and HMM-PUC-MaxEnt are initialized without any bivariate features for the emission probability distributions. The parameters for univariate features are initialized by sampling from $\mathcal{N}(\cdot|0,1)$.

In algorithm `StructureLearningPuc-MaxEnt` (Figure 4.10, the threshold for adding the features was set to 0.01. The maximum change in Newton-Raphson algorithm was set to 0.0001.

Chapter 8 contains estimates of correlation and persistence estimates for a large number of models. These estimates were obtained by simulating data from these models, 500 data sequences per each sequence of the training data used to learn the parameters of a model. The length of each sequence was the same as in the training data set.