Bayesian Centroid Estimation

by

Luis E. Carvalho

Sc. M., Brown University, 2004

Sc. M., Federal University of Ceará, Brazil, 2002

Sc. M., Federal University of Rio de Janeiro, Brazil, 2000

B. S., Federal University of Ceará, Brazil, 1998

Submitted in partial fulfillment of the requirements
for the Degree of Doctor of Philosophy in the
Program in Applied Mathematics at Brown University

Providence, Rhode Island

May, 2009

This dissertation by Luis E. Carvalho is accepted in its present form by the Department of Applied Mathematics as satisfying the dissertation requirement for the degree of Doctor of Philosophy.

Date _____ _____

Charles Lawrence, Director

Recommended to the Graduate Council

Date _____ _____

Stuart Geman, Reader

Date _____ _____

Ben Raphael, Reader

Date _____ _____

Alexander Brodsky, Reader

Approved by the Graduate Council

Date _____ _____

Sheila Bonde, Dean of the Graduate School

Vitæ

Luis Eduardo Ximenes Carvalho was born on September, 3, 1975 in Fortaleza, Brazil. He graduated *magna cum laude* from Federal University of Ceará in January 1998 in Civil Engineering. He then started working with logistics while attending a M.Sc. program in Transportation Engineering from Federal University of Rio de Janeiro, from which he graduated on October 2000. He also graduated on August 2002 from Federal University of Ceará receiving a M.Sc. in Computer Science. He then entered the graduate program at Brown University in September 2002, defending this Ph.D. thesis on July 31, 2008.

Ad astra per aspera

For

Dátila

Acknowledgements

First and foremost, I would like to thank my adviser, Prof. Chip Lawrence, for his continuous support and inspiring motivation. He was always motivating me to push the envelope and find that next result! From our first meeting, when he initially proposed me the idea of the centroid estimator, to day of the defense he enthusiastically offered me excellent guidance while providing me all the freedom I needed.

I would also like to thank many professors at Brown with whom I have worked and from whom I have learned mathematical, biological and non-scientific subjects! In particular, I would like to thank Prof. Basilis Gidas for his initial support and discussions. I am really grateful to Prof. Alex Brodsky for my "internship" at his lab where I did most of my biological learning and bioinformatics training. It was a much rewarding experience, specially because of the fruitful discussions, developments, and friendships; many thanks to my friends at the Brodsky lab, specially to Prof. Nicola Neretti, Richard Park, and Sara Hillenmeyer.

Most importantly, I thank my family. I have no words to thank Dátila, my wife, for her unconditional support, strength, inspiration, and love. Nothing would be possible without her, her caring encouragement and companionship through both the difficulties and joys. My deepest appreciation goes to my parents—including my parents-in-law—for their support (both financial and through prayers) and encouragement. I am mostly indebted to you all.

It is impossible to remember everyone that had a role in my research, but I would like to thank all students and friends that contributed to my graduate education through discussions, help, and/or beer. In particular I would like to thank the

Lawrence group for helpful criticism and invaluable comments during my research and the Brownzilians for making my stay in Providence much more enjoyable.

Contents

Bibliography 84

List of Tables

List of Figures

List of Algorithms

Abstract of "Bayesian Centroid Estimation" by Luis E. Carvalho, Ph.D., Brown University, May, 2009.

Maximum likelihood estimators have traditionally dominated discrete inference for a long time. In this work we apply statistical decision theory to derive a new contender that minimizes posterior Hamming loss: the centroid estimator. We show that the centroid estimator is also the minimum Bayes risk estimator for a family of loss functions, including the important case when the data being compared is not only categorical, but also ordinal and interval. The centroid estimator is formally characterized as a solution to a discrete optimization problem having posterior marginal distributions as inputs. We discuss both specific constraints of interest and broad conditions under which this optimization problem becomes tractable, present efficient algorithms for its solution, and offer further generalizations to centroid estimation. We apply centroid estimation to many applications of general, well-known models—partition set pertinence models, hidden Markov models, change point models, and graphical models—and to classical problems in computational biology—sequence alignment, RNA secondary structure prediction, and reconstruction of ancestral states.

CHAPTER 1

Introduction

Consider the probability distribution on the space of three binary variables. Figure 1.1 shows one possible distribution for the eight points in this space: the black point $L(1,1,1)$ has probability p_1, gray points have probability p_2 as labeled including $C(0,0,0)$, and white, unshaded, points have zero probability. The edges serve simply as spatial guidance as they connect points that differ only in one component.

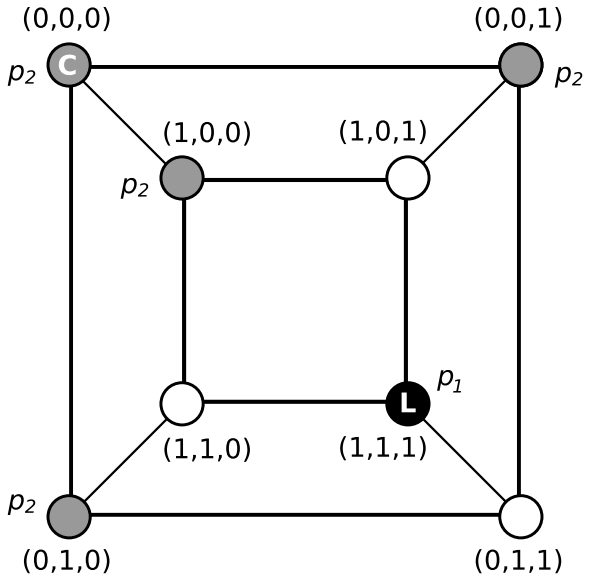

FIGURE 1.1. Simple probability space.

We assume that $p_1 > p_2$, and so L is the point with the highest probability. If we were to elect one of the points to be a "representative" of the ensemble, point L certainly has appeal as it incorporates the most plausible choice. However, if we set p_1 to be not much larger than p_2, say $p_1 < 2p_2$, then the group of gray points has total probability $4p_2$ and so the point C, although not having high probability itself, nevertheless is close to a region of the space with high total probability. We will

revisit this simple example in the next chapter, but for now it suffices to note that C *centers* itself in a region of high probability and thus is also appealing in this sense.

This effect of points with individual high probability, like L in our example, being far from regions of the space with high total probability, like the region comprised of points connected to C, becomes more dramatic as the size of the space grows. In the next chapter we argue that this effect can be better understood if we cast this choice of a representative point as a statistical inference problem—where the representative point becomes a point inference, an *estimator*—and then show that point C arises from a more principled and discriminative criterion than the one behind point L.

In this work we discuss and characterize a class of estimators in discrete spaces we call *centroid* estimators. Centroid estimators are obtained from a Bayesian statistical approach by optimizing an averaged criterion over the probability distribution on the ensemble that takes differences between point components in consideration.

In the next chapter we formally present the centroid estimator as a discrete optimization problem defined on marginal probabilities for each point component. We then discuss unconstrained and constrained versions of this problem to later introduce more generalized versions of the centroid that arise from more structured spaces. Chapter 3 discusses some applications of the centroid to general models, while Chapter 4 focuses on applications to computational biology. The common theme pervading the text is dynamic programming; the principle of optimality described by Bellman [**Bel57**] and further generalized by Bertele and Brioschi [**BB72**] is used thoroughly here, albeit in a different, more graph theoretical form. Before discussing centroid estimation, however, we need a few basic concepts from graph theory [**Die05, Gol04**].

1. Concepts

A *graph* G is a pair (V, E) where V is the *vertex* set and $E \subset V^2$ is the *edge* set. The vertex and edge set of a graph G are denoted by $V(G)$ and $E(G)$ respectively. The cardinality of a set S is denoted by $|S|$; the number of vertices in G is then $|V(G)|$, for example. If $e = (u, v) \in E$ we then say that u is *adjacent* to v. We define

the *neighborhood* of vertex v as the set of vertices that are adjacent to v, denoted by $\textsc{Adj}_G(v) = \{u \in V(G) : (u, v) \in E(G)\}$. The *degree* of $v \in V(G)$ is defined as $|\textsc{Adj}_G(v)|$. If any two vertices in G are adjacent to each other, $E(G) = V(G)^2$, then G is said to be a *complete* graph. Pairwise non-adjacent vertices are called *independent*. A set $S \subset V(G)$ where no two vertices are adjacent, i.e. $(u, v) \notin E(G)$ for $u, v \in S$, is called an independent or *stable* set.

A graph G' is a *subgraph* of graph G if $V(G') \subset V(G)$ and $E(G') \subset E(G)$. Given a subset of vertices $W \subset V(G)$ of G, G' is said to be a subgraph *induced* by W if $V(G') = W$ and $E(G') = \{(u, v) \in E(G) : u, v \in W\}$. A *clique* K of a graph G is a maximal complete subgraph of G: the subgraph of G induced by $K \cup \{v\}$ for any other vertex $v \notin K$ is not complete.

A *walk* W of length $n > 0$ in a graph G is a sequence $(v_1, v_2, \cdots, v_n)$ such that $(v_i, v_{i+1}) \in E(G)$. If the v_i are all distinct then W is a *path* with ends v_1 and v_n and we say that v_1 and v_n are linked by W. A path graph P is then a non-empty graph of the form $V(P) = \{v_1, \cdots, v_n\}$ and $E(P) = \{(v_i, v_{i+1}) : i = 1, \ldots, n-1\}$ where the v_i are all distinct. If $P = (v_1, \ldots, v_n)$, $n \geq 3$, is a path, then $C = (v_1, \ldots, v_n, v_1)$ is called a *cycle*. A non-empty graph G is called *connected* there exists (at least) a path linking any two of its vertices.

An acyclic graph, one that contains no cycles, is called a *forest*. A connected forest is called a *tree*. The vertices of a tree are also called *nodes*; if a node has unitary degree then it is called a *leaf*, otherwise it is an *internal* node. There is a unique path between any two nodes u and v in a tree T, denoted by uTv. We can distinguish a node r in a tree T as the *root* in order to define a partial order in T: r is then the minimum, and a node u precedes node v, denoted by $u \preceq v$, if $u \in vTr$. T is then said to be *rooted* at r. A subgraph of tree is called a *subtree*, and we denote by T_v the subtree rooted at v and defined as $T_v = \{u \in T : v \in uTr\}$. We also denote $\textsc{Child}_T(v) = \{u \in T : (u, v) \in E(T), v \preceq u\}$ and $\textsc{Parent}_T(v) = u$ if $v \in \textsc{Child}_T(u)$.

The *post-order* σ of the nodes in a tree T from node r can be defined recursively from routine `PostOrder` in Algorithm 1.1. Note that if u precedes v then $\sigma(u) \geq \sigma(v)$,

and that $\sigma(r) = |T|$. We can use the post-order σ of the nodes in a rooted tree T at r to define the *height* of a node v in T,

$$\textsc{Height}_T(v) = \begin{cases} 0, & \text{if } v \text{ is a leaf} \\ 1 + \max\limits_{u \in \textsc{Adj}_T(v), \sigma(u) < \sigma(v)} \textsc{Height}_T(u), & \text{otherwise,} \end{cases}$$

the *depth* of v,

$$\textsc{Depth}_T(v) = \begin{cases} 0, & \text{if } v = r \\ 1 + \max\limits_{u \in \textsc{Adj}_T(v), \sigma(u) > \sigma(v)} \textsc{Depth}_T(u), & \text{otherwise,} \end{cases}$$

and the depth of an edge $e = (u, v)$: $\textsc{Depth}_T(e) = \max_{u,v}\{\textsc{Depth}_T(u), \textsc{Depth}_T(v)\}$.

Input: Tree T rooted at r.
Output: Post-order σ.
function `PostOrder`(T, r);
begin
  $i \leftarrow 0$; $\sigma \leftarrow \{\}$;
  `POaux`(T, r);
  return σ;
end

function `POaux`(T, v);
begin
  for $u \in \textsc{Child}_T(v)$ **do**
    `POaux`(T, u);
  end
  $i \leftarrow i + 1$;
  $\sigma(v) \leftarrow i$;
end

Algorithm 1.1: Post-order of a rooted tree.

Let G be a graph, T a tree, and let $\mathcal{V} = \{V_t\}_{t \in T}$, $V_t \subset V(G)$ for all $t \in T$, be the family of *parts* of the pair $(T, \mathcal{V})$. A *tree-decomposition* of G is a pair $(T, \mathcal{V})$ that satisfies the following conditions:

(1) $V(G) = \cup_{t \in T} V_t$
(2) If $(u, v) \in E(G)$ then there exists $t \in T$ such that $u, v \in V_t$
(3) For $r, t, s \in T$, if t is in the path connecting r and s in T then $V_r \cap V_s \subseteq V_t$

The *width* of a tree-decomposition $(T, \mathcal{V})$ of G is defined as $\max_{t \in T} |V_t| - 1$, and the *tree-width* $\mathrm{TW}(G)$ of G is the minimum width over all tree-decompositions of G.

Figure 1.2 illustrates three tree-decompositions for a graph G with widths (a) 10, (b) 5, and (c) 2; the first tree-decomposition is trivial since $V_{i_1} = V(G)$, while the last tree-decomposition is minimum, that is, it realizes the tree-width of G.

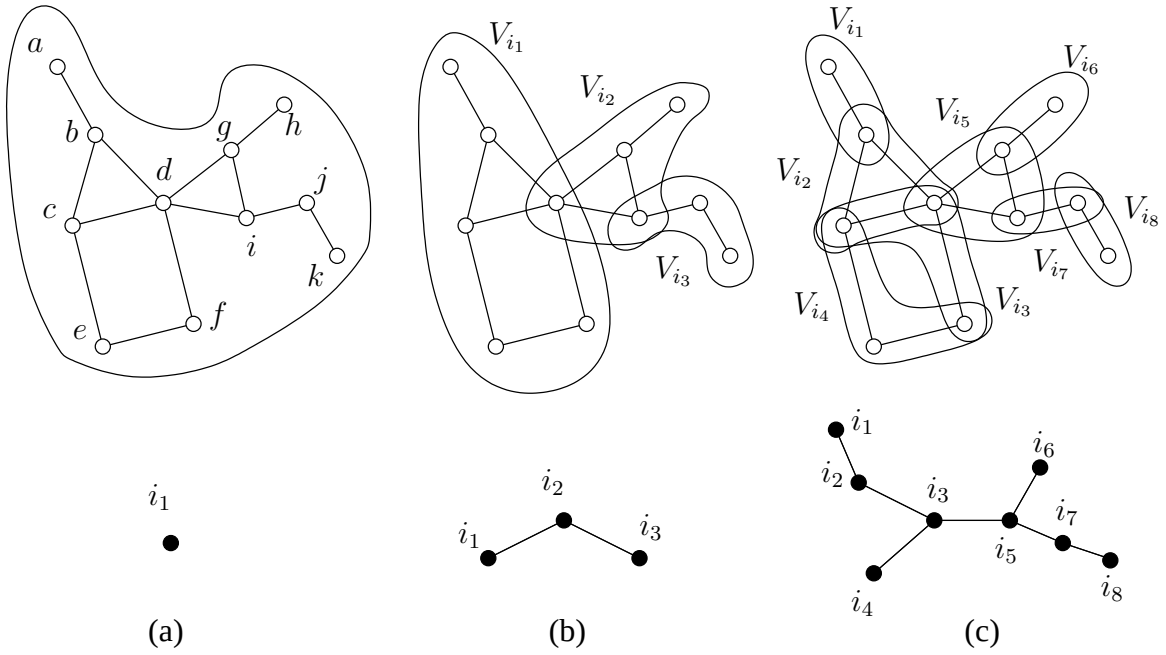

FIGURE 1.2. Examples of tree-decomposition.

A k-tree can be defined recursively as follows: a k-tree with $k+1$ vertices is complete graph; a k-tree with $n+1$ vertices, $n > k+1$, can be built from k-tree T with n vertices by adding a vertex v to T and making v adjacent to the vertices in a clique of size k in T. It can be shown that a k-tree has tree-width k, since a k-tree is a triangulated graph with maximum clique size $k+1$.

A *directed graph* (or *digraph*) D is a pair (V, A), $A \subset V^2$, together with two maps init: $A \to V$ and ter: $A \to V$ assigning to each *arc* $a \in A$ an initial vertex $\text{init}(a)$ and a terminal vextex $\text{ter}(a)$. A digraph $D = (V, A)$ is an *orientation* of an undirected graph G if $V = V(G)$, $A = E(G)$, and for every $e \in E(G)$, $(\text{init}(e), \text{ter}(e)) \in A$. A *transitive orientation* D of a graph G is an orientation where $a, b \in E(D)$ with $\text{ter}(a) = \text{init}(b)$ implies that there exists $c \in E(D)$ such that $\text{init}(c) = \text{init}(a)$ and $\text{ter}(c) = \text{ter}(b)$. A graph is said to be a *comparability* graph if it admits a transitive orientation.

CHAPTER 2

Centroid Estimation

In this chapter we introduce the centroid estimator as a Bayes risk minimizer for discrete spaces. We adopt a general approach and regard the estimation procedure as a purely optimization technique, given the sufficient statistics. As a result, the nature of the text is very algorithmic as the main results are characterizations of posterior risk minimizers and how to efficiently obtain them. We abstract from specifying probability distributions here, and delay a more practical treatment to the next chapters when we focus on applications.

1. Bayesian discrete estimators

The main interest of this work is Bayesian inference on a discrete space, that is, our desired estimator is a vector of size n and lives in a *parameter* space $\Theta \subset A^n$ where $A = \{a_j\}_{j=1}^m$ is a *discrete* set. As it is usual in Bayesian statistics, we equip our multidimensional parameter space Θ with a *prior* probability distribution $P(\theta|\eta)$, $\theta \in \Theta$, where η is a hyperparameter. The prior probability $P(\theta|\eta)$ represents our belief in θ, conditional on η, *before* observing any data. The observed data X belongs to a sample space $\mathcal{X}$ with conditional probability distribution $P(X|\theta, \eta)$, the *likelihood* of θ. After observing X we change our belief about θ as measured by the *posterior* probability and given by Bayes's rule:

$$P(\theta|X, \eta) = \frac{P(X|\theta, \eta)P(\theta|\eta)}{\sum_{\theta\in\Theta} P(X|\theta, \eta)P(\theta|\eta)}$$

The term in the denominator of Bayes's rule, $P(X|\eta) = \sum_{\theta\in\Theta} P(X|\theta, \eta)P(\theta|\eta)$, is the *marginal* probability of the data. Now let us assume, for simplicity, that η is known and omit it from the notation.

Our goal is to find a representative $\hat{\theta}$ from Θ that best captures — in a sense that we will make precise shortly — the posterior $P(\theta|X)$. This representative, which is clearly a function of the data X, is known as an *estimator*.

A principled way to obtain an estimator is to cast the estimation procedure as a decision problem. Let $\mathcal{A}$ be a set of actions and $L : \Theta \times \mathcal{A} \to \mathbb{R}^+$ be a loss function. An estimator $\hat{\theta} : \mathcal{X} \to \Theta$ is then a decision based on X (when $\mathcal{A} \equiv \Theta$) [**LC03, CL00**]. We now restrict our interest to estimators that minimize the *posterior risk* with respect to L:

$$\rho_L(\hat{\theta}) = E_{\theta|X} L(\theta, \hat{\theta}) = \sum_{\theta \in \Theta} L(\theta, \hat{\theta}) P(\theta|X).$$

For each loss function L we then have an estimator $\hat{\theta}_L = \arg\min_{\tilde{\theta} \in \Theta} \rho_L(\tilde{\theta})$, where we adopt the subscript notation to emphasize that the estimator depends on the loss function. Thus, according to this principled characterization, the performance of an estimator depends largely on the choice of its loss function. If, for example, the estimator is evaluated by some metric M as criterion, then using M as loss would certainly result in improved performance [**Goo96**]. Ideally, the loss function L should also yield a tractable optimization problem to define $\hat{\theta}_L$.

Let us denote by $I(\cdot)$ the indicator function. A common choice for L is the zero-one loss,

$$Z(\theta, \tilde{\theta}) = \prod_{i=1}^{n} I(\theta_i \neq \tilde{\theta}_i) = I(\theta \neq \tilde{\theta}).$$

By minimizing the posterior risk with respect to Z we obtain the ubiquitous maximum *a posteriori* (MAP) estimator $\hat{\theta}_M$ [**Bes86**]:

$$\hat{\theta}_M = \arg\min_{\tilde{\theta} \in \Theta} E_{\theta|X} Z(\theta, \tilde{\theta}) = \arg\min_{\tilde{\theta} \in \Theta} \left\{ 1 - E_{\theta|X} I(\theta = \tilde{\theta}) \right\} = \arg\max_{\tilde{\theta} \in \Theta} P(\tilde{\theta}|X).$$

Given the discrete nature of Θ, a more general and natural choice for a loss function would be to compare points by each of their components, that is, using a Hamming loss. The Hamming loss H between points $\theta, \tilde{\theta} \in \Theta$ is defined as

$$H(\theta, \tilde{\theta}) = \sum_{i=1}^{n} I(\theta_i \neq \tilde{\theta}_i).$$

Loss function H is clearly more discriminative than Z as it compares point components instead of points directly.

Since Hamming loss considers differences in point components, it can be further generalized to

$$C(\theta, \tilde{\theta}) = \sum_{i=1}^{n} c(\theta_i, \tilde{\theta}_i)$$

where c is some loss function on A^2 with $0 < c(a_1, a_2) < \infty$ and $c(a_1, a_2) = c(a_2, a_1)$ for all $a_1, a_2 \in A$. Clearly, when $c(a_1, a_2) = I(a_1 \neq a_2)$ we have the usual Hamming loss.

We are now in position to present our main estimator.

DEFINITION 1 (Centroid estimator). The (generalized) *centroid* estimator is the posterior risk minimizer for C,

$$\hat{\theta}_C = \arg\min_{\tilde{\theta} \in \Theta} E_{\theta|X} C(\theta, \tilde{\theta}),$$

and the (regular) centroid estimator is the posterior risk minimizer for H.

1.1. Comparison to MAP. To help gain insight on the centroid estimator, let us compare it to the MAP estimator in a simple example. Consider initially the simple case of three correlated binary variables as in the introduction. Suppose that, similarly to the example in the introduction, after observing data the probabilities of the eight combinations of these variables are those shown in Figure 2.1 with $p_2 < p_1 < 2p_2$.

As Figure 2.1 shows, while the point $L(1, 1, 1)$ is the most likely, it does not seem to represent the data well, since the only other points with positive probabilities are the four points that are about as different from this point as possible and together they are between 2 to 4 times more probable than point L. Point M is the mean with coordinates $(p_1 + p_2, p_1 + p_2, p_1 + p_2)$ and, since $p_1 + p_2 < 1/2$, it is always closer to $C(0, 0, 0)$ in any L_p metric, $p > 1$.

For the general case of n binary variables, where $p_2 < p_1 < (n-1)p_2$, we could still have a cluster of $n+1$ points that differ from some point C by no more than one component and such that each has probability p_2. The point at the opposite end of

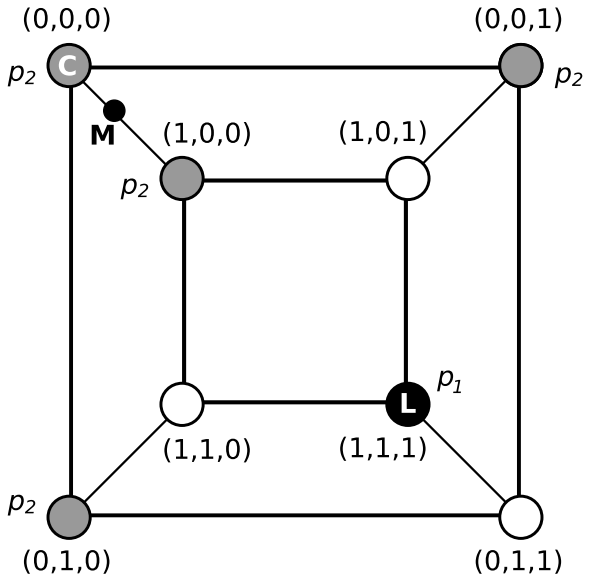

FIGURE 2.1. Simple probability space and two different estimators: black point $L(1,1,1)$ has probability p_1 and is the MAP estimate; gray points have probability p_2 as labeled, with $p_2 < p_1 < 2p_2$, including $C(0,0,0)$, and white, unshaded, points have zero probability.

the hypercube L has probability p_1, all its neighbors have zero probability, and differs from all the rest with positive probability in at least $n-1$ of its n components. Since $p_1+(n+1)p_2 = 1$, which implies $1/(2n) < p_2 < 1/(n+2)$, if we choose $p_2 = 1/(n+3)$ then we would have $p_1 = 2/(n+3)$. The ratio of the posterior probability mass for the cluster around point C to that around L is $(n+1)p_2/p_1 = (n+1)/2$. Thus, as the dimensionality of the problem increases, the proportion of posterior mass around point C becomes arbitrarily large when compared to L, and so does the distance between C and L. Nevertheless, since $p_2 < p_1$, point L is the most likely after observing the data, the MAP estimator.

As the hypercube example suggests, the centroid estimator is more robust, closer to regions of the posterior probability space that concentrate probability mass, than the MAP estimator. Since Hamming loss more finely differentiates points, we can draw an analogy between the centroid estimator and the Bayesian mean or median on continuous spaces. In fact, we can adopt a parametrization that builds a correspondence between our discrete Θ and a continuous space.

Consider a matrix $\vec{\xi} = (\xi_{ij})$ where ξ_{ij} indicates if position j, $j = 1, \ldots, n$, holds symbol $a_i \in A$. We now vectorize $\vec{\xi}$

$$\xi = \mathtt{vec}(\vec{\xi}) = (\xi_{11}, \ldots, \xi_{m1}, \ldots, \xi_{1n}, \ldots, \xi_{mn})^T$$

and our estimators. Note that the space is now $\Xi = \{\xi \in \{0,1\}^{mn} : \sum_{i=1}^{m} \xi_{ij} = 1 \text{ for all } j\}$, that is, each position should have been occupied by exactly one state.

It should also be noted that this representation is completely equivalent to the previous one: for each ξ we have an unique θ and vice versa, and both represent the same point in the space. To see this, it is sufficient to define the one-to-one map $f : \xi \mapsto \theta$, where $f_j(\xi) = \arg_{i=1,\ldots,m}\{\xi_{ij} = 1\}$ and $f_{ij}^{-1}(\theta) = I(\theta_j = i)$. Also note that we still have the same (prior) probability measure on Ξ as we had on Θ, since $P(\xi) = P(f^{-1}(\theta))$.

Define the p-th power loss function as $L_p(\xi, \tilde{\xi}) = \sum_{i=1}^{mn} |\xi_i - \tilde{\xi}_i|^p$, $p > 0$. But

$$\sum_{i=1}^{mn} |\xi_i - \tilde{\xi}_i|^p = \sum_{i=1}^{mn} I(\xi_i \neq \tilde{\xi}_i) = 2 \sum_{i=1}^{n} I\Big((f(\xi))_i \neq (f(\tilde{\xi}))_i\Big)$$

since $\xi_i, \tilde{\xi}_i \in \{0,1\}$ and so $L_p(\xi, \tilde{\xi}) = 2H(f(\xi), f(\tilde{\xi}))$. Thus, the centroid estimator is the posterior risk minimizer for L_p in Ξ.

The centroid estimator coincides, under the ξ-parametrization, with the Bayesian mean when $p = 2$ and with the Bayesian median when $p = 1$; the mode occurs when $p = 0$ and thus is less sensible to distant points. Moreover, when $p = 2$, $\hat{\xi} = f^{-1}(\hat{\theta}_C)$ is the closest point to $\bar{\xi} = E_{\xi|X}\xi$, since

$$\begin{aligned}
\rho_{L_2}(\hat{\xi}) &= E_{\xi|X} L_2(\xi, \hat{\xi}) \\
&= \sum_{\xi \in \Xi} \sum_{i=1}^{mn} (\xi_i - \bar{\xi}_i + \bar{\xi}_i - \hat{\xi}_i)^2 P(\xi|X) \\
&= E_{\xi|X} L_2(\xi, \bar{\xi}) + L_2(\bar{\xi}, \hat{\xi})
\end{aligned}$$

is minimized and so $L_2(\bar{\xi}, \hat{\xi})$ must also be minimum.

1.2. Gain functions and characterization. Before presenting a more adequate characterization of the centroid estimator, let us make some considerations. First, we note that

$$\begin{aligned}\arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X}C(\theta,\tilde{\theta}) &= \arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X}\sum_{i=1}^{n} c(\theta_i,\tilde{\theta}_i)\\ &= \arg\max_{\tilde{\theta}\in\Theta} E_{\theta|X}\sum_{i=1}^{n}\frac{p-c(\theta_i,\tilde{\theta}_i)}{s}\\ &= \arg\max_{\tilde{\theta}\in\Theta} E_{\theta|X}G(\theta,\tilde{\theta})\end{aligned}$$

for arbitrary p and $s>0$, where, from above,

$$G(\theta,\tilde{\theta})=\sum_{i=1}^{n} g(\theta_i,\tilde{\theta}_i)=\sum_{i=1}^{n}\frac{p-c(\theta_i,\tilde{\theta}_i)}{s}.$$

It should be clear that, apart from the location and scale parameters p and s, G and C are equivalent for our purpose of defining $\hat{\theta}_C$. We will be using this representation based on a *gain* function G from now on; the reasons will be clear when we present a characterization of the centroid estimator.

It should also be noted that we could further generalize the gain function by allowing different marginal gain functions,

$$G'(\theta,\tilde{\theta})=\sum_{i=1}^{n} g_i(\theta_i,\tilde{\theta}_i);$$

for now we concentrate on G, but as will be shown later, there is no essential difference between the treatment using G or G'.

Since $A=\{a_j\}_{j=1}^m$ is discrete, G can be more concisely represented as a symmetric matrix where $G_{kj}=(p-c(a_k,a_j))/s$. Note that G has $m(m+1)/2-2$ free parameters. Now, if we denote $\mathbf{e}_a=[I(a=a_j)]_{j=1}^m$, the indicator vector for $a\in A$, then under this representation we have

$$G(\theta,\tilde{\theta})=\sum_{i=1}^{n}\mathbf{e}_{\theta_i}^T G\mathbf{e}_{\tilde{\theta}_i}.$$

The main advantage of this matrix representation is to provide us a concise characterization of the centroid estimator.

THEOREM 1. *Denote by* $\mathbf{p}_i = [P(\theta_i = a_j|X)]_{j=1}^m$ *the vector of marginal posterior probabilities for the i-th component. Then* $\hat{\theta}_C = \arg\max_{\tilde{\theta}\in\Theta} \sum_{i=1}^n \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i$.

PROOF. The result follows directly from the centroid estimator definition:

$$\begin{aligned}
\hat{\theta}_C = \arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X} C(\theta, \tilde{\theta}) &= \arg\max_{\tilde{\theta}\in\Theta} E_{\theta|X} G(\theta, \tilde{\theta}) \\
&= \arg\max_{\tilde{\theta}\in\Theta} E_{\theta|X} \sum_{i=1}^n \mathbf{e}_{\theta_i}^T G \mathbf{e}_{\tilde{\theta}_i} \\
&= \arg\max_{\tilde{\theta}\in\Theta} \sum_{i=1}^n \mathbf{e}_{\tilde{\theta}_i}^T G (E_{\theta|X} \mathbf{e}_{\theta_i}) \\
&= \arg\max_{\tilde{\theta}\in\Theta} \sum_{i=1}^n \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i
\end{aligned}$$

where $\mathbf{p}_i = E_{\theta|X} \mathbf{e}_{\theta_i} = [P(\theta_i = a_j|X)]_{j=1}^m$. □

Theorem 1 makes plain that the marginal posterior probabilities are sufficient to define the centroid estimator. For the remainder of this chapter we concentrate on the centroid estimator and put the question of how to compute marginals aside, simply assuming they are available to determine the centroid estimate. It is however both desired and expected that deriving a centroid estimator should be no harder than computing marginal probabilities. We postpone discussing about the computational complexity of marginalization and centroid estimation to the next chapters, when we address specific applications.

2. Unconstrained centroid estimation

If there are no constraints defining Θ, that is, $\Theta = A^n$, then our centroid estimator can be determined component-wise from Theorem 1:

$$(\hat{\theta}_C)_i = \arg\max_{a \in A} \mathbf{e}_a^T G \mathbf{p}_i = \arg\max_{a_j, j=1,\ldots,m} (G\mathbf{p}_i)_j.$$

In this case we call $\hat{\theta}_C$ a *consensus* estimator, denoted by $\hat{\theta}_C^*$, because we just need to elect, for the i-th component, the maximizer of $G\mathbf{p}_i$, where each component is a linear combination on the marginal posterior probabilities. In the particular case of Hamming loss we have $G = I_m$, the identity of order m, and so $(\hat{\theta}_C^*)_i$ is simply the marginal posterior maximizer. On the other hand, we can obtain a further generalized gain G' by having a matrix G_i for each component; in this case the consensus estimator can be similarly obtained with

$$(\hat{\theta}_C^*)_i = \arg\max_{a_j, j=1,\ldots,m} (G_i\mathbf{p}_i)_j.$$

Let us now explore gain matrices that are related to categorical, ordinal, and interval estimation on Cartesian spaces. We start with a simple case when A is a binary set.

2.1. Binary set factors. When $m = 2$ we can set $A = \{0, 1\}$ and

$$G = \begin{bmatrix} g_0 & g \\ g & g_1 \end{bmatrix} \simeq \begin{bmatrix} 1 & 0 \\ 0 & \gamma \end{bmatrix}$$

by taking location and scale in account, that is, setting $g = 0$ and multiplying G by g_0^{-1} (assuming $g_0 > 0$).

If we further set $\mathbf{p}_i = [1 - \pi_i, \pi_i]^T$ then $G\mathbf{p}_i = [1 - \pi_i, \gamma\pi_i]^T$, and so

$$(\hat{\theta}_C)_i = I(\gamma\pi_i > 1 - \pi_i) = I\left(\pi_i > \frac{1}{1+\gamma}\right).$$

If we take each θ_i to represent an indicator then the parameter γ can be interpreted as a true negative to true positive ratio, a way to measure the sensitivity/specificity

trade-off in estimation. Not surprisingly, when $\gamma = 1$ — as in Hamming loss — the estimation is indifferent to either type of inference error and we have a "true", unbiased consensus.

We might want to be more flexible in our formulation by attributing different sensitivity trade-offs. Such a generalization, as discussed before, leads to gain matrices G_i for each i-th component which, in this case, is equivalent to specifying γ_i to obtain

$$(\hat{\theta}_C)_i = I\left(\pi_i > \frac{1}{1+\gamma_i}\right).$$

2.2. Categorical gain. When comparing categories it is usual to assign a gain when the categories match and, independent of the categories being compared, a smaller gain (often null) when they disagree. Thus, in our matrix representation, categorical inference yields G with the following structure: $G_{jj} = g_j$ for $1 \leq j \leq m$, and $G_{kj} = g_0$ for $1 \leq i \neq j \leq m$ where $g_0 < \min_j g_j$. Furthermore, we can center G by setting $g_0 = 0$, and normalize G by multiplying it by $(\max_j g_j)^{-1}$.

Let $\mathbf{g}$ be the diagonal of G and denote by $*$ the Hadamard product of two vectors: $\mathbf{u} * \mathbf{v} = [u_i v_i]_{i=1}^n$. Our consensus estimator is then

$$(\hat{\theta}_C)_i = \underset{a_j, j=1,\ldots,m}{\arg\max} (\mathbf{g} * \mathbf{p}_i)_j = \underset{a_j, j=1,\ldots,m}{\arg\max} \mathbf{g}_j (\mathbf{p}_i)_j,$$

that is, for each component we select the gain-weighted marginal posterior probability maximizer.

2.3. Ordinal gain. In ordinal gain the categories are evaluated according to a specific order, and we shall assume w.l.g. that the gain from comparing a_k to a_j *does not increase* with $|k-j|$. Let then $\gamma_d = f(d)$, for some non-increasing function f, be the gain from comparing a_k to a_j, $|k-j| = d$. We can now set $\gamma_{m-1} = 0$ to center and $\gamma_0 = 1$ to scale the gain matrix G.

The function f allows for *relative* order between categories as it specifies distinct gains for each absolute order difference d. If $f'' < 0$, f is concave, and categories are relatively more similar, closer in relative order; if $f'' > 0$, f is convex, and categories are relatively less similar, farther in relative order. In this sense a more natural choice

for f is to be linear, $f'' = 0$, in which case the absolute order is considered. The gain matrix is then specified by

$$\gamma_i = 1 - \frac{i}{m-1}$$

for $i = 0, \ldots, m-1$.

It is important to note that G is a *Toeplitz* matrix, that is, all elements in the same diagonal are equal. This property allows $G\mathbf{p}_i$ to be computed with $O(m \log m)$ complexity using the discrete FFT. In fact, if

$$\mathbf{g} = [1, \gamma_1, \ldots, \gamma_{m-2}, \underbrace{0, \ldots, 0}_{m-1}, \gamma_{m-2}, \ldots, \gamma_1]^T$$

and

$$\mathbf{p}'_i = [\underbrace{0, \ldots, 0}_{m-2}, \mathbf{p}_i^T, \underbrace{0, \ldots, 0}_{m-2}]^T$$

then

$$G\mathbf{p}_i = \Big[\mathrm{DFT}\Big(\mathrm{IDFT}(\mathbf{g}) * \mathrm{IDFT}(\mathbf{p}'_i)\Big)\Big]_{i=m-1}^{2m-2}$$

where $\mathrm{DFT}(\mathbf{v})$ and $\mathrm{IDFT}(\mathbf{v})$ are the direct and inverse discrete Fourier transforms of a vector $\mathbf{v}$ and both can be computed on $O(|\mathbf{v}| \log |\mathbf{v}|)$ time using the FFT.

2.4. Interval gain. Another degree of generalization is attained when categories represent intervals in a line and can then not only be partially ordered but also compared according to some distance between them. There is now much more freedom to set the entries in G; however, if the intervals are non-overlapping the categories can be totally ordered and so it should be required that

$$\max_{i,j\,:\,|i-j|=k} G_{ij} \leq \min_{i,j\,:\,|i-j|=k'} G_{ij}$$

whenever $k < k'$ for consistency.

3. Constrained centroid estimation

At first sight, for a general parameter space Θ there might be no other way to find the centroid, defined by

$$\hat{\theta}_C = \arg\max_{\tilde{\theta}\in\Theta} \sum_{i=1}^{n} \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i,$$

then searching Θ exhaustively. However, the set of possible configurations in Θ might be structured enough so as to allow a more efficient optimization approach. In the next sections we present two such cases of this scenario with important applications.

3.1. Binary set factors with exclusivity constraints. Recalling the discussion in Section 2.1, $\Theta \subset \{0,1\}^n$, $G = \mathrm{diag}([1,\gamma])$, and $\pi_i = P(\theta_i = 1|X)$. Since

$$\begin{aligned} \mathbf{e}_{\tilde{\theta}_i}^T G p_i &= \mathbf{e}_{\tilde{\theta}_i}^T \begin{bmatrix} 1-\pi_i \\ \gamma\pi_i \end{bmatrix} \\ &= (1-\pi_i)(1-\tilde{\theta}_i) + \gamma\pi_i\tilde{\theta}_i \\ &= 1-\pi_i + (1+\gamma)\Big[\pi_i - (1+\gamma)^{-1}\Big]\tilde{\theta}_i \end{aligned}$$

for each i-th component, we have

$$\hat{\theta}_C = \arg\max_{\tilde{\theta}\in\Theta} \sum_{i=1}^{n} \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i = \arg\max_{\tilde{\theta}\in\Theta} \sum_{i=1}^{n} \Big[\pi_i - (1+\gamma)^{-1}\Big]\tilde{\theta}_i.$$

If Θ satisfies a linear constraint $\sum_{i\in J} w_i\theta_i \le C$, for some index set $J \subset \{1,\ldots,n\}$, then finding the centroid is equivalent to solving a *knapsack* problem [**MT90**]. In particular, constraints of the type $\sum_{i\in J_k} \theta_i \le 1$, which we call *exclusivity* constraints, are often used when we wish to assign at most one element of J_k to some class k. Solving a knapsack problem is known to be NP-hard [**MT90, PS98**]; however, for centroids defined by exclusivity constraints we have the following result:

THEOREM 2. *If $\Theta \subset \{0,1\}^n$ is such that $\theta \in \Theta$ satisfies a set of conditions $\{C_k\}_{k=1}^K$ of the form C_k: $\sum_{i\in J_k} \theta_i \le 1$, where $\{J_k\}_{k=1}^K$ is a collection of index sets ($J_k \subset$*

$\{1, \ldots, n\}$, $1 \leq k \leq K$), then, for $\gamma \leq 1$, $\hat{\theta}^*_C$ also satisfies each condition C_k, $1 \leq k \leq K$, that is, $\hat{\theta}^*_C \equiv \hat{\theta}_C$.

PROOF. If we take the joint marginal on the k-th constraint we have:

$$\sum_{i \in J_k} P\Big(\big\{\theta : \theta_i = 1, \theta_j = 0, j \in J_k \setminus \{i\}\big\}|X\Big) + P\Big(\big\{\theta_i = 0, i \in J_k\big\}|X\Big) = \sum_{i \in J_k} \pi_i + P\Big(\big\{\theta_i = 0, i \in J_k\big\}|X\Big) = 1$$

and so $\sum_{i \in J_k} \pi_i < 1$. If $(\hat{\theta}^*_C)_i = 0$ for all i, then the consensus estimator respects all constraints trivially. Now suppose otherwise and take some i^* such that $(\hat{\theta}^*_C)_{i^*} = 1$; from the definition of $\hat{\theta}^*_C$, $\pi_{i^*} > 1/(1+\gamma)$ and in addition, when $\gamma \leq 1$ as stated, $\pi_{i^*} > 1/2$. But then

$$\sum_{i \in J_k \setminus \{i^*\}} \pi_i < 1 - \pi_{i^*} < 1/2$$

which guarantees that $i^* = \arg\max_{i \in J_k} \pi_i$. Since the choice of i^* and k was arbitrary, i^* is the maximizing index for all C_k such that $i^* \in J_k$ and thus all constraints are consistently satisfied. □

A couple of remarks are in order. First we note that as we lower γ below unity we increase our demands on specificity of the centroid estimator. As a result, some position i might not be selected even if it is likely, that is, we might have set $(\hat{\theta}_C)_i = 1$ if $\gamma \leq 1$ were higher. A good advice is then to examine the centroid estimator pondering the selection of each of its components according to our choice of γ: a more specific criterion yields a more reliable selection in which we are proportionally more confident on $(\hat{\theta}_C)_i = 1$ than on $(\hat{\theta}_C)_i = 0$.

The main idea behind Theorem 2 is that if $i \in J_k$ and $i \in J_{k'}$ for some k, k' then there should be no conflict when setting $(\hat{\theta}_C)_i$ from each constraint independently since such a conflict would contradict the maximality of π_i. For this reason conditions C_k are called exclusivity constraints. Exclusivity constraints are used in Chapter 4 when we discuss sequence alignment and RNA secondary structure prediction.

3.2. Transition models. In many cases a parameter θ represents a *walk* in a graph $G = (S, F)$. Vertex set S is the called the *state* set and F the *transition* set. The possible starting states for a walk are represented by an *initial* (state) set $I \subset S$. A simple example of G is illustrated in Figure 2.2; a typical θ in this case should not have $\theta_i = 0$ for $i > \min_j\{\theta_j \neq 0\}$, that is, once the walk leaves state 0 it cannot return to this state.

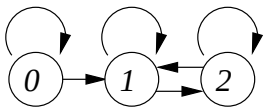

FIGURE 2.2. Example of transition graph $G = (S, F)$, where $S = \{0, 1, 2\}$, $I = \{0\}$, and $F = \{(0, 0), (0, 1), (1, 1), (1, 2), (2, 1), (2, 2)\}$.

The transition graph can be used to represent constraints on consecutive components of θ and thus is the main ingredient of our next model.

DEFINITION 2 (Transition model). A *transition model* that generates a sequence X of length n as data is defined by a parameter space $\Theta \subset S^n$ such that $\theta \in \Theta$ if and only if $\theta_1 \in I$ and $(\theta_i, \theta_{i+1}) \in F$ for $1 \leq i < n$, a (prior) probability distribution on Θ, and conditional probability distributions on X given $\theta \in \Theta$.

Since Θ comprises all *walks* of length n in G starting at some state in I and with transitions in F, it is likely that $|\Theta| \gg n$. When $|\Theta|$ is large, computing the marginal posterior probabilities $\mathbf{p}_i$ is, in general, a difficult task. However for specific transition models of interest — namely hidden Markov and change point models — the marginal posterior probabilities can be computed in polynomial time. These models are presented in the next chapter. In contrast to marginalization, however, the centroid (walk) estimator is always obtained efficiently as follows: we compute partial maximum sums C_k and back pointers b_k recursively using dynamic programming, as performed by Algorithm 2.1.

THEOREM 3. *Algorithm 2.1 computes the centroid estimator for transition models.*

Input: Marginal posterior probabilities $\mathbf{p}_i$, $i = 1, \ldots, n$, gain matrix G.
Output: Centroid estimate $\hat{\theta}_C$.
`% initialization`
for $s \in S$ **do**
  if $s \in I$ **then** $C_0(s) \leftarrow 0$; **else** $C_0(s) \leftarrow -\infty$;
end
`% recursion`
for $k = 1, \ldots, n$ **do**
  for $s \in S$ **do**
    $C_k(s) \leftarrow \mathbf{e}_s^T G \mathbf{p}_k + \max_{t \in S:(t,s) \in F} C_{k-1}(t)$;
    $b_k(s) \leftarrow \arg\max_{t \in S:(t,s) \in F} C_{k-1}(t)$;
  end
end
`% termination`
$C_{n+1} \leftarrow \max_{s \in S} C_n(s)$; `%` $\max_{\tilde{\theta} \in \Theta} \sum_{i=1}^{n} \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i$
$b_{n+1} \leftarrow \arg\max_{s \in S} C_n(s)$;
`% backtrack to recover` $\hat{\theta}_C$
$(\hat{\theta}_C)_n \leftarrow b_{n+1}$;
for $k = n-1, \ldots, 1$ **do**
  $(\hat{\theta}_C)_k \leftarrow b_{k+1}((\hat{\theta_C})_{k+1})$;
end

ALGORITHM 2.1: Centroid estimator for transition models.

PROOF. The proof follows directly from induction. If $n = 1$ then

$$C_2 = \max_{\tilde{\theta} \in \Theta} \sum_{i=1}^{n} \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i = \max_{s \in S} \mathbf{e}_s^T G \mathbf{p}_1 = \max_{s \in S} C_1(s),$$

$(\hat{\theta}_C) = b_2 = \arg\max_{s \in S} C_1(s)$, and the algorithm is correct. Let us assume now that

$$C_k(s) = \mathbf{e}_s^T G \mathbf{p}_k + \max_{\tilde{\theta}_1, \ldots, \tilde{\theta}_{k-1}: \tilde{\theta} \in \Theta} \sum_{i=1}^{k-1} \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i$$

for some $k < n$ is the partial maximum sum up to position $k-1$ ending at state s such that $(\tilde{\theta}_{k-1}, s) \in F$. Then, by induction,

$$\begin{aligned} C_{k+1}(s) &= \mathbf{e}_s^T G \mathbf{p}_{k+1} + \max_{t \in S:(t,s)\in F} C_k(t) \\ &= \mathbf{e}_s^T G \mathbf{p}_{k+1} + \max_{t \in S:(t,s)\in F} \left\{ \mathbf{e}_t^T G \mathbf{p}_k + \max_{\tilde{\theta}_1,\ldots,\tilde{\theta}_{k-1}:\tilde{\theta}\in\Theta} \sum_{i=1}^{k-1} \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i \right\} \\ &= \mathbf{e}_s^T G \mathbf{p}_{k+1} + \max_{\tilde{\theta}_1,\ldots,\tilde{\theta}_k:\tilde{\theta}\in\Theta} \sum_{i=1}^{k} \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i. \end{aligned}$$

Moreover, since we are keeping track of which component was chosen as argument maximizer for the running sum in b, $\hat{\theta}_C$ can correctly be recovered from b. □

Algorithm 2.1 is also efficient as it runs in polynomial time: in the recursion step, for $1 \leq k \leq n$, we check all elements of F when assessing the maximum over all back neighbors of each $s \in S$, and so the complexity is $\mathcal{O}(n|F|)$, and hence *linear* in n.

4. k-th order centroid estimation

The regular centroid estimator, defined by

$$\hat{\theta}_C = \arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X} H(\theta, \tilde{\theta}) = \arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X} \sum_{i=1}^{n} \Big[1 - I(\theta_i = \tilde{\theta}_i) \Big]$$

is a particular case of the generalized centroid estimator, as previously stated. The estimation is driven by the sum of the posterior marginals for each component, as discussed above. On the other extreme we have the maximum *a posteriori* (MAP) estimator

$$\hat{\theta}_M = \arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X} I(\theta \neq \tilde{\theta}) = \arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X} \left[1 - \prod_{i=1}^{n} I(\theta_i = \tilde{\theta}_i) \right]$$

where the estimation is based on a zero-one loss function and is driven by the posterior full joint distribution.

Another possible generalization of centroid estimation is to provide a class of estimators that fills the gap between $\hat{\theta}_C$ and $\hat{\theta}_M$.

DEFINITION 3 (k-th order centroid estimator). Define

$$\mathcal{J}_k = \big\{ J \subset \{1, \ldots, n\} \,:\, |J| = k \big\},$$

the family of index sets with cardinality k. The *k-th order* centroid estimator is then defined as

$$\hat{\theta}_C^{(k)} = \arg\min_{\tilde{\theta} \in \Theta} E_{\theta|X} \sum_{J \in \mathcal{J}_k} \left[1 - \prod_{i \in J} I(\theta_i = \tilde{\theta}_i) \right].$$

It follows that $\hat{\theta}_C = \hat{\theta}_C^{(1)}$, the original centroid is simply the first order centroid, and $\hat{\theta}_M = \hat{\theta}_C^{(n)}$, the MAP estimator is the n-th order centroid. As an example, consider the second order centroid estimator:

$$\begin{aligned} \hat{\theta}_C^{(2)} &= \arg\min_{\tilde{\theta} \in \Theta} E_{\theta|X} \sum_{i=1}^{n-1} \sum_{j>i}^{n} \Big[1 - I(\theta_i = \tilde{\theta}_i) I(\theta_j = \tilde{\theta}_j) \Big] \\ &= \arg\max_{\tilde{\theta} \in \Theta} \sum_{i=1}^{n-1} \sum_{j>i}^{n} P(\theta_i = \tilde{\theta}_i, \theta_j = \tilde{\theta}_j | X). \end{aligned}$$

The estimation of the k-th order centroid involves the posterior joint probability of each of the possible combinations on k components. Even for unconstrained spaces this is in general a hard undertaking. However, depending on the structure of the parameter space we can further restrict our centroid estimator.

A first restriction is when we expect the parameter components to exhibit only *local* dependence. In this case, by focusing on short range effects between components, we can define a *sequential* k-th order centroid:

$$\hat{\theta}_C^{[k]} = \arg\min_{\tilde{\theta} \in \Theta} E_{\theta|X} \sum_{j=1}^{n-k+1} \left[1 - \prod_{i=0}^{k-1} I(\theta_{j+i} = \tilde{\theta}_{j+i}) \right].$$

A common case is when the parameter space specifies a *graph model*, in which case the interest resides only in the posterior joint distribution of parameter components that form a clique in the model. This is the topic of the next section, where we concentrate on a specific, more structured parameter space and argue that k-th order centroid estimation can be reasonably restrained and becomes more tractable.

5. Graph models

We now extend the transition models to consider higher order interactions among components of our parameter space. Given a state space S, let $G = (V, E)$, $V = \{1, \ldots, n\}$, be a graph representing the *structure* of Θ: each vertex $i \in V$ represents a component in θ, and we require that $\theta_i \in S$ and for each clique K in G we define a set of constraints $\mathcal{C}(K) \subset S^{|K|}$, the possible realizations of $(\theta_i)_{i \in K}$. Of course, K is only constrained in fact if $\mathcal{C}(K)$ is a proper subset of $S^{|K|}$. We call G the *structure* graph of Θ.

We denote the *restriction* of $R = (r_i)_{i=1}^m$ to the index set $J = \{j_i\}_{i=1}^l$ as $R[J] = (r_{j_i})_{i=1}^l$. As an example, if $K = \{3, 6, 7\}$ is a clique in G, $\theta[K] = (\theta_3, \theta_6, \theta_7)$. If $S = \{0, 1\}$, all triples $K = (i, 2i, 2i{+}1)$ are cliques in G, and $\mathcal{C}(K) = \{(0,0,0), (0,1,1), (1,0,1), (1,1,0)\}$, then both $\theta[K] = (1,1,1)$ and $\theta[K] = (0,0,1)$ are invalid for any triple K. The conditional restriction of Θ to V_j given U from V_i is $\Theta[V_j|V_i(U)] = \{R \in \Theta[V_j] : R_v = U_v, v \in V_i\}$. If $V_j = \{1,2,3\}$, $V_i = \{2,4,5\}$ and $U = \{(0,0,0),(0,1,1)\}$, then $\Theta[V_j|V_i(U)] = \{(0,0,0),(1,0,1)\}$ since $\theta_2 = 0$ from V_i and U.

Definition 4 (Graph model). A *graph model* that generates a sequence X of length n as data is defined by a parameter space $\Theta \subset S^n$ such that $\theta \in \Theta$ if and only if for each clique K in the structure graph G of Θ we have $\theta[K] \in \mathcal{C}(K)$, a probability distribution on Θ, and conditional probability distributions on X given $\theta \in \Theta$.

A graph model is more general than a transition model. In particular, the structure graph for a transition model is a path graph, and so, since the cliques in the transition model are edges, we can represent constraints as transitions. We also know that the centroid estimator can be obtained in linear time in n for transition models. Under which conditions can we have a similar result for graph models? We were able to exploit the linear structure of a transition model to find the centroid estimator using dynamic programming: the partial sum up to position i could be extended from the partial sum at position $i-1$ since the only restrictions we needed to verify occurred in consecutive positions. Clearly, to guarantee the satisfaction of constraints in graph

models we need to check every clique in G, and so the task of finding the centroid becomes harder as the cliques in G grow in size.

Ultimately, our main concern is with the "connectivity" of G, since the higher G is connected the more difficult it is to combine partial sums to define the centroid. In transition models G is a path, which is minimally connected: if any vertex is removed from G the graph becomes disconnected. The smaller the tree-width of a graph G the less G is connected and the more G resembles a tree. As a matter of fact, it is not hard to see that trees have unitary tree-width and that since a clique should be wholly contained in a part of a tree-decomposition, the tree-width of a graph is lower bounded by the size of the maximum clique in the graph. However, given the tree structure provided by a tree-decomposition of G, we should be able to devise an algorithm that computes, stores, and updates optimal partial solutions of our centroid estimation optimization problem, much like in a dynamic programming approach and similar to the class of algorithms described in [**Bod97**].

The technique consists of representing partial solutions — in our case, partial sums — as tables that are stored at each node of the rooted tree T and computed in bottom-up order, from the leaves to the root. The order of execution is then defined by a *post-ordering* $\sigma : V(T) \mapsto \{1, \ldots, |V(T)|\}$ of the nodes. The table for each node $i \in T$ is computed from the tables of the children of i when restricted to V_i and the sum at V_i for each feasible configuration, as the algorithm progresses inductively by building the partial sum for the subtree rooted at i. Since we need to construct the centroid, we also store backtrack tables and proceed in top-down order from the root to find our estimator. Algorithm 2.2 stores the partial maximum sums in C_i and backtrack pointers in B_{ij} for $i \in T$ and $j = \textsc{Child}_T(i)$. We denote by $\textsc{Desc}_T(i) = \cup_{j \in T_i} V_j$.

THEOREM 4. *Algorithm 2.2 computes the centroid estimator for graph models.*

PROOF. The proof is very similar to the proof for Algorithm 2.1: as we iteratively compute tables of partial solutions along the post-ordering of T, we apply induction

Input: Marginal posterior probabilities $\mathbf{p}_v$, $v \in V$, gain matrix G.
Output: Centroid estimate $\hat{\theta}_C$.
Data: Tree-decomposition $(T, \{V_i\}_{i \in T})$ rooted at r, post-order σ of T from r.

```
% recursion
```

for $k = 1, \ldots, |T|$ **do**
  $i \leftarrow \sigma^{-1}(k)$;
  for $U \in \Theta[V_i]$ **do**
    $C_i(U) \leftarrow \sum_{v \in V_i \setminus \textsc{Desc}_T(i)} \mathbf{e}_{U_v}^T G \mathbf{p}_v$;
    for $j \in \textsc{Child}_T(i)$ **do**
      $C_i(U) \leftarrow C_i(U) + \max_{R \in \Theta[V_j | V_i(U)]} C_j(R)$;
      $B_{ij}(U) \leftarrow \arg\max_{R \in \Theta[V_j | V_i(U)]} C_j(R)$;
    end
  end
end

```
% termination
```

$C_0 \leftarrow \max_{U \in \Theta[V_r]} C_r(U)$; % $\max_{\tilde{\theta} \in \Theta} \sum_{i=1}^n \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i$
$B_0 \leftarrow \arg\max_{U \in \Theta[V_r]} C_r(U)$;

```
% backtrack to recover θ̂_C
```

$\hat{\theta}_C[V_r] \leftarrow B_0$;
for $k = |T| - 1, \ldots, 1$ **do**
  $i \leftarrow \sigma^{-1}(k)$;
  $j \leftarrow \textsc{Parent}_T(i)$;
  $\hat{\theta}[V_i] \leftarrow B_{ji}(\hat{\theta}_C[V_j])$;
end

ALGORITHM 2.2: Centroid estimator for graph models.

on the size of T. If $T = \{r\}$ then

$$C_0 = \max_{\tilde{\theta} \in \Theta} \sum_{i=1}^n \mathbf{e}_{\tilde{\theta}_i}^T G \mathbf{p}_i = \max_{U \in \Theta[V_r]} C_r(U) = \max_{U \in \Theta} \sum_{v \in V_r} \mathbf{e}_{U_v}^T G \mathbf{p}_v$$

and $B_0 = \arg\max_{U \in \Theta} \sum_{v \in V_r} \mathbf{e}_{U_v}^T G \mathbf{p}_v$. By the induction hypothesis,

$$C_j(U) = \sum_{v \in V_j} \mathbf{e}_{U_v}^T G \mathbf{p}_v + \max_{\tilde{\theta} \in \Theta[\textsc{Desc}_T(j) \setminus V_j]} \sum_{u \in \textsc{Desc}_T(j) \setminus V_j} \mathbf{e}_{\tilde{\theta}_u}^T G \mathbf{p}_u$$

is the partial maximum sum for the subtree rooted at j with $\tilde{\theta}[V_j] = U$. From the recursive step on C_i we have

$$C_i(U) = \sum_{v \in V_i \setminus \textsc{Desc}_T(i)} \mathbf{e}_{U_v}^T G \mathbf{p}_v + \sum_{j \in \textsc{Child}_T(i)} \max_{R \in \Theta[V_j | V_i(U)]} C_j(R),$$

where C_j, for $j \in \textsc{Child}_T(i)$, is always available since $\sigma(j) < \sigma(i)$.

But

$$\max_{R \in \Theta[V_j | V_i(U)]} C_j(R) = \max_{\tilde{\theta} \in \Theta[\textsc{Desc}_T(j) | V_i(U)]} \sum_{u \in \textsc{Desc}_T(j)} \mathbf{e}_{\tilde{\theta}_u}^T G \mathbf{p}_u$$

from the hypothesis and

$$\sum_{v \in V_i \setminus \textsc{Desc}_T(i)} \mathbf{e}_{U_v}^T G \mathbf{p}_v = \sum_{v \in V_i} \mathbf{e}_{U_v}^T G \mathbf{p}_v - \sum_{j \in \textsc{Child}_T(i)} \sum_{w \in V_i \cap V_j} \mathbf{e}_{U_w}^T G \mathbf{p}_w$$

and so

$$\begin{aligned}
C_i(U) &= \sum_{v \in V_i} \mathbf{e}_{U_v}^T G \mathbf{p}_v - \sum_{j \in \textsc{Child}_T(i)} \sum_{w \in V_i \cap V_j} \mathbf{e}_{U_w}^T G \mathbf{p}_w \\
&+ \sum_{j \in \textsc{Child}_T(i)} \max_{\tilde{\theta} \in \Theta[\textsc{Desc}_T(j) | V_i(U)]} \sum_{u \in \textsc{Desc}_T(j)} \mathbf{e}_{\tilde{\theta}_u}^T G \mathbf{p}_u \\
&= \sum_{v \in V_i} \mathbf{e}_{U_v}^T G \mathbf{p}_v + \sum_{j \in \textsc{Child}_T(i)} \max_{\tilde{\theta} \in \Theta[\textsc{Desc}_T(j) \setminus V_i]} \sum_{u \in \textsc{Desc}_T(j) \setminus V_i} \mathbf{e}_{\tilde{\theta}_u}^T G \mathbf{p}_u \\
&= \sum_{v \in V_i} \mathbf{e}_{U_v}^T G \mathbf{p}_v + \max_{\tilde{\theta} \in \Theta[\textsc{Desc}_T(i) \setminus V_i]} \sum_{j \in \textsc{Child}_T(i)} \sum_{u \in \textsc{Desc}_T(j) \setminus V_i} \mathbf{e}_{\tilde{\theta}_u}^T G \mathbf{p}_u \\
&= \sum_{v \in V_i} \mathbf{e}_{U_v}^T G \mathbf{p}_v + \max_{\tilde{\theta} \in \Theta[\textsc{Desc}_T(i) \setminus V_i]} \sum_{u \in \textsc{Desc}_T(i) \setminus V_i} \mathbf{e}_{\tilde{\theta}_u}^T G \mathbf{p}_u.
\end{aligned}$$

While computing C_i from bottom-up in T we also build B_{ij} from the argument maximizers. It is now straightforward to correctly recover $\hat{\theta}_C$ from B_{ij} in top-down order from T. □

The complexity of Algorithm 2.2 depends heavily on the computation of tables for each node in T, and since the size of the table for node i is $\mathcal{O}(|S|^{|V_i|})$, the complexity grows exponentially with the tree-width of G. If we however restrict G to be a graph

of *bounded tree-width*, $\mathrm{TW}(G) \leq \kappa$, then the size of each table becomes $\mathcal{O}(|S|^{\kappa})$ and with $|T| = \mathcal{O}(n)$ recursive operations we have a total linear complexity of $\mathcal{O}(n|S|^{\kappa})$, albeit with a likely large constant.

5.1. Graph centroid. Consider now the k-th order centroid in light of the structure graph G. It seems redundant to define an estimator that takes an index set J where the components are not adjacent in G; in fact, we might not even be interested in accounting for index sets with size exactly k or that greatly overlap. A tree-decomposition of G provides us a way to address the connectivity of G and define a loss function for our next estimator.

DEFINITION 5 (Graph centroid estimator). Let $\mathcal{T} = (T, \{V_i\}_{i \in T})$ be a tree-decomposition for the structure graph G of a graph model. The *graph centroid estimator* is defined as

$$\hat{\theta}_{G,\mathcal{T}} = \arg\min_{\tilde{\theta} \in \Theta} E_{\theta|X} \sum_{i \in T} \left[1 - \prod_{v \in V_i} I(\theta_v = \tilde{\theta}_v) \right].$$

Most of the interest in the graph centroid resides on minimum tree-decompositions of G — decompositions that realize the tree-width of G — where the parts V_i are judiciously chosen to avoid redundancy and reduce overlap. The motivation is to generalize the centroid estimator and minimally account for the connectivity of G. In any case, since there may exist many minimum tree-decompositions for G, we should attend to which components of θ belong to each part in $\mathcal{T}$, or, put differently, we should design the influence of each component in the loss function according to its relevance. On the other extreme, note that for the trivial tree-decomposition $\mathcal{T}_0 = (\{r\}, \{V\})$, $\hat{\theta}_{G,\mathcal{T}_0}$ is the MAP estimator.

Not surprisingly, the graph centroid is easily characterized by the marginal joints over the parts of the tree-decomposition of G:

$$\begin{aligned}\hat{\theta}_{G,\mathcal{T}} &= \arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X} \sum_{i\in T}\left[1 - \prod_{v\in V_i} I(\theta_v = \tilde{\theta}_v)\right] \\ &= \arg\max_{\tilde{\theta}\in\Theta} \sum_{i\in T} E_{\theta|X} \prod_{v\in V_i} I(\theta_v = \tilde{\theta}_v) \\ &= \arg\max_{\tilde{\theta}\in\Theta} \sum_{i\in T} P(\theta[V_i] = \tilde{\theta}[V_i]|X) = \arg\max_{\tilde{\theta}\in\Theta} \sum_{i\in T} P_i(\tilde{\theta}[V_i]),\end{aligned}$$

where, to simplify the notation, we defined $P_i(U) = P(\theta[V_i] = U|X)$.

Since we are maintaining the same structure of the graph model, deriving the graph centroid can be accomplished easily from an adaptation of Algorithm 2.2 in Algorithm 2.3. In fact, both algorithms have even the same complexity when $\mathrm{TW}(G) \leq \kappa$.

THEOREM 5. *Algorithm 2.3 computes the graph centroid estimator.*

PROOF. The proof follows from a direct adaptation of the proof for Algorithm 2.2, as both algorithms take advantage of the structure induced by the tree decomposition. The induction is simpler, however: if $T = \{r\}$ then $\max_{\tilde{\theta}\in\Theta} \sum_{i\in T} P_i(\tilde{\theta}[V_i]) = \max_{\tilde{\theta}\in\Theta} C_r(\tilde{\theta})$, and, by the induction hypothesis,

$$C_j(U) = \max_{R\in\Theta[\mathrm{DESC}_T(j)|V_j(U)]} \sum_{k\in T_j} P_k(R[V_k])$$

Data: Tree-decomposition $\mathcal{T} = (T, \{V_i\}_{i \in T})$ rooted at r, post-order σ of T from r.
Input: Marginal posterior probabilities $P_i(U)$ for $i \in T$.
Output: Graph centroid estimate $\hat{\theta}_{G,\mathcal{T}}$.

```
% recursion
```
for $k = 1, \ldots, |T|$ **do**
  $i \leftarrow \sigma^{-1}(k)$;
  for $U \in \Theta[V_i]$ **do**
    $C_i(U) \leftarrow P_i(U)$;
    for $j \in \textsc{Child}_T(i)$ **do**
      $C_i(U) \leftarrow C_i(U) + \max_{R \in \Theta[V_j|V_i(U)]} C_j(R)$;
      $B_{ij}(U) \leftarrow \arg\max_{R \in \Theta[V_j|V_i(U)]} C_j(R)$;
    end
  end
end
```
% termination
```
$C_0 \leftarrow \max_{U \in \Theta[V_r]} C_r(U)$; % $\max_{\tilde{\theta} \in \Theta} \sum_{i \in T} P_i(\tilde{\theta}[V_i])$
$B_0 \leftarrow \arg\max_{U \in \Theta[V_r]} C_r(U)$;
```
% backtrack to recover $\hat{\theta}_{G,\mathcal{T}}$
```
$\hat{\theta}_{G,\mathcal{T}}[V_r] \leftarrow B_0$;
for $k = |T| - 1, \ldots, 1$ **do**
  $i \leftarrow \sigma^{-1}(k)$;
  $j \leftarrow \textsc{Parent}_T(i)$;
  $\hat{\theta}_{G,\mathcal{T}}[V_i] \leftarrow B_{ji}(\hat{\theta}_C[V_j])$;
end

Algorithm 2.3: Graph centroid estimator.

and so

$$\begin{aligned}
C_i(U) &= P_i(U) + \sum_{j \in \textsc{Child}_T(i)} \max_{R \in \Theta[V_j|V_i(U)]} C_j(R) \\
&= P_i(U) + \sum_{j \in \textsc{Child}_T(i)} \max_{R \in \Theta[\textsc{Desc}_T(j)|V_i(U)]} \sum_{k \in T_j} P_k(R[V_k]) \\
&= P_i(U) + \max_{R \in \Theta[\textsc{Desc}_T(i)|V_i(U)]} \sum_{j \in \textsc{Child}_T(i)} \sum_{k \in T_j} P_k(R[V_k]) \\
&= \max_{R \in \Theta[\textsc{Desc}_T(i)|V_i(U)]} \sum_{j \in T_i} P_j(R[V_j]).
\end{aligned}$$

The correctness of constructing $\hat{\theta}_G$ from B_{ij} now follows from Theorem 4 since the backtrack tables are identical for both algorithms. □

5.2. Sequential k-th order centroid. In the previous section we have used a tree-decomposition to define a loss function. We can now adopt the complementary approach for the sequential k-th order centroid: define a *k-th order sequential graph* G_k on n vertices — containing the sequential interactions we want to capture — as $E(G_k) = \{(i,j) : 0 < |i-j| < k\}$.

FACT 6. *G_k is a $(k-1)$-tree.*

PROOF. We use induction on $n = |V(G)|$. When $n = k$ then G_k is complete, and so a $(k-1)$-tree. Suppose now that G_k is a $(k-1)$-tree for some $n > k$; but then, by the definition of G_k, if we add vertex $v = n+1$ then v should be adjacent to $n-k-1, \ldots, n$ which form a clique since no two of them are k positions apart, and so G_k remains a $(k-1)$-tree. □

By taking k consecutive components, we can build a tree-decomposition for G_k: a *path* $T = (\{1, \ldots, n-k+1\}, \{(i, i+1)\}_{i=1}^{n-k})$ and parts $V_i = \{i, \ldots, i+k-1\}$. Since $|V_i| = k$ for all $i \in T$, we know that this is a minimum tree-decomposition. Moreover, the graph centroid estimator is exactly the sequential k-th order centroid, and so we now know how to compute the latter in time $\mathcal{O}(n|S|^{k-1})$.

6. Credibility Intervals

Given a point $\tilde{\theta}$ on our parameter space Θ and a real number $0 \leq \alpha \leq 1$, we start by defining the *α-credibility level* around $\tilde{\theta}$ as

$$D_\alpha(\tilde{\theta}) = \min_{0 \leq D \leq n} \left\{ P\Big(\{\theta : H(\theta, \hat{\theta}) \leq D\} | X\Big) \geq \alpha \right\}$$

and the *α-credibility level* as $D^*_\alpha = \min_{\tilde{\theta} \in \Theta} D_\alpha(\hat{\theta})$. Ideally we want to select an estimator $\hat{\theta}$ that is tight in the sense that $D_\alpha(\hat{\theta}) = D^*_\alpha$, but such an estimator could

not be unique since many points might satisfy this condition for a fixed α. We say that $\hat{\theta}$ is in the *α-tight credibility set* (or that $\hat{\theta}$ is *α-tight*) if $P(\{\theta : H(\theta, \hat{\theta}) \leq D^*_\alpha\}|X) \geq \alpha$.

Instead, we could pick a different criterion and try to minimize a *weighted* version of D_α over α,

$$D_F(\hat{\theta}) = \int_0^1 D_\alpha(\hat{\theta}) F(\alpha) d\alpha.$$

In particular, if F is constant, we have the following result:

THEOREM 7. *When $F(\alpha) \propto 1$, $\hat{\theta}_C = \arg\min_{\tilde{\theta} \in \Theta} D_F(\tilde{\theta})$.*

PROOF. We want to minimize

$$\begin{aligned} D(\tilde{\theta}) &= \int_0^1 D_\alpha(\tilde{\theta}) d\alpha \\ &= \int_{\alpha_0=0}^{\alpha_1} 0 \, d\alpha + \int_{\alpha_1}^{\alpha_2} 1 \, d\alpha + \cdots + \int_{\alpha_n}^{\alpha_{n+1}=1} n \, d\alpha \\ &= \sum_{i=1}^{n+1} \int_{\alpha_{i-1}}^{\alpha_i} (i-1) d\alpha = \sum_{i=1}^{n+1} (i-1)(\alpha_i - \alpha_{i-1}) \\ &= \sum_{i=0}^{n} i \cdot P\Big(H(\theta, \tilde{\theta}) = i | X\Big) = E_{\theta|X} H(\theta, \tilde{\theta}) \end{aligned}$$

where $\alpha_i = P(\{\theta : H(\theta, \tilde{\theta}) \leq i - 1\}|X)$ and so, by the definition of $\hat{\theta}_C$, the result is proven. □

Since we usually have no reason to put more weight on any particular α, the previous theorem shows us that the centroid is a reasonably tight estimator, at least in this non-informative averaged sense.

If the posterior probability distribution P on Θ is *separable*, that is, if $P(\theta|X) = \prod_{i=1}^n P(\theta_i|X)$, the centroid estimator can be shown to be *uniformly* tight, as the next theorem claims.

THEOREM 8. *If P is separable, then $\hat{\theta}_C$ is α-tight for $0 \leq \alpha \leq 1$.*

PROOF. Let $h_k(\tilde{\theta}) = P(\{\theta : H(\theta, \tilde{\theta}) = k\}|X)$ and $q_i = P(\theta_i = \tilde{\theta}_i|X)$. Then

$$\begin{aligned} H_k(\tilde{\theta}) &= P(\{\theta : H(\theta, \tilde{\theta}) \leq k\}|X) = \sum_{l=0}^{k} h_l(\tilde{\theta}) \\ &= \underbrace{\prod_{i=1}^{n} q_i}_{h_0(\tilde{\theta})} + \underbrace{\prod_{i=1}^{n} q_i \left[\sum_{i=1}^{n} \left(\frac{1-q_i}{q_i} \right) \right]}_{h_1(\tilde{\theta})} + \underbrace{\prod_{i=1}^{n} q_i \left[\sum_{i=1}^{n} \sum_{j \neq i} \left(\frac{1-q_i}{q_i} \right) \left(\frac{1-q_j}{q_j} \right) \right]}_{h_2(\tilde{\theta})} \\ &\quad + \cdots + \underbrace{\prod_{i=1}^{n} q_i \left[\sum_{i_1=1}^{n} \sum_{i_2 \neq i_1} \cdots \sum_{i_k \neq i_{k-1}, \ldots, i_1} \prod_{l=1}^{k} \left(\frac{1-q_{i_l}}{q_{i_l}} \right) \right]}_{h_k(\tilde{\theta})} \end{aligned}$$

If we select a position i, then $H_k(\tilde{\theta})$ and be seen as a function of q_i as

$$H_k(\tilde{\theta}) = q_i C_k \left[\left(\frac{1-q_i}{q_i} \right) T_k + R_k \right]$$

where $C_k = \prod_{j \neq i} q_j > 0$, R_k, and T_k do not depend on q_i. Moreover,

$$R_k - T_k = \sum_{i_1 \neq i} \sum_{i_2 \neq i_1} \cdots \sum_{i_k \neq i_{k-1}, \ldots, i_1} \prod_{l=1}^{k} \left(\frac{1-q_{i_l}}{q_{i_l}} \right) > 0$$

contains the higher order terms in $H_k(\tilde{\theta})$ that do not depend on q_i. Now,

$$\begin{aligned} \frac{\partial H_k(\tilde{\theta})}{\partial q_i} &= C_k \Big[(1/q_i - 1)T_k + R_k + q_i(-T_k/q_i^2) \Big] \\ &= C_k(R_k - T_k) > 0 \end{aligned}$$

and so we just need to maximize each $q_i = P(\theta_i = \tilde{\theta}_i|X)$ to maximize $H_k(\tilde{\theta})$ for any $0 \leq k \leq n$. Then, clearly, $\hat{\theta}$ maximizes $H_k(\tilde{\theta})$ and should be α-tight for $\alpha \leq H_k(\hat{\theta}_C)$. Since this is valid for all k, $\hat{\theta}_C$ is tight for all α, as desired. □

The applicability of the previous theorem might seem reduced since the class of posterior distributions is too specific. However, in cases when the posterior distribution is too complex we can resort to a variational Bayes method [**Att00, BG03**] where the approximating posterior is separable and so the resulting centroid is uniformly tight.

CHAPTER 3

Applications of Centroid Estimation

In the previous chapter we presented the centroid estimator. We characterized the estimation procedure as a discrete optimization problem defined on marginal posterior distributions. In contrast to the more abstract approach we adopted when discussing general properties of the centroid estimator, here we focus on specific, but broad, applications: clustering, variable selection, change point models, hidden Markov models, and graphical models.

The centroid for these applications fall into the same categories from the previous chapter, and so we know how to find our estimator; now we just need to obtain the marginal posterior distributions. For the following applications — and for illustrative purposes — we are going to settle with a simple specification. To set the likelihood of θ we assume that X_i is conditionally independent given θ and a set of hyper-parameters η, $P(X|\theta,\eta) = \prod_{i=1}^{n} P(X_i|\theta_i,\eta)$.

1. Partition set pertinence models

Our problem in this section is a *classification* problem: we are given n objects, which we represent by the set $J = \{1,\ldots,n\}$, and we wish to assign each object to exactly one group from a family $\{S_j\}_{j=1}^{m}$. Once we have assigned the objects to their groups we should have $\cup_{j=1}^{m} S_j = J$ and, since $\cap_{j=1}^{m} S_j = \emptyset$, $\{S_j\}_{j=1}^{m}$ is said to be a *partition* family.

If each i-th component of our parameter space $\Theta = \{1,\ldots,m\}^n$ represents the pertinence of an element i to group j, that is, $\theta_i = j$ if and only if $i \in S_j$, $i = 1,\ldots,n$, then we have a (partition) set pertinence model. Since Θ arises from a Cartesian product we can immediately conclude that the centroid estimator is a consensus estimator. Now, to derive our estimator we just need to define the distribution on Θ.

The prior can be vaguely informative and depend on the size of each set S_j, $n_j = \sum_{i=1}^{n} I(\theta_i = j)$,

$$P(\theta|\eta) = P(\theta|n_1, \ldots, n_m)P(n_1, \ldots, n_m|\eta) = \frac{\prod_{j=1}^{m} n_j!}{n!} P(n_1, \ldots, n_m|\eta),$$

where a usual choice for $P(n_1, \ldots, n_m|\eta)$ is the multinomial $n_1, \ldots, n_m|\eta \sim MN(n, \eta)$, $\eta = \{\alpha_1, \ldots, \alpha_m\}$ with $\alpha_j > 0$, $j = 1, \ldots, m$, and $\sum_{j=1}^{m} \alpha_j = 1$. A non-informative prior can be obtained when $\alpha_j = 1/m$ for all j, and so $P(\theta) = m^{-n}$. Our prior is then

$$P(\theta|\eta) = \frac{\prod_{j=1}^{m} n_j!}{n!} \frac{n!}{\prod_{j=1}^{m} n_j!} \prod_{j=1}^{m} \alpha_j^{n_j} = \prod_{j=1}^{m} \alpha_j^{n_j}.$$

Even though it is straightforward to define the centroid estimator once the marginal posterior distributions for each component are known, even for the simple setup we have described it is hard to obtain the joint posterior, and hence the marginal posteriors, given the combinatorial nature of our problem. Given this limitation, we can still estimate the marginal posteriors through samples from a Markov chain Monte Carlo (MCMC) method. It is important to note the difference in computational complexity between finding the centroid, which is linear on n, and finding the marginal posterior distributions, which is exponential on n.

In the next sections we explore two examples that adopt this approach: clustering and variable selection. The centroid estimator has already found application in these models: Casella and Moreno [**CM06**] treat variable selection in normal regression models through the use of intrinsic priors and select the model with highest posterior probability. Smith and Fahrmeir [**SF07**] consider model selection in functional magnetic resonance imaging analysis with indicator variables for inclusion of regressors defined on a lattice, and use marginal probabilities for variable selection. Tadesse *et al.* [**MGTV05**] formulate a clustering problem using a multivariate normal mixture model. Observations are allocated to classes according to the mode of a marginal posterior distribution, and variables are selected if their marginal posterior probability of above a threshold a; since their a parameter is equivalent to $1/(1+\gamma)$, γ being the sensitivity-specificity ratio described in Section 2.1, their estimator is a γ-centroid.

1.1. Clustering. By clustering we mean the discrimination of n data points X, each with dimension d, into m groups. Adopting the prior distribution on Θ as discussed above we have $P(\theta|\alpha) = \prod_{j=1}^{m} \alpha_j^{n_j}$. We can now choose to specify the group parameters through a hierarchical model: $\alpha \sim \text{Dir}(\xi)$. The resulting prior is then

$$P(\theta) = \int_{[0,1]^m} P(\theta|\alpha)P(\alpha)d\alpha = \frac{\Gamma(\sum_{j=1}^{m}\xi_j)}{\prod_{j=1}^{m}\Gamma(\xi_j)} \frac{\prod_{j=1}^{m}\Gamma(n_j+\xi_j)}{\Gamma(n+\sum_{j=1}^{m}\xi_j)} \propto \prod_{j=1}^{m}\Gamma(n_j+\xi_j).$$

For simplicity, let us assume that X_i is multivariate Gaussian and i.i.d. conditional on the assignment to group j and the parameters that describe the distribution for group j:

$$X_i|\theta_i = j, \beta_j, \Sigma_j \overset{\text{iid}}{\sim} N(\beta_j, \Sigma_j), \quad i = 1, \ldots, n,$$

where β_j and Σ_j have conjugate priors,

$$\begin{aligned} \beta_j|\Sigma_j &\overset{\text{iid}}{\sim} N(\mu_j, \Sigma_j/\kappa_j) \\ \Sigma_j &\overset{\text{iid}}{\sim} \text{Inv-Wi}(\nu_j, \Lambda_j^{-1}) \end{aligned}$$

with hyperparameters μ_j, κ_j, ν_j, and Λ_j, for $j = 1, \ldots, m$. After integrating out the nuisance parameters β and Σ we are left with the following likelihood:

$$P(X|\theta) \propto \prod_{j=1}^{m}(1+n_j/\kappa_j)^{-d/2} \prod_{i=1}^{d}\Gamma\Big(\frac{n_j+\nu_j+1-i}{2}\Big)|\Lambda_j+S_j|^{-(n_j+\nu_j)/2}$$

where

$$\begin{aligned} S_j &= V_j + M_j \\ V_j &= \sum_{\{i:\theta_i=j\}} (X_i-\bar{X}_j)(X_i-\bar{X}_j)^T \\ M_j &= \frac{n_j\kappa_j}{n_j+\kappa_j}(\bar{X}_j-\mu_j)(\bar{X}_j-\mu_j)^T \end{aligned}$$

and so S_j is the sample variance and $\bar{X}_j = \sum_{\{i:\theta_i=j\}} X_i/n_j$ is the mean for group j. We note that n_j, $\bar{X}_j$, and S_j are the sufficient statistics.

To define the centroid estimator we need the marginals $P(\theta_i|X)$ or, since we are only concerned with the maximizer, the marginal joint $P(\theta_i, X) \propto P(\theta_i|X)$. However, to compute $P(\theta_i, X)$ is a daunting endeavour since it comprises exploring an exponential number of realizations of θ over all possible components other than i. Another possible approach is to draw samples $\theta^{(s)}$, $s = 1, \ldots, N$, from the posterior $P(\theta|X)$ using a MCMC method and find the centroid from the estimated marginals

$$\hat{P}(\theta_i = j|X) = \frac{\sum_{s=1}^{N} I(\theta_i^{(s)} = j)}{N}.$$

Let us denote $\theta_{[i]} = \theta[\{1, \ldots, n\} \setminus \{i\}]$. We observe that

$$P(\theta_i = k|\theta_{[i]}, X) = \frac{P(\theta_i = k, \theta_{[i]}, X)}{\sum_{l=1}^{m} P(\theta_i = l, \theta_{[i]}, X)}$$

and so an efficient Gibbs sampling scheme [**GG84**] can be implemented as follows. Let i be the current component to be sampled according to $P(\theta_i|\theta_{[i]}, X)$ and $\theta_i = k$ at the current iteration; store for each group j, n_j, $\bar{X}_j$, V_j, and S_j, and then sample θ_i according to weights $w_l = P(\theta_i = l, \theta_{[i]}, X)/P(\theta_i = k, \theta_{[i]}, X)$: $w_k = 1$ and

$$\begin{aligned} w_l &= \Big(1 - \frac{1}{n_k + \kappa_k}\Big)^{-d/2} \Big(1 + \frac{1}{n_l + \kappa_l}\Big)^{-d/2} \frac{\psi(n_l + \nu_l + 1)}{\psi(n_k + \nu_k)} \frac{n_l + \xi_l + 1}{n_k + \xi_k} \\ &\quad \frac{|\Lambda_k + S_k'|^{-(n_k+\nu_k-1)/2}}{|\Lambda_k + S_k|^{-(n_k+\nu_k)/2}} \frac{|\Lambda_l + S_l'|^{-(n_k+\nu_k+1)/2}}{|\Lambda_l + S_l|^{-(n_l+\nu_l)/2}} \end{aligned}$$

for $l \neq k$, where $\psi(n) = \prod_{i=1}^{d} \Gamma((n+1-i)/2)/\Gamma((n+1-i)/2-1/2) = \Gamma(n/2)/\Gamma((n-d)/2)$, and S'_k and S'_l are the updated variances of groups k and l:

$$\begin{aligned}
n'_k &= n_k - 1 \\
\bar{X}'_k &= (n_k \bar{X}_k - X_i)/n'_k \\
V'_k &= V_k - n'_k/n_k(\bar{X}'_k - X_i)(\bar{X}'_k - X_i)^T \\
M'_k &= \frac{n'_k \kappa_k}{n'_k + \kappa_k}(\bar{X}'_k - \mu_k)(\bar{X}'_k - \mu_k)^T \\
S'_k &= V'_k + M'_k \\
n'_l &= n_l + 1 \\
\bar{X}'_l &= (n_l \bar{X}_l + X_i)/n'_l \\
V'_l &= V_l + n_l/n'_l(\bar{X}_l - X_i)(\bar{X}_l - X_i)^T \\
M'_l &= \frac{n'_l \kappa_l}{n'_l + \kappa_l}(\bar{X}'_l - \mu_l)(\bar{X}'_l - \mu_l)^T \\
S'_l &= V'_l + M'_l
\end{aligned}$$

After sampling $\theta_i = l$, if $l \neq k$ we update $n_l = n'_l$, $\bar{X}_l = \bar{X}'_l$, $V_l = V'_l$, $S_l = S'_l$, and the sufficient statistics for group k accordingly and proceed to the next iteration cycling over components.

1.2. Variable selection. Let our data X be a d-dimensional response vector in a linear regression model with n-by-d design matrix D and regressors β, $X|\beta, \Sigma \sim N(D\beta, \Sigma)$. In some cases, especially when $n \gg d$, we wish to select a subset of explanatory variables in β that best explains the outcome X. One possible approach is to use a latent binary vector θ where θ_i defines whether β_i should be included or not in the regression. We adopt the following model to accommodate such specification: if β_i is selected, $\theta_i = 1$, and we set $\beta_i|\sigma_i^2 \sim N(\mu_i, \sigma_i^2)$; otherwise, $\theta_i = 0$, and we set $\beta_i|\sigma_i^2 \sim N(0, \sigma_i^2/\kappa_i)$, where κ_i controls how strongly we shrink β_i to zero. This specification is adapted from a model known as "spike-and-slab" [**IR05**]. Putting the

two conditions together, we have

$$\beta_i|\sigma_i^2, \theta_i \overset{\text{iid}}{\sim} N(\theta_i\mu_i, \theta_i\sigma_i^2 + (1-\theta_i)\sigma_i^2/\kappa_i).$$

An equivalent, but conciser, formulation is $\beta|S, \theta \sim N(\hat{\mu}, S)$, where $\hat{\mu} = \theta * \mu$, the Hadamard product between θ and μ, and $S = \text{Diag}(\theta_i\sigma_i^2 + (1-\theta_i)\sigma_i^2/\kappa_i)$. For now, we delay the definition of Σ and S, only noting that they could be random.

Integrating β yields the likelihood

$$\begin{aligned} P(X|\theta, \Sigma, S) \quad \propto \quad & |\Sigma|^{-1/2}|S|^{-1/2}|D^T\Sigma^{-1}D + S^{-1}|^{-1/2} \\ & \exp\left\{ -\frac{1}{2}(X - D\hat{\mu})^T(\Sigma + DSD^T)^{-1}(X - D\hat{\mu}) \right\}. \end{aligned}$$

In order to eliminate the variance parameters, we assume a simplified model with common nuisance σ^2, $\Sigma = \sigma^2 I$ and $S = \sigma^2\text{Diag}((1-\theta_i)/\kappa_i + \theta_i)$, and set a conjugate prior $\sigma^2 \sim \text{Inv-}\chi^2(\nu, \tau^2)$. The marginal likelihood is then

$$\begin{aligned} P(X|\theta) \quad \propto \quad & \prod_{i=1}^{n}[(1-\theta_i)/\kappa_i + \theta_i]^{-1/2}|D^TD + \text{Diag}((1-\theta_i)\kappa_i + \theta_i|^{-1/2} \\ & [\nu\tau^2 + (X - D\hat{\mu})^T(I + D\,\text{Diag}((1-\theta_i)/\kappa_i + \theta_i)D^T)^{-1}(X - D\hat{\mu})]^{(\nu+n)/2}, \end{aligned}$$

while the prior on θ can be specified analogously to the previous section: $P(\theta) \propto \Gamma(\xi_0 + n - \sum_{i=1}^n \theta_i)\Gamma(\xi_1 + \sum_{i=1}^n \theta_i)$.

For the same reasons discussed in the previous section, we should also resort here to MCMC sampling to obtain $P(\theta_i|X)$ and hence the centroid estimate. Moreover, even for this simple model we can see that it is more challenging to devise a Gibbs sampling scheme as before since the functions of sufficient statistics are much more complex.

2. Change point models

A *change point model* is a transition model on states $S = \{0, \ldots, k\}$ and only two types of transitions: self-transitions (i, i), $i \in S$, and forward transitions $(i, i+1)$, $i \in S \setminus \{k\}$. Figure 3.1 pictures the transition graph of a change point model.

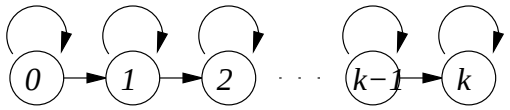

FIGURE 3.1. Transition graph of a change point model with k change points.

We then say that component j is a change point iff $\theta_{j-1} \neq \theta_j$. If we restrict all elements in Θ to have exactly k change points, then we can represent any element more economically as $\theta = \{c_1, \ldots, c_k\}$, where $\theta_{c_i - 1} = i - 1$ and $\theta_{c_i} = i$ for $i = 1, \ldots, k$. We note that the change point positions define subsequences $c_i : c_{i+1} - 1 \doteq \{c_i, \ldots, c_{i+1} - 1\}$, for $i = 0, \ldots, k$, where we make the convention that $c_0 = 1$ and $c_{k+1} = n + 1$.

To define the likelihood we assume that, conditional on θ, subsequences $X[c_i : c_{i+1} - 1]$ and $X[c_j : c_{j+1} - 1]$ are independent for $i \neq j$. In addition, for each state i we define a hyper-parameter set η_i that now fully specifies the likelihood:

$$P(X|\theta, \eta) = \prod_{i=0}^{k} P(X[c_i : c_{i+1} - 1]|\eta_i).$$

One of the important assumptions for this model is that the prior distribution should not depend on the change points, which yields the flat $P(\theta) = \binom{n-1}{k}^{-1}$. The number of change points can later be randomized and determined by regarding this decision as a model selection problem. In this case, the prior on Θ assumes the familiar form $P(\theta) = P(\theta|k)P(k) = \binom{n-1}{k}^{-1} P(k)$. The most common choices for the distribution on k are: $P(k) = 1/(k_{\max}+1)$ for $k = 0, \ldots, k_{\max}$ and $P(k) = 0$ otherwise; $k \sim \text{Bin}(n-1, p)$, where p can be interpreted as a fragmentation parameter, the proportion of change points on θ; and $P(k) \propto g_k e^{-E(k)}$, a Boltzmann distribution with degeneracies $g_k = \binom{n-1}{k}$ and energy function E. The selection of the number of change points is then based on the full posterior

$$P(k|X) = \frac{\sum_{\theta \in \Theta} P(X|\theta, k) P(\theta|k) P(k)}{\sum_{k'=0}^{n-1} \sum_{\theta \in \Theta} P(X|\theta, k') P(\theta|k') P(k')}$$

by, for example, taking the argument maximizer.

Now, to find the centroid estimator we need to consider the change point representation of θ and how it translates the Hamming loss to a new loss function. Figure 3.2

shows a schematic representation of Hamming loss between θ and $\tilde{\theta}$ around the i-th subsequence of θ. To account for the loss in this subsequence we define a function $h_i(\theta, \tilde{\theta})$ and distinguish four cases: (i) $c_{i-1} < \tilde{c}_i < c_i$, in which case $h_i(\theta, \tilde{\theta}) = c_i - \tilde{c}_i$; (ii) $\tilde{c}_i < c_{i-1} < c_i$, $h_i(\theta, \tilde{\theta}) = c_i - c_{i-1}$, and so $h_i(\theta, \tilde{\theta}) = c_i - \max\{\tilde{c}_i, c_{i-1}\}$ when $\tilde{c}_i < c_i$; (iii) $c_i < \tilde{c}_i < c_{i+1}$, $h_i(\theta, \tilde{\theta}) = \tilde{c}_i - c_i$; and (iv) $c_i < c_{i+1} < \tilde{c}_i$, $h_i(\theta, \tilde{\theta}) = c_{i+1} - c_i$, and so $h_i(\theta, \tilde{\theta}) = \min\{\tilde{c}_i, c_{i+1}\} - c_i$ when $\tilde{c}_i > c_i$.

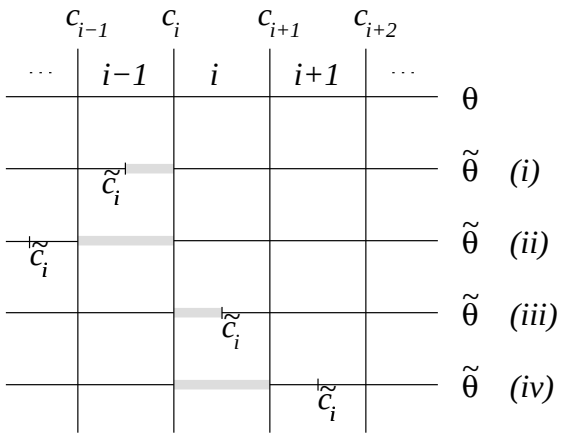

FIGURE 3.2. Representation of $h_i(\theta, \tilde{\theta})$. The highlighted gray parts in $\tilde{\theta}$ mark the partial Hamming loss between θ and $\tilde{\theta}$ due to the i-th subsequence, $h_i(\theta, \tilde{\theta})$.

The following lemma gives us Hamming loss under change point representation.

LEMMA 9. *The Hamming distance between* $\theta = \{c_i\}_{i=1}^k$ *and* $\tilde{\theta} = \{\tilde{c}_i\}_{i=1}^k$ *is* $H(\theta, \tilde{\theta}) = \sum_{i=1}^k h_i(\theta, \tilde{\theta})$*, where*

$$h_i(\theta, \tilde{\theta}) = I(\tilde{c}_i < c_i)(c_i - \max\{\tilde{c}_i, c_{i-1}\}) + I(\tilde{c}_i > c_i)(\min\{\tilde{c}_i, c_{i+1}\} - c_i).$$

PROOF. We use induction on the number of change points k. The result clearly holds for $k = 1$ since $H(\theta, \tilde{\theta}) = |c_1 - \tilde{c}_1|$. Now, by the induction hypothesis,

$$\begin{aligned} \sum_{i=1}^{k} h_i(\theta, \tilde{\theta}) &= H(\theta[1 : c_{k+1}], \tilde{\theta}[1 : c_{k+1}]) + I(\tilde{c}_k > c_{k+1})(c_{k+1} - c_k) \\ &= H(\theta[1 : c_k], \tilde{\theta}[1 : c_k]) + I(\tilde{c}_k > c_k)(\min\{\tilde{c}_k, c_{k+1}\} - c_k). \end{aligned}$$

Thus,

$$\begin{aligned}\sum_{i=1}^{k+1} h_i(\theta,\tilde{\theta}) &= H(\theta[1:c_k],\tilde{\theta}[1:c_k]) + I(\tilde{c}_k > c_k)(\min\{\tilde{c}_k, c_{k+1}\} - c_k) \\ &\quad + I(\tilde{c}_{k+1} < c_{k+1})(c_{k+1} - \max\{\tilde{c}_{k+1}, c_k\}) \\ &\quad + I(\tilde{c}_{k+1} > c_{k+1})(\min\{c_{k+2}, \tilde{c}_{k+1}\} - c_{k+1}) \\ &= H(\theta[1:c_{k+1}],\tilde{\theta}[1:c_{k+1}]) + I(\tilde{c}_{k+1} > c_{k+1})(\min\{\tilde{c}_{k+1}, c_{k+2}\} - c_{k+1})\end{aligned}$$

since the terms $I(\tilde{c}_k > c_k)(\min\{\tilde{c}_k, c_{k+1}\} - c_k)$ and $I(\tilde{c}_{k+1} < c_{k+1})(c_{k+1} - \max\{\tilde{c}_{k+1}, c_k\})$ account for the loss in the subsequence from c_k to c_{k+1}. This last expansion can be more easily seen with the aid of Figure 3.3 where the indicator summands are illustrated.

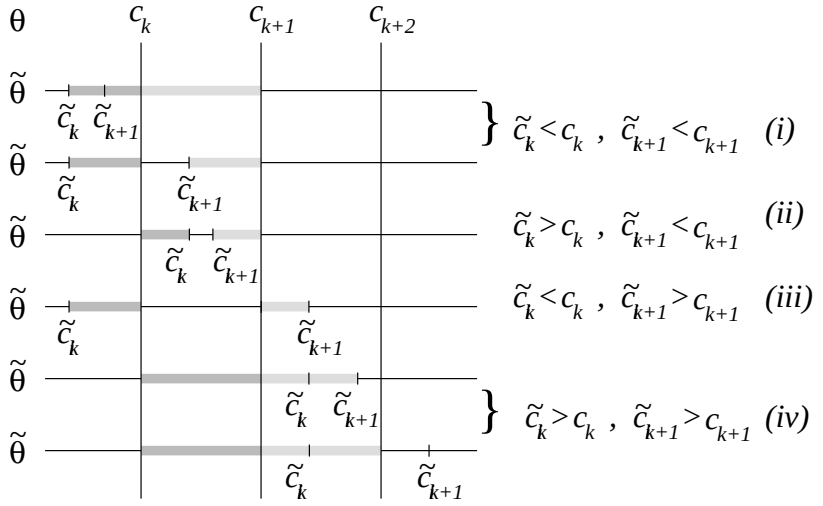

FIGURE 3.3. Four cases for extending the Hamming loss from the k-th change point to the $(k+1)$-th change point. Dark and light gray regions indicate partial Hamming loss due to the k-th subsequence and $(k+1)$-th subsequence respectively. Cases (i) and (ii), where $\tilde{c}_{k+1} < c_{k+1}$, account for $c_{k+1} - \max\{\tilde{c}_{k+1}, c_k\}$ loss, while cases (iii) and (iv), where $\tilde{c}_{k+1} > c_{k+1}$, account for $\min\{c_{k+2}, \tilde{c}_{k+1}\} - c_{k+1}$ loss.

Finally, when $c_{k+1} = n+1$, $\sum_{i=1}^{k} h_i(\theta,\tilde{\theta}) = H(\theta,\tilde{\theta})$. □

We are now ready to present the centroid estimator for change point models.

THEOREM 10 (Centroid estimator for change point models). *The centroid estimator for change point models is given by* $\hat{\theta}_C = \arg\min_{\tilde{\theta}\in\Theta} \sum_{i=1}^{k} A_i(\tilde{c}_i)$, *where*

$$\begin{aligned} A_i(p) &= \sum_{j=i+1}^{n-k+i} I(p<j)\Big[\sum_{l=i}^{j-1}(j-\max\{p,l\})P(c_i=j,c_{i-1}=l|X)\Big] \\ &\quad + I(p>j)\Big[\sum_{l=j+1}^{n-k+i+1}(\min\{p,l\}-j)P(c_i=j,c_{i+1}=l|X)\Big]. \end{aligned}$$

PROOF. From the definition of centroid estimator and the previous lemma we have

$$\hat{\theta}_C = \arg\min_{\tilde{\theta}\in\Theta} E_{\theta|X} H(\theta,\tilde{\theta}) = \arg\min_{\tilde{\theta}\in\Theta} \sum_{i=1}^{k}\sum_{\theta\in\Theta} h_i(\theta,\tilde{\theta})P(\theta|X) \doteq \arg\min_{\tilde{\theta}\in\Theta} \sum_{i=1}^{k} g_i(\tilde{\theta}).$$

But then

$$\begin{aligned} g_i(\tilde{\theta}) &= \sum_{c_1=2}^{n-k+1}\cdots\sum_{c_k=c_{k-1}+1}^{n} h_i(\theta,\tilde{\theta})P(\theta|X) \\ &= \sum_{c_{i-1}=i}^{n-k+i-1}\sum_{c_i=c_{i-1}+1}^{n-k+i}\sum_{c_{i+1}=c_i+1}^{n-k+i+1}\Big[I(\tilde{c}_i<c_i)(c_i-\max\{\tilde{c}_i,c_{i-1}\}) \\ &\quad + I(\tilde{c}_i>c_i)(\min\{\tilde{c}_i,c_{i+1}\}-c_i)\Big]P(c_{i-1},c_i,c_{i+1}|X) \\ &= \sum_{j=i+1}^{n-k+i} I(\tilde{c}_i<j)\Big[\sum_{l=i}^{j-1}(j-\max\{\tilde{c}_i,l\})P(c_i=j,c_{i-1}=l|X)\Big] \\ &\quad + I(\tilde{c}_i>j)\Big[\sum_{l=j+1}^{n-k+i+1}(\min\{\tilde{c}_i,l\}-j)P(c_i=j,c_{i+1}=l|X)\Big] \\ &= A_i(\tilde{c}_i). \end{aligned}$$

□

To compute $A_i(p)$, for $i=1,\ldots,k$ and $p=i+1,\ldots,n-k+i$, it is sufficient to have, for $i=0,\ldots,k$, $P(c_i,c_{i+1}|X)$ or, equivalently, $a_i(j,l) \doteq \sum_{\{\theta:c_i=j,c_{i+1}=l\}} P(X|\theta)$ since

$$P(c_i=j,c_{i+1}=l|X) = \sum_{\{\theta:c_i=j,c_{i+1}=l\}} P(X|\theta)P(\theta)/P(X) \propto a_i(j,l)$$

and the argument minimizer for the centroid is the same if we substitute $a_i(j,l)$ for $P(c_i = j, c_{i+1} = l|X)$ in $A_i(p)$. Our sufficient statistics $a_i(j,l)$ can be obtained using *forward* and *backward* recursive partial sums

$$\begin{aligned} f_i(j) &= \sum_{c_1} \cdots \sum_{c_{i-1}} \prod_{p=0}^{i-2} P(X[c_p : c_{p+1} - 1]|\eta_p) P(X[c_{i-1} : j - 1]|\eta_{i-1}) \\ &= \sum_{c_{i-1}} \Big[\sum_{c_1} \cdots \sum_{c_{i-2}} \prod_{p=0}^{i-3} P(X[c_p : c_{p+1} - 1]|\eta_p) P(X[c_{i-2} : c_{i-1} - 1|\eta_{i-2}) \Big] \\ &\quad \cdot P(X[c_{i-1} : j - 1]|\eta_{i-1}) \\ &= \sum_{c_{i-1}} f_{i-1}(c_{i-1}) P(X[c_{i-1} : j - 1]|\eta_{i-1}) \end{aligned}$$

with $f_0(1) \doteq 1$ and

$$\begin{aligned} b_i(j) &= \sum_{c_{i+1}} \cdots \sum_{c_k} P(X[j : c_{i+1} - 1]|\eta_i) \prod_{p=i+1}^{k} P(X[c_p : c_{p+1} - 1]|\eta_p) \\ &= \sum_{c_{i+1}} P(X[j : c_{i+1} - 1]|\eta_i) \\ &\quad \cdot \Big[\sum_{c_{i+2}} \cdots \sum_{c_k} P(X[c_{i+1} : c_{i+2} - 1|\eta_{i+1}) \prod_{p=i+2}^{k} P(X[c_p : c_{p+1} - 1]|\eta_p) \Big] \\ &= \sum_{c_{i+1}} P(X[j : c_{i+1} - 1]|\eta_i) b_{i+1}(c_{i+1}) \end{aligned}$$

with $b_{k+1}(n+1) \doteq 1$, since

$$\begin{aligned} a_i(j,l) &= \sum_{c_1} \cdots \sum_{c_{i-1}} \sum_{c_{i+2}} \cdots \sum_{c_k} \prod_{p=0}^{i-2} P(X[c_p : c_{p+1} - 1]|\eta_p) P(X[c_{i-1} : j-1]|\eta_{i-1}) \\ &\quad \cdot P(X[j : l-1]|\eta_i) P(X[l : c_{i+2} - 1]|\eta_{i+1}) \prod_{p=i+2}^{k} P(X[c_p : c_{p+1} - 1]|\eta_p) \\ &= \Big[\sum_{c_1} \cdots \sum_{c_{i-1}} \prod_{p=0}^{i-2} P(X[c_p : c_{p+1} - 1]|\eta_p) P(X[c_{i-1} : j-1]|\eta_{i-1}) \Big] \\ &\quad \cdot P(X[c_{i-1} : j-1]|\eta_{i-1}) \\ &\quad \cdot \Big[\sum_{c_{i+2}} \cdots \sum_{c_k} P(X[l : c_{i+2} - 1]|\eta_{i+1}) \prod_{p=i+2}^{k} P(X[c_p : c_{p+1} - 1]|\eta_p) \Big] \\ &= f_i(j) P(X[j : l-1]|\eta_i) b_{i+1}(l). \end{aligned}$$

Note that even if in the sum we intend to minimize the i-th term depends only on $\tilde{c}_i$, we still need to keep the feasibility of $\tilde{\theta}$ by guaranteeing $\tilde{c}_i < \tilde{c}_{i+1}$. This local dependency on consecutive components of the centroid estimator allows us to explore a dynamic programming scheme similar to the approach we followed in the previous chapter for transition models and derive Algorithm 3.1.

The proof of the correctness of Algorithm 3.1 is omitted here for being a straightforward application of the principle of optimality [**Bel57**] and very similar to the proofs of previous centroid algorithms in the last chapter.

In the next sections we present some examples of change point models and centroid estimation in this context.

2.1. Discrete composition. Let A be an alphabet with letter set $\{a_l\}_{l=1}^{d}$. In general, for a transition model with transition set S, we define the likelihood of an assignment θ of each position of X to a state in S, a (discrete) composition, by

$$P(X|\theta) = \prod_{i=1}^{n} P(X_i|\theta_i = j) = \prod_{i=1}^{n} \prod_{l=1}^{d} \alpha_{jl}^{I(X_i = a_l)}$$

Input: $A_i(p)$, $i = 1, \ldots, k$, $p = i+1, \ldots, n-k+i$.
Output: Centroid estimate $\hat{\theta}_C = \{\hat{c}_i\}_{i=1}^k$.
`% initialization`
for $p = 1, \ldots, n-k$ **do**
  $C_0(p) \leftarrow 0$;
end
`% recursion`
for $i = 1, \ldots, k$ **do**
  for $p = i+1, \ldots, n-k+i$ **do**
    $C_i(p) \leftarrow A_i(p) + \min_{l=j,\ldots,p-1} C_{i-1}(l)$;
    $b_i(p) \leftarrow \arg\min_{l=j,\ldots,p-1} C_{i-1}(l)$;
  end
end
`% termination`
$C_{k+1} \leftarrow \min_{l=k+1,\ldots,n} C_k(l)$; `%` $\min_{\tilde{\theta}\in\Theta} \sum_{i=1}^k A_i(\tilde{c}_i)$
$b_{k+1} \leftarrow \arg\min_{l=k+1,\ldots,n} C_k(l)$;
`% backtrack to recover` $\hat{\theta}_C$
$\hat{c}_k \leftarrow b_{k+1}$;
for $i = k-1, \ldots, 1$ **do**
  $\hat{c}_i \leftarrow b_{i+1}(\hat{c}_{i+1})$;
end

ALGORITHM 3.1: Centroid estimator for change point models.

where $\eta_j = (\alpha_{j1}, \ldots, \alpha_{jd})$ is the parameter set for state j with $\sum_{l=1}^d \alpha_{jl} = 1$. As usual in Bayesian statistics, instead of point estimating η_j we randomize and average it out; given the likelihood above, a reasonable distribution is the conjugate Dirichlet $\eta_j \sim \mathrm{Dir}(\xi_j)$ with hyperparameters ξ_j. The likelihood then becomes

$$P(X|\theta, \eta) = \prod_{j\in S} \prod_{i:\theta_i=j} \prod_{l=1}^d \alpha_{jl}^{I(X_i=a_l)} = \prod_{j\in S} \prod_{l=1}^d \alpha_{jl}^{n_{jl}}$$

where $n_{jl} = \sum_{i=1}^n I(\theta_i = j) I(X_i = a_l)$. Let us also define $n_j = \sum_{l=1}^d n_{jl}$ and $\bar{\xi_j} = \sum_{l=1}^d \xi_{jl}$. The marginal likelihood is then

$$P(X|\theta) = \prod_{j\in S} \int_{[0,1]^d} \prod_{l=1}^d \alpha_{jl}^{n_{jl}} \frac{\bar{\xi_j}}{\prod_{l=1}^d \Gamma(\xi_{jl})} \prod_{l=1}^d \alpha_{jl}^{\xi_{il}-1} d\eta_j \propto \prod_{j\in S} \frac{\prod_{l=1}^d \Gamma(n_{jl}+\xi_{jl})}{n_j + \bar{\xi_j}}.$$

In particular, for k change point models we have $P(X|\theta) = \prod_{i=0}^{k} P(X[c_i : c_{i+1}-1])$ where

$$P(X[c_i : c_{i+1} - 1]) \propto \frac{\prod_{l=1}^{d} \Gamma(n_{il} + \xi_{il})}{c_{i+1} - c_i + \bar{\xi}_i}$$

can then be used to find the centroid estimator in Algorithm 3.1.

2.2. Gaussian segmentation. In this example let us consider the same setup, a transition model with transition set S, but now assume that, conditional on the assignment θ, we have

$$X_i|\theta_i = j, \eta \doteq (\beta, \sigma^2) \overset{\text{iid}}{\sim} N(D\beta_j, \sigma_j^2 I_m)$$

where X_i has dimension m, D is a m-by-d design matrix with $m > d$, and I_m denotes the identity matrix of order m. Assuming conjugate priors for β and σ^2,

$$\begin{aligned} \beta_j|\sigma_j^2 &\sim N(\mu_j, \sigma_j^2/\kappa_j I_d) \\ \sigma_j^2 &\sim \text{Inv-}\chi^2(\nu_j, \tau_j^2) \end{aligned}$$

for $j \in S$, we obtain

$$P(X|\theta) \propto \prod_{j \in S} |I_d + n_j/\kappa_j D^T D|^{-1/2} \Gamma\Big(\frac{n_j m + \nu_j}{2}\Big) (\nu_j \tau_j^2 + V_j)^{-(n_j m + \nu_j)/2}$$

where

$$V_j = n_j(\bar{X}_j - D\mu_j)^T (I_m + n_j/\kappa_j DD^T)^{-1} (\bar{X}_j - D\mu_j) + \sum_{i:\theta_i = j} (X_i - \bar{X}_j)^T (X_i - \bar{X}_j),$$

$\bar{X}_j = \sum_{i:\theta_i=j} X_i/n_j$, and $n_j = \sum_{i=1}^{n} I(\theta_i = j)$.

As in the previous example, we should have $\{i : \theta_i = j\} = \{c_j, \ldots, c_{j+1} - 1\}$, $n_j = c_{j+1} - c_j$, and so

$$P(X[c_i : c_{i+1} - 1]) \propto |I_d + n_j/\kappa_j D^T D|^{-1/2} \Gamma\Big(\frac{n_j m + \nu_j}{2}\Big) (\nu_j \tau_j^2 + V_j)^{-(n_j m + \nu_j)/2}.$$

3. Hidden Markov models

A particular, albeit ubiquitous, case of graph models arises when we restrict the structure graph to a sequential graph and require the prior probability distribution to satisfy the Markov property in the following definition.

DEFINITION 6 (Hidden Markov model). A *k-th order hidden Markov model (HMM)* is a graph model where the structure graph G is a $(k+1)$-th order sequential graph G_{k+1} and we define sets of initial constraints $\mathcal{I} \doteq \mathcal{C}(\{1,\ldots,k\})$ and transition constraints $\mathcal{C} \doteq \mathcal{C}(V_i)$ for each clique $V_i = \{i,\ldots,i+k\}$, $i = 1,\ldots,n-k$. The probability distribution on Θ is given by

$$P(\theta) = \prod_{j=k}^{n-1} P(\theta_{j+1}|\theta_{j+1-k},\ldots,\theta_j) P(\theta_1,\ldots,\theta_k)$$

for $\theta \in \Theta$ and the likelihood of θ for data X is given by

$$P(X|\theta) = \prod_{j=1}^{n} P(X_j|\theta_j).$$

It is clear from the definition above that we just need to define *initial* probabilities $P(\theta_1,\ldots,\theta_k)$ for $(\theta_1,\ldots,\theta_k) \in \mathcal{I}$ and *transition* probabilities $P(\theta_{j+1}|\theta_{j+1-k},\ldots,\theta_j)$ for $j = k,\ldots,n-1$ and $(\theta_{j+1-k},\ldots,\theta_{j+1}) \in \mathcal{C}$ to define $P(\theta)$. For instance, the graph model becomes a simpler transition model when $k = 1$, hence $\mathcal{I} = I$ and $\mathcal{C} = F$, the initial and transition sets, and the transition probabilities are usually represented as weights in F. Moreover, marginal posterior probabilities are readily available from the forward and backward algorithms [**Rab93**], and the generalized centroid can then be computed using Algorithm 2.1.

Since G is G_{k+1}, a $(k+1)$-th order sequential graph, the graph centroid is a sequential $(k+1)$-th order centroid. In addition, in this particular case, the cliques V_i represent sets of components that are related by a k-th order Markov dependency. Thus, the graph centroid estimator accounts for differences on locally correlated components, that is, the distance considers minimal groups of components that together describe the probability structure of the parameter space. So, as k grows, the higher

the influence of a discrepancy in one component when comparing two points of Θ: since there are k cliques that contain a component i, if two points disagree in i this accounts for a loss of at least k. Also, consider two components i and j; if two points differ in both i and j then their distance is at least $k + \min(k, |i - j|)$. If i and j are close relative to k ($|i - j| < k$) then we incur in a smaller loss, lesser than $2k$, than if they are far apart. This behavior is justified in light of the Markov property since close components are more strongly correlated, should report a similar likelihood, and thus we incur in smaller penalties for point differences in these components.

Another advantage to k-th order HMM is that the marginal joint posteriors $P_i(\theta[i : i + k]) \doteq P(\theta_i, \ldots, \theta_{i+k}|X)$ can be computed efficiently with a generalized version of the forward and backward algorithms. With the marginal joints we can now find our graph centroid. For convenience, we provide a version of Algorithm 2.3 that is specific for our current model in Algorithm 3.2. We adopt the following transition set notation: $\mathcal{F} = \{(T, U) \in \mathcal{C}^2 : U_j = T_{j+1}, j = 1, \ldots, k\}$.

COROLLARY 11. *Algorithm 3.2 computes the graph centroid estimator for k-th order hidden Markov models.*

PROOF. The correctness follows directly from Theorem 5 by using the tree decomposition of Section 5.2 and setting the post-order as the identity by rooting T on $r \doteq n - k$. □

To illustrate some applications we revisit the examples from the previous section under a first-order HMM. Since in this simple case we have a transition model, the likelihood for both examples was already discussed. Now we just need to define the prior distribution on Θ, which accounts for initial probabilities $\pi_j = P(\theta_1 = j)$ and transition probabilities $a_{ij} = P(\theta_{l+1} = j|\theta_l = i)$ for $i, j \in S$, the transition set. As in the previous section on change point models, we assume π and a_i are nuisance vectors and set a hierarchical conjugate distribution on them: $\pi \sim \text{Dir}(\xi_0)$ and $a_i \sim \text{Dir}(\xi_i)$.

Input: Marginal posterior probabilities $P_i(U)$ for $i = 1, \ldots, n-k$.
Output: Graph centroid estimate $\hat{\theta}_G$ for k-th order HMM.
% initialization
for $U \in \mathcal{C}$ **do**
  if $U[1:k] \in \mathcal{I}$ **then** $C_0(U) \leftarrow 0$; **else** $C_0(U) \leftarrow -\infty$;
end
% recursion
for $i = 1, \ldots, n-k$ **do**
  for $U \in \mathcal{C}$ **do**
    $C_i(U) \leftarrow P_i(U) + \max_{T \in \mathcal{C}:(T,U) \in \mathcal{F}} C_{i-1}(T)$;
    $B_i(U) \leftarrow \arg\max_{T \in \mathcal{C}:(T,U) \in \mathcal{F}} C_{i-1}(T)$;
  end
end
% termination
$C_{n-k+1} \leftarrow \max_{U \in \mathcal{C}} C_{n-k}(U)$; % $\max_{\tilde{\theta} \in \Theta} \sum_{i=1}^{n-k} P_i(\tilde{\theta}[i:i+k])$
$B_{n-k+1} \leftarrow \arg\max_{U \in \mathcal{C}} C_{n-k}(U)$;
% backtrack to recover $\hat{\theta}_G$
$\hat{\theta}_G[n-k:n] \leftarrow B_{n-k+1}$;
for $i = n-k-1, \ldots, 1$ **do**
  $\hat{\theta}_G[i:i+k] \leftarrow B_{i+1}(\hat{\theta}_G[i+1:i+k+1])$;
end

ALGORITHM 3.2: Graph centroid estimator for k-th order hidden Markov models.

4. Conditional independence models

In the previous chapter, when we presented graph models we abstracted from specifying a prior probability distribution on the parameter space Θ. The following definition helps us define a distribution on Θ and is closely related to the concept of structure graph.

DEFINITION 7 (Conditional independence graph). The *(conditional) independence graph* of θ is $G(V = \{1, \ldots, n\}, E)$, where

$$(i,j) \notin E \Leftrightarrow \theta_i \perp\!\!\!\perp \theta_j | \{\theta_k : k \in V \setminus \{i,j\}\}$$

that is, vertices i and j are not adjacent in G if and only if θ_i and θ_j are conditionally independent given the remaining components in θ.

When the structure graph coincides with the independence graph we have a *graphical model*. Since $P(\theta) > 0$ for all θ that satisfy the structure graph, the pairwise Markov property of the independence graph is equivalent to the following Gibbs property by the Hammersley-Clifford Theorem [**HC68**]: if $\mathcal{K}(G)$ is the collection of cliques in G and ψ_C is a (clique) potential over $C \in \mathcal{K}(G)$, then

$$P(\theta) \propto \prod_{C \in \mathcal{K}(G)} \psi_C(\theta[C]).$$

Given a tree decomposition $(T, \{V_i\}_{i \in T})$ of G, let $v(G) = |\{i \in T : C \subset V_i\}|$, the number of parts that include $C \in \mathcal{K}(G)$. We can now define potentials

$$\tilde{\psi}_{V_i} = \prod_{C \in \mathcal{K}(G) \cap V_i} \psi_C(\theta[C])^{1/v(C)}$$

and so express the distribution on Θ by

$$P(\theta) \propto \prod_{i \in T} \tilde{\psi}_{V_i}(\theta[V_i]),$$

since each clique in G is contained in at least a part of T. In addition, if we assume that $P(X|\theta) = \prod_{i \in T} P(X[V_i]|\theta[V_i])$, then

$$P(\theta|X) \propto \prod_{i \in T} \tilde{\psi}_{V_i}(\theta[V_i]) P(X[V_i]|\theta[V_i]) \doteq \prod_{i \in T} \phi_{V_i}(X, \theta[V_i]).$$

Given this particular form for the posterior, proportional to a product of part potentials, we can use the message-passing junction-tree algorithm to compute $P(\theta[V_i]|X)$ for all $i \in T$ [**LS88**]. Having the part marginals, we can then obtain the graph centroid from Algorithm 2.3. It is, of course, also possible to compute $P(\theta_j|X)$ from any $P(\theta[V_i]|X)$ such that $j \in V_i$ and find the centroid afterwards using Algorithm 2.2.

4.1. A stochastic grammar for binary sequences. Consider the following grammar B: we define terminals $\{0, 1\}$, non-terminals $\{E, O\}$, and production rules that depend on the *height* h of the node in the parse tree:

$$E \to \begin{cases} EE^* \mid OO, & \text{if } h > 1 \\ 0, & \text{if } h = 1 \end{cases} \quad \text{and} \quad O \to \begin{cases} EO^* \mid OE, & \text{if } h > 1 \\ 1, & \text{if } h = 1 \end{cases}.$$

If a node v is a leaf in a parse tree P then $\textsc{Height}_P(v) = 0$ and v corresponds to a terminal production. We call the nodes v where $\textsc{Height}_P(v) > 1$ *internal*. We further require the parse trees for B to be *complete* binary trees, that is, for any internal node v in a parse tree P with height h and left and right children u and w, $\textsc{Height}_P(u) = \textsc{Height}_P(w) = h - 1$. It now follows that any production S from B must have length 2^k, for some non-negative integer k, and the root of any parse tree for S in B has height $k+1$. It is important to note that a parse tree P is completely specified by the binary productions at its leaves. Moreover, the root of P is E (O) if and only if the number of ones at the leaves of P is even (odd). Figure 3.4 shows two examples of parse trees.

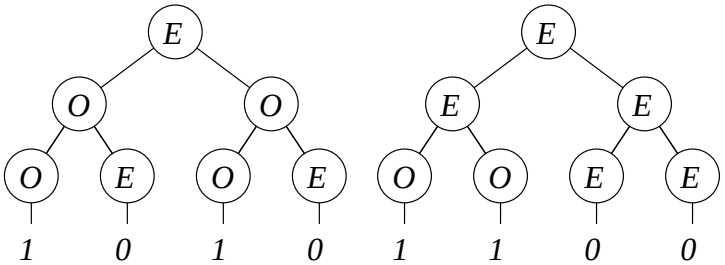

FIGURE 3.4. Unique parse trees for productions (a) 1010 and (b) 1100 from grammar B.

Let θ be a sequence of length n produced by B and P_θ its parse tree. For each node $v \in P_\theta$ we denote the realization of v given θ by $\phi_\theta(v) \in \{E, O\}$. In addition, given a parse tree P, let us denote by v^* the left child of node v in P, by $r(P)$ the root of P, and by $h_P(v)$ the height of v in P. We can now define a (prior) probability distribution on $\Theta = \{0,1\}^n$ induced by B by assigning an initial probability p_S that the root of a parse tree is E and production probabilities p_E and p_O that the first rule (starred in the definition of the rules above, where the left child is always an E) is applied given that the node is E and O respectively. We can then define $P(\theta) = P(P_\theta)$ where

$$P(P_\theta) = \left[p_S^{I(\phi_\theta(r(P_\theta))=E)} (1-p_S)^{I(\phi_\theta(r(P_\theta))=O)} \right] \prod_{v:h_{P_\theta}(v)>1} \left[p_E^{I(\phi_\theta(v^*)=E)} (1-p_E)^{I(\phi_\theta(v^*)=O)} \right]^{I(\phi_\theta(v)=E)} \left[p_O^{I(\phi_\theta(v^*)=E)} (1-p_O)^{I(\phi_\theta(v^*)=O)} \right]^{I(\phi_\theta(v)=O)}.$$

As an example, the parse trees in Figure 3.4 have probabilities (a) $p_S(1-p_E)(1-p_O)^2$ and (b) $p_S p_E^2(1-p_E)$.

It is also worth observing from $P(P_\theta)$ that even though we talk of a parse tree P, the conditional independence graph for the realizations given a production is not actually a tree: the children of an internal node v are adjacent since their realizations are jointly specified by v through the grammar rules. The graph P, however, is not much far from a tree as its largest clique is a triangle. In fact, we can produce a minimum tree-decomposition $\mathcal{T}(P)$ for P as follows: define a tree T with all the internal nodes v from P where each v is connected to its children in P but the children are not adjacent to each other, and for each internal node v define a part V_v containing v and its children. Since $TW(\mathcal{T}(P)) = 2$, $\mathcal{T}(P)$ is minimum and we conclude that $TW(P) = 2$. We call $\mathcal{T}(P)$ the *canonical* tree-decomposition for P.

Data X comprises white-noisy observations from the leaves of a production from B: $X_i|\theta_i \overset{\text{iid}}{\sim} N(\theta_i, \sigma^2)$. Using the message-passing junction-tree algorithm—or the canonical tree-decomposition—we can now compute $P(\theta_i|X)$ and find our centroid estimator. However, since we are ultimately comparing parse trees, it is more reasonable to extend θ to include ϕ_θ, the internal node realizations. To see why this extension better captures the variability on the parameter space, consider the change in two positions i and j of θ: when $i = 1$ and $j = 2$, for example, we change one internal node in P_θ, the parent of i and j, but when $i = 1$ and $j = 3$ we change three internal nodes, namely the parent of i, the parent of j, and the grandparent of i and j. Thus, a new loss function defined on Θ for θ and $\tilde{\theta}$ that is induced by the Hamming

loss on (θ, ϕ_θ) and $(\tilde{\theta}, \phi_{\tilde{\theta}})$ is better suited to define our centroid estimator. We refer to the centroid obtained from the usual Hamming loss on θ as the *naïve* centroid.

Figure 3.5 illustrates the conditional independence graphs for the same productions in Figure 3.4. The Hamming distance from graphs (a) and (b) is four, as highlighted by gray discording nodes. The common canonical tree-decomposition is depicted in (c).

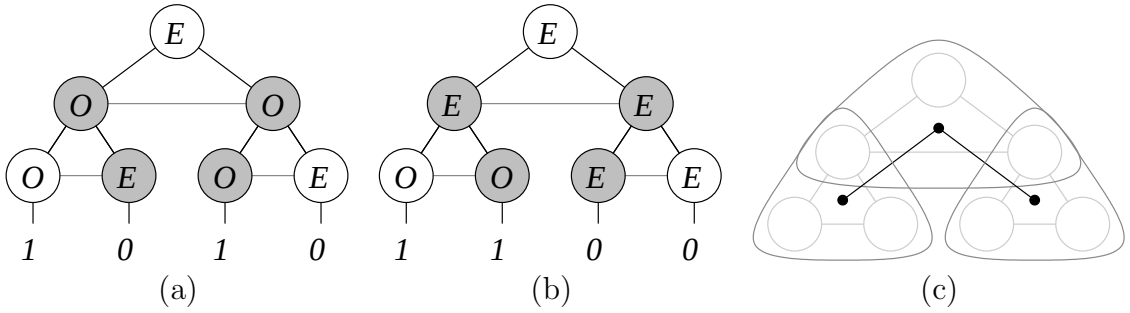

FIGURE 3.5. Conditional independence graphs for productions (a) 1010 and (b) 1100 from grammar B, and (c) their canonical tree-decomposition.

The graph centroid defined by the canonical tree-decomposition has an appealing interpretation: since the parts of the tree-decomposition correspond to production rules, the graph centroid minimizes the expected number of different production rules, as opposed to the expected number of different realizations minimized by the centroid estimator.

Examples. Let us define $p_S = 0.7$, $p_E = p_O = 0.5$, and $\sigma = 0.5$. If we observe $X = (0.81, 0.34, 0.36, -0.56)$, then $\hat{\theta}_M = (1, 0, 1, 0)$ and $\hat{\theta}_C = (1, 1, 0, 0)$. The MAP and centroid estimators are pictured in Figures 3.4 and 3.5 as (a) and (b) respectively. Note that even though X_2 and X_3 are closer to zero, both the MAP and centroid assign one to one of these positions as having an even number of ones in θ is favored by $p_S > 0.5$.

Table 3.1 lists the posterior Hamming distance distribution centered at MAP and centroid estimates. The graph centroid and centroid estimates coincide in this case. The posterior probability of $\hat{\theta}_M$ is higher than $\hat{\theta}_C$ ($D = 0$, highlighted in bold), as expected, but the centroid is placed in a region that concentrates probability mass

more compactly around it, as can be seen by $P(H(\theta, \hat{\theta}_C) = 2|X) > P(H(\theta, \hat{\theta}_M = 2|X)$ ($D = 2$, highlighted in bold). The expected posterior Hamming distance $\bar{D}$ is given in the last line, and the centroid estimate minimizes it, as expected.

TABLE 3.1. Posterior Hamming distance distribution centered at MAP and centroid estimates for $p_S = 0.7$, $p_E = p_O = 0.5$, $\sigma = 0.5$, and $X = (0.81, 0.34, 0.36, -0.56)$.

D	$P(H(\theta, \hat{\theta}_M) = D\|X)$	$P(H(\theta, \hat{\theta}_C) = D\|X)$
0	**0.259**	0.240
2	0.046	**0.133**
3	0.287	0.285
4	0.375	0.308
5	0.032	0.035
$\bar{D}$	2.613	**2.528**

Let us now consider the case when $p_S = 1$ and all productions have an even number of ones, $p_E = p_O = \sigma = 0.5$ as before, and

$$X = (-0.63, 1.59, -0.03, -0.95, 1.09, 1.23, 1.07, 1.28).$$

Our estimates are

$$\begin{aligned} \hat{\theta}_M &= (0,1,1,0,1,1,1,1) \\ \hat{\theta}_C &= (0,1,0,0,1,1,0,1) \\ \hat{\theta}_G &= (0,1,0,0,0,1,1,1) \\ \hat{\theta}_N &= (0,1,0,0,1,1,1,1) \end{aligned}$$

where $\hat{\theta}_N$ is the naïve centroid estimate. Since $\sum_{i=1}^{8}(\hat{\theta}_N)_i = 5$, odd, $P(\hat{\theta}_N|X) = 0$. Table 3.2 lists the posterior Hamming distance centered at each estimate. As before, the centroid estimate centers itself in a region of high posterior probability mass concentration. The graph centroid is similar to the centroid.

4.2. Reconstruction of ancestral states. Let us define a *phylogeny* as a binary tree T with n leaves (and hence $n-1$ internal nodes) and denote the *leaf set* of T by

TABLE 3.2. Posterior Hamming distance distribution centered at MAP, centroid, and graph centroid estimates for $p_S = 1$, $p_E = p_O = 0.5$, $\sigma = 0.5$, and $X = (-0.63, 1.59, -0.03, -0.95, 1.09, 1.23, 1.07, 1.28)$.

D	$P(H(\theta, \hat{\theta}_M) = D\|X)$	$P(H(\theta, \hat{\theta}_C) = D\|X)$	$P(H(\theta, \hat{\theta}_G) = D\|X)$
0	**0.267**	0.226	0.212
2	0.009	0.101	0.122
4	0.058	0.335	0.328
6	0.661	0.333	0.333
8	0.004	0.005	0.005
$\bar{D}$	4.251	**3.581**	3.594

$\mathcal{L}(T)$. Each node in T represents a species, and we partition the nodes in T between observations $X \in \mathcal{L}(T)$ that are extant species at the leaves and the internal nodes θ that are ancestral species. To each edge (u, v) in T we assign a weight t_{uv} that measures the evolutionary time distance between u and v. We also define a (discrete) set of character states, traits D that each node can realize, that is, $X_v, \theta_u \in D$ for $v \in \mathcal{L}(T)$ and $u \in T \setminus \mathcal{L}(T)$. Our task is to reconstruct the ancestral states θ having observed X.

The probabilities of changing states after time t is summarized by a substitution rate matrix Q,

$$Q(t) = \begin{bmatrix} P(d_1|d_1, t) & \cdots & P(d_m|d_1, t) \\ \vdots & \ddots & \vdots \\ P(d_1|d_m, t) & \cdots & P(d_m|d_m, t) \end{bmatrix},$$

or by defining an infinitesimal rate matrix Q_0 and defining $Q(t) = \exp(Q_0 t)$. Note that by the definition of Q we can represent the probability of observing trait d_i after evolutionary time t from d_j as $P(d_i|d_j, t) = \mathbf{e}_j^T Q(t) \mathbf{e}_i$.

Let us assume that the phylogeny is rooted, that is, we know that the root is the common ancestor to all other species. The root r provides a partial order in G, that is, u precedes v, if u is closer to r in T than v, $|uTr| < |vTr|$. Let us define a probability distribution $\pi = [P(\theta_r = d_j)]_{j=1}^m$ for traits in r; if we denote $\text{PARENT}_T(v)$

as v^*, we then have the joint probability

$$P(X,\theta) = P(\theta_r) \prod_{v \in T \setminus \mathcal{L}(T)} P(\theta_v|\theta_{v^*}, t_{v^*v}) \prod_{l \in \mathcal{L}(T)} P(X_l|\theta_{l^*}, t_{l^*l}).$$

We further assume that the phylogeny is known, including the branch lengths, and drop the conditional on t in what follows.

To compute marginal posterior distributions for the traits at species v, let us first define $\mathbf{l}_v \doteq [P(X[T_v]|\theta_v = d_j]_{j=1}^m$, the likelihood of θ_v under the observed states in the subtree rooted at v. If v is an extant species, we set $\mathbf{l}_v = \mathbf{e}_j$, where $X_v = d_j$. The likelihood vector $\mathbf{l}_p$ of species p with left child v and right child w is given by

$$P(X[T_p]|\theta_p) = \sum_{\theta_v, \theta_w} P(X[T_v]|\theta_v) P(\theta_v|\theta_p) P(X[T_w]|\theta_w) P(\theta_w|\theta_p)$$

and so $\mathbf{l}_p = [Q(t_{pv})\mathbf{l}_v] * [Q(t_{pw})\mathbf{l}_w]$.

Now, define $\mathbf{u}_v \doteq [P(X[T \setminus T_v], \theta_v = d_j)]_{j=1}^m$, the partial joint of θ_v and the observed states *not* in the subtree rooted at v. We can apply a recursion similar to the one for $\mathbf{l}_v$: we set $\mathbf{u}_r = \pi$, the prior in the root state, as the base; given parent species p and children v and w we have

$$P(X[T \setminus T_v], \theta_p) = \sum_{\theta_w} P(X[T_w]|\theta_w) P(\theta_w|\theta_p) P(X[T \setminus T_p], \theta_p)$$

and thus $[P(X[T \setminus T_v], \theta_p = d_j)]_{j=1}^m = [Q(t_{pw})\mathbf{l}_w] * \mathbf{u}_p$, from which we get

$$P(X[T \setminus T_v], \theta_v) = \sum_{\theta_p} P(X[T \setminus T_v], \theta_p) P(\theta_v|\theta_p)$$

and so $\mathbf{u}_v = Q(t_{pv})^T([Q(t_{pw})\mathbf{l}_w] * \mathbf{u}_p)$.

We can now compute two important marginal posterior distributions. Let us start with the marginal joint:

$$[P(\theta_v = d_j, X)]_{j=1}^m = [P(X[T_v]|\theta_v = d_j) P(X[T \setminus T_v], \theta_v = d_j]_{j=1}^m = \mathbf{l}_v * \mathbf{u}_v.$$

Since $P(X) = \sum_{\theta_v} P(\theta_v, X) = \langle \mathbf{l}_v, \mathbf{u}_v \rangle$, where $\langle \cdot, \cdot \rangle$ denotes dot product, we obtain

$$\mathbf{p}_v \doteq [P(\theta_v = d_j | X)]_{j=1}^m = \frac{\mathbf{l}_v * \mathbf{u}_v}{\langle \mathbf{l}_v, \mathbf{u}_v \rangle}.$$

The centroid phylogeny for the ancestral states is now simply a consensus estimator on the marginals $\mathbf{p}_v$. Algorithm 3.3 computes $\mathbf{l}_v$, $\mathbf{u}_v$, and $\mathbf{p}_v$ for all v and outputs the centroid. We note that the first part of the algorithm where $\mathbf{l}_v$ is computed is equivalent to Felsenstein's algorithm [**Fel81**].

Input: Phylogeny T, branch lengths $\{t_{uv}\}$, substitution rate matrix Q, prior on root π, observed states X, and gain matrix G.
Output: Centroid phylogeny $\hat{\theta}_C$.
Data: Post-order σ of T from r.
`% compute` $\mathbf{l}_v$
for $i = 1, \ldots, |T|$ **do**
  $p \leftarrow \sigma^{-1}(i)$;
  if $p \in \mathcal{L}(T)$ **then**
    $\mathbf{l}_p \leftarrow \mathbf{e}_j$ where $X_p = d_j$;
  else
    $v \leftarrow$ LEFT$_T(p)$; $w \leftarrow$ RIGHT$_T(p)$;
    $\mathbf{l}_p \leftarrow [Q(t_{pv})\mathbf{l}_v] * [Q(t_{pw})\mathbf{l}_w]$;
  end
end
`% compute` $\mathbf{u}_v$
for $i = |T|, \ldots, 1$ **do**
  $p \leftarrow \sigma^{-1}(i)$;
  if $i = |T|$ **then**
    $\mathbf{u}_p \leftarrow \pi$; `% root`
  else
    $v \leftarrow$ LEFT$_T(p)$; $w \leftarrow$ RIGHT$_T(p)$;
    $\mathbf{u}_v \leftarrow Q(t_{pv})^T([Q(t_{pw})\mathbf{l}_w] * \mathbf{u}_p)$;
    $\mathbf{u}_w \leftarrow Q(t_{pw})^T([Q(t_{pv})\mathbf{l}_v] * \mathbf{u}_p)$;
  end
end
`% compute` $\mathbf{p}_v$ `and` $\hat{\theta}_C$
for $v \in T \setminus \mathcal{L}(T)$ **do**
  $\mathbf{p}_v \leftarrow \dfrac{\mathbf{l}_v * \mathbf{u}_v}{\langle \mathbf{l}_v, \mathbf{u}_v \rangle}$;
  $(\hat{\theta}_C)_v \leftarrow \arg\max_{j=1,\ldots,m} G\mathbf{p}_v$;
end

ALGORITHM 3.3: Centroid phylogeny for reconstruction of ancestral states.

Since the phylogeny is a rooted binary tree, it has unitary tree-width. We can define the canonical (minimum) tree-decomposition $\mathcal{T} = \{T', \{V_e\}_{e \in E(T)}\}$ of a phylogeny T in the following way: for each edge $e = (u, v) \in E(T)$ assign a part $V_e = \{u, v\}$ and a node $v_e \in T'$; an edge $(v_e, v_f) \in T'$ if and only if e and f share a node in T and either $\textsc{Depth}_T(e) = \textsc{Depth}_T(f) = 1$, and so both e and f are incident to the root of T, or $\textsc{Depth}_T(e) \neq \textsc{Depth}_T(f)$. We can see that T' is simply the spanning subgraph of the line graph of T induced by edges of increasing depth from the root of T.

The canonical tree-decomposition can now be used to define the graph centroid phylogeny. Similarly to the previous example, the graph centroid phylogeny minimizes the expected number of different parent-child pair realizations. It is more reasonable as it addresses the minimal correlations between components of θ and should then find a better compromise between the MAP and centroid estimators in representing the posterior distribution in the parameter space of phylogenies.

The graph centroid estimator depends, not surprinsingly, on the posterior pairwise marginals for each parent-child edge in the phylogeny. From one of the previous intermediary results we have

$$\begin{aligned} \mathbf{P}_{pv} \doteq P(\theta_v, \theta_p | X) &= P(X[T \setminus T_v], \theta_p) P(\theta_v | \theta_p) P(X[T_v] | \theta_v) / P(X) \\ &= \left[\Big([Q(t_{pw}) \mathbf{l}_w] * \mathbf{u}_p \Big) \mathbf{l}_v^T \right] * Q(t_{pv}) / \langle \mathbf{l}_v, \mathbf{u}_v \rangle. \end{aligned}$$

Algorithm 3.4 takes $\mathbf{l}_v$ and $\mathbf{u}_v$ as computed from Algorithm 3.3, computes the pairwise marginals $\mathbf{P}_{pv}$ for each parent-child pair p and v, and outputs the graph centroid. This algorithm is a direct adaptation of Algorithm 2.3 tailored for the canonical tree-decomposition we have for rooted binary trees, as in the case of phylogenies.

Example. Jermann *et al.* [**JOSB95**] studied the evolution of the RNase protein family in artiodactyls—the mammal order of even-toed ungulates that includes pig, camel, deer, sheep, and ox—by applying parsimony to reconstruct 13 ancestral ribonuclease sequences and studying their biochemical properties. They find strong evidence that parsimony yields plausible ancient sequences as they observe a significant change

Input: Phylogeny T, branch lengths $\{t_{uv}\}$, substitution rate matrix Q, partial likelihoods $\mathbf{l}$ and joints $\mathbf{u}$ from Algorithm 3.3.
Output: Graph centroid phylogeny $\hat{\theta}_G$.
Data: Post-order σ of T from r.

```
% compute pairwise marginals
```
for $i = 1, \ldots, |T|$ **do**
  $p \leftarrow \sigma^{-1}(i)$;
  if $p \in \mathcal{L}(T)$ **then**
    for $j = 1, \ldots, m$ **do** $C_p(j) \leftarrow 0$;
  else
    $v \leftarrow$ LEFT$_T(p)$; $w \leftarrow$ RIGHT$_T(p)$;
    `% left pairwise marginal`
    $\mathbf{P}_{pv} \leftarrow \left[\Big([Q(t_{pw})\mathbf{l}_w] * \mathbf{u}_p\Big)\mathbf{l}_v^T\right] * Q(t_{pv}) / \langle \mathbf{l}_v, \mathbf{u}_v \rangle$;
    for $j = 1, \ldots, m$ **do**
      $C_p(j) \leftarrow C_p(j) + \max\limits_{k=1,\ldots,m} \Big\{(\mathbf{P}_{pv})_{jk} + C_v(k)\Big\}$;
      $B_v(j) \leftarrow \arg\max\limits_{k=1,\ldots,m} \Big\{(\mathbf{P}_{pv})_{jk} + C_v(k)\Big\}$;
    end
    `% right pairwise marginal`
    $\mathbf{P}_{pw} \leftarrow \left[\Big([Q(t_{pv})\mathbf{l}_v] * \mathbf{u}_p\Big)\mathbf{l}_w^T\right] * Q(t_{pw}) / \langle \mathbf{l}_w, \mathbf{u}_w \rangle$;
    for $j = 1, \ldots, m$ **do**
      $C_p(j) \leftarrow C_p(j) + \max\limits_{k=1,\ldots,m} \Big\{(\mathbf{P}_{pw})_{jk} + C_w(k)\Big\}$;
      $B_w(j) \leftarrow \arg\max\limits_{k=1,\ldots,m} \Big\{(\mathbf{P}_{pw})_{jk} + C_w(k)\Big\}$;
    end
  end
end
`% backtrack to recover` $\hat{\theta}_G$
$(\hat{\theta}_G)_r \leftarrow \arg\max\limits_{k=1,\ldots,m} C_r(k)$;
for $i = |T| - 1, \ldots, 1$ **do**
  $v \leftarrow \sigma^{-1}(i)$;
  $p \leftarrow$ PARENT$_T(v)$;
  $(\hat{\theta}_G)_v \leftarrow B_v\Big((\hat{\theta}_G)_p\Big)$;
end

ALGORITHM 3.4: Graph centroid phylogeny for reconstruction of ancestral states.

in behaviour, namely a fivefold increase in catalytic activity, for the founding ancestor of the artiodactyl lineage. They further hypothesize that this change corresponds to the period when ruminant digestion arose in the artiodactyls, and note that substitution at a single position, namely from glycine (G) to aspartic acid (D) at residue 38, accounts for essentially all of the increased catalytic activity against duplex RNA. Their reconstruction is shown in Figure 3.6; the substitution of interest occurs from ancestral species **h** to **g**.

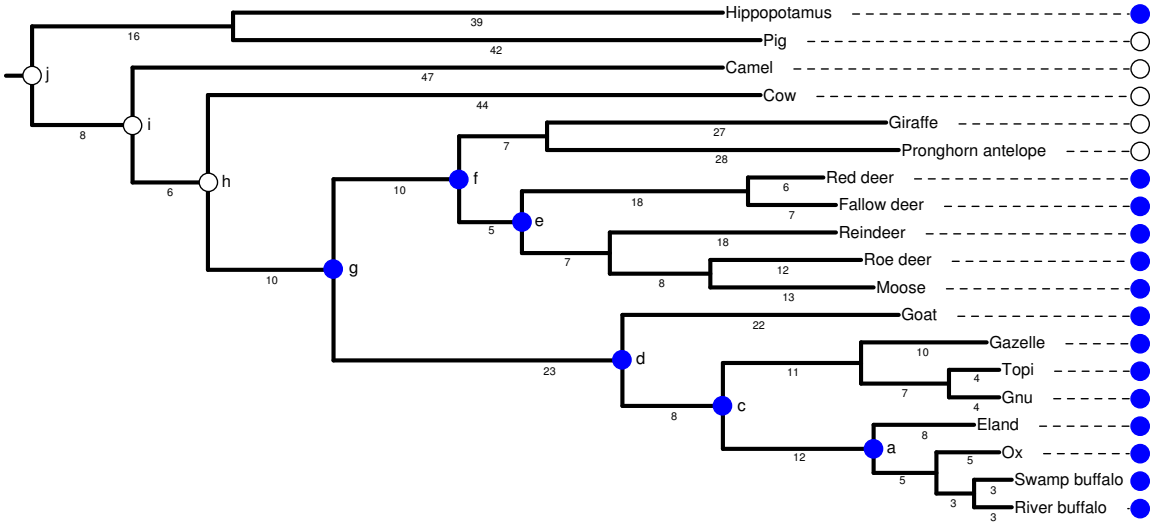

FIGURE 3.6. Parsimony reconstruction of ancestral states for residue 38 in artiodactyl ribonuclease from Jermann *et al.* (branch lengths in millions of years). Ancient characters of interest are labeled; glycine (G) in white and aspartic acid (D) in blue.

Schluter [**Sch95**] applied a maximum likelihood method based on Markov trait evolution to estimate the ancestral states in the same phylogeny. He estimated substitution rates of $q_{GD} = 0.0164$ and $q_{DG} = 0.0102$ and found the ML phylogeny. In contrast to the results from Jenmann *et al.*, his ML phylogeny had aspartic acid in all ancient species of interest, as opposed to glycine in the most ancient species **j**, **i**, and **h**. Schluter conducted log-likelihood ratio tests to assess the statistical significance of his phylogeny: while the likelihood ratios for the three most ancient species are all less than 1.4 ($p > 0.41$), the likelihood ratio for the ML estimate was 7.4 ($p = 0.045$), mildly significant at level 95%. However, applying variations of the log-likelihood

ratio test he concluded, in particular, that a transition from glycine to aspartic acid between ancestors **g** and **h** is not statistically supported.

Let us set $D = \{\mathtt{D}, \mathtt{G}\}$, a non-informative prior at the root $P(\theta_r = \mathtt{D}) = 0.5$, and assume a faster substitution rate $q_{GD} = 0.04$. In this scenario and for the labeled ancestral species of interest, the MAP estimate from our model coincides with the ML estimate from Schluter, and both the centroid and graph centroid coincide with the parsimony phylogeny from Jermann *et al.*. The MAP estimate assigns `D` to all ancestral species, while the centroid assigns `G` to four ancestral species. Figure 3.7 shows posterior marginal distributions for each node in the phylogeny; since the centroid is a consensus estimator we can easily identify the estimated character for each node by taking the state with posterior probability greater than one half. The graph centroid disagrees with the centroid in only one species, the common ancestral of hippopotamus and pig: the graph centroid assigns `G` and is thus closer to a parsimony solution.

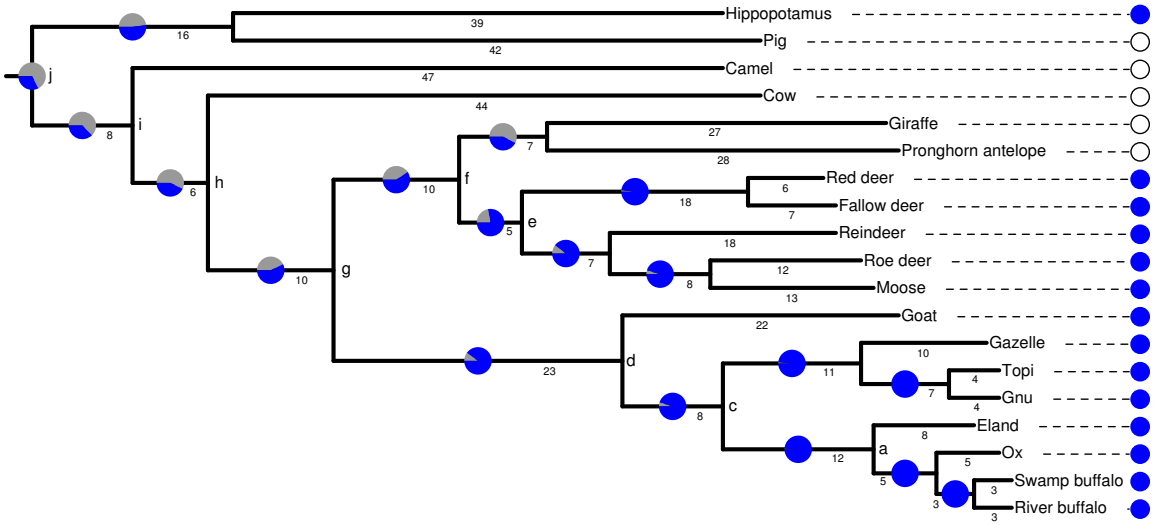

FIGURE 3.7. Centroid reconstruction of ancestral states for residue 38 in artiodactyl ribonuclease from Jermann *et al.* (branch lengths in millions of years). Ancient characters of interest are labeled; glycine (G) in white/gray and aspartic acid (D) in blue. Pie charts represent posterior marginal distributions at highest depth node.

Figure 3.8 presents the posterior probability mass and cumulative distributions for the Hamming distance centered at the centroid and MAP estimates. Even though the MAP estimate has an individual posterior mass that is almost ten times greater than the posterior mass of the centroid estimate, the centroid estimate accumulates more than half of the probability mass of the space within Hamming distance three, and thus situates itself in a region of high concentration of probability mass. Moreover, at level 95% the centroid estimate has a smaller credibility interval than the MAP estimate: $D_\alpha(\hat{\theta}_C) = 6 < D_\alpha(\hat{\theta}_G) = 9$.

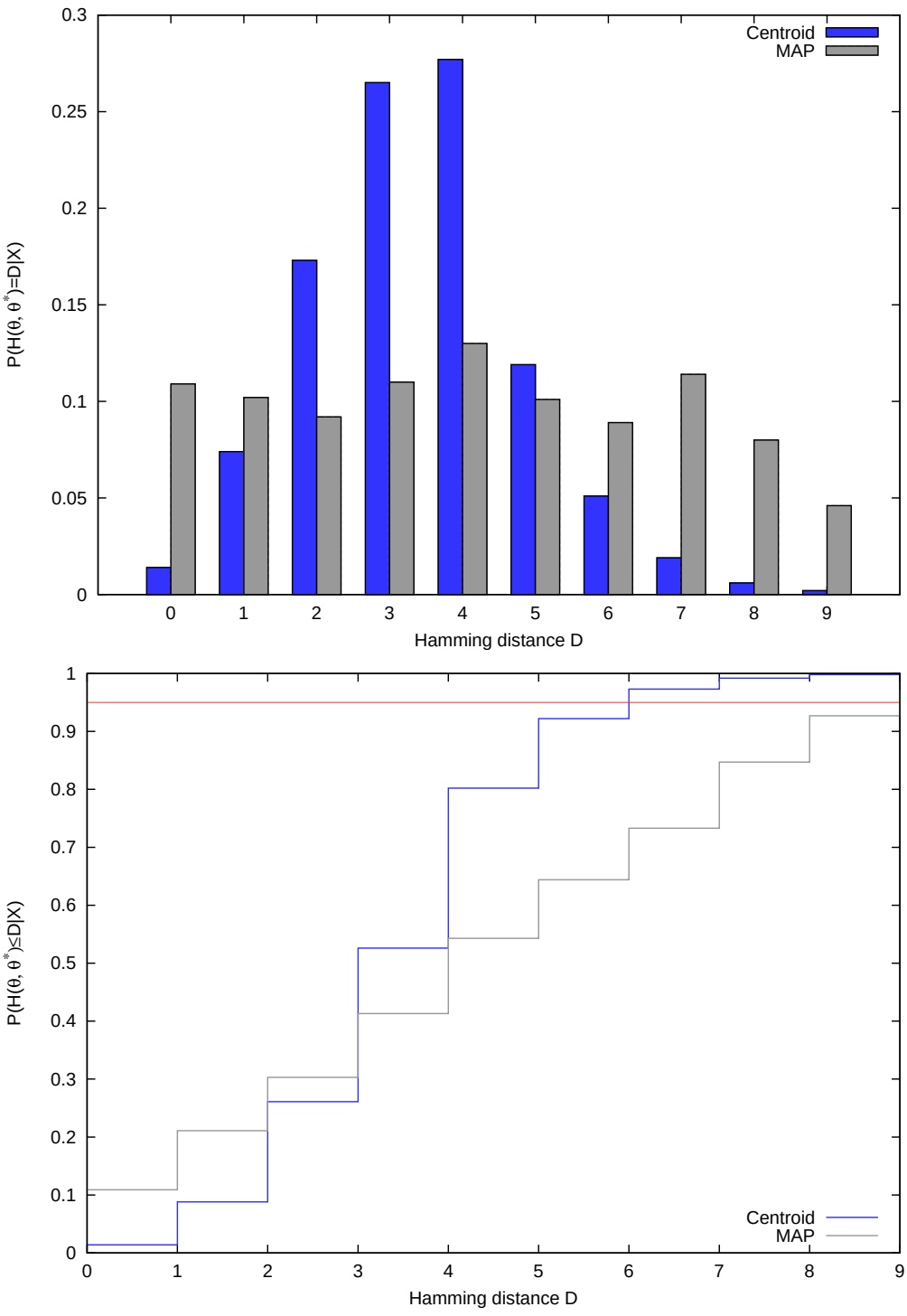

FIGURE 3.8. Posterior probability mass (top) and cumulative (bottom) distributions for Hamming distance centered at centroid and MAP phylogeny estimates.

CHAPTER 4

Applications of Centroid Estimation to Computational Biology

In this chapter we present some specific applications to computational biology. These applications are specific in the sense that the models that describe them are more complex and require a more detailed treatment than the one adopted for the previous chapters. We still pursue the same approach as before, however: we discuss possible prior and likelihood distributions and how to compute the marginal distributions, and later derive the centroid estimator from a discrete optimization problem.

1. Sequence alignment

Sequence alignment is one of the oldest and most common routines in computational biology. Earlier approaches considered sequence alignment as a pure optimization problem where a dynamic programming algorithm is usually employed to find optimal global [**NW70**], local [**SW81**], and restricted local block [**San72**] alignments between a pair of sequences. Heuristics are also common, specially in database searches where the sequence sizes are considerable [**KA93, AGM+90**].

Alternatively, since an optimal alignment depends on a score matrix and gap penalties, a fully probabilistic approach can be adopted where these input parameters are randomized. The main advantage of a probabilistic alignment is that it is principled: as usual in Bayesian statistics, the uncertainty about alignment parameters is averaged out and the statistical significance, or better, evidence of any specific alignment can be assessed. Moreover, depending on the probability distributions on the sequences and alignments, we can also profit from the same recursive structure that yielded efficient dynamic programming algorithms for optimal sequence alignment and obtain marginal distributions with no worse computational complexity.

To formalize the sequence alignment problem we adopt, with some modifications, the notation from [**LL99**]. Let R^1 and R^2 be two sequences to be aligned; $X \doteq (R^1, R^2)$ is our observed data. An alignment $\theta \subset \{0,1\}^{|R^1||R^2|}$ is a vector of indicator variables: θ_v, $v = \langle i, j\rangle$, indicates if residue R^1_i is aligned to residue R^2_j. Since each residue in one sequence can only be paired with at most another residue in the other sequence, we must require that $\sum_{i=1}^{|R^1|} \theta_{\langle i,j\rangle} \leq 1$ for $j = 1, \ldots, |R^2|$ and $\sum_{j=1}^{|R^2|} \theta_{\langle i,j\rangle} \leq 1$ for $i = 1, \ldots, |R^1|$. Moreover, to simplify the computations, transpositions are forbidden, that is, additional collinearity constraints should be considered: $\theta_{\langle i,j\rangle} + \theta_{\langle k,l\rangle} \leq 1$, for $1 \leq i < k \leq |R^1|$ and $1 \leq l < j \leq |R^2|$. We note that Θ is completely shaped by exclusivity constraints. Finally, the scoring matrix Ψ and gap penalties Λ are nuisance parameters in our model.

We also note that the gap positions are irrelevant for our model, that is, the order in which insertions and deletions occur within a sequence of gaps is immaterial as different orders generate the same prior probability for the alignment. It is, however, usual when using a dynamic programming scheme to find the best alignment according to some criteria to fix the gap ordering to avoid aliasing: first insertions are allowed and then deletions or vice-versa.

The joint distribution has the characteristic form

$$P(X, \theta, \Psi, \Lambda) = P(X|\theta, \Psi)P(\Psi)P(\theta|\Lambda)P(\Lambda),$$

where the likelihood is

$$P(X|\theta, \Psi) \propto \exp\left\{ \sum_{i=1}^{|R^1|} \sum_{j=1}^{|R^2|} \theta_{\langle i,j\rangle} \log \Psi(R^1_i, R^2_j) \right\}$$

and $\Psi = \sum_i \alpha_i \psi_i$, where each ψ_i is a scoring matrix from a family of known matrices like PAM [**DSO78**] or BLOSUM [**HH92**], and $\alpha \sim MN(1; \xi)$ for some hyperparameter vector ξ with $\xi_i > 0$ for all i and $\sum_i \xi_i = 1$.

Webb *et al.* [**WLL02**] and Liu and Lawrence [**LL99**] define a prior on Θ based on gap opening and extension penalties $\Lambda = (\lambda_o, \lambda_e)$:

$$P(\theta|\Lambda) \propto \exp\{g(\theta)\lambda_o + l(\theta)\lambda_e\},$$

where $g(\theta)$ and $l(\theta)$ return the number and total length of gaps in θ. Alternatively, Zhu *et al.* [**ZLL98**] derive a probabilistic version of Sankoff's algorithm where Λ is the number of gaps in θ and define a flat prior $P(\theta|g(\theta) = \Lambda) \propto 1$. It is common to define a non-informative prior for Λ as well.

One important consequence of the exclusivity-constrained shape of Θ and the particular form of the joint distribution of X and θ is that we are able to compute partial forward and backward probabilities and obtain $P(\theta_v = 1|X)$ efficiently [**LL99, WLL02, ZLL98**].

1.1. Centroid estimation. With the marginal probabilities $\pi_v = P(\theta_v = 1|X)$ available, we can now find the centroid alignment following the discussion from Section 3.1: we define the sensitivity-specificity ratio γ and seek

$$\hat{\theta}_C = \arg\max_{\hat{\theta}\in\Theta} \sum_{i=1}^{|R^1|} \sum_{j=1}^{|R^2|} \Big[\pi_{\langle i,j\rangle} - (1+\gamma)^{-1}\Big]\theta_{\langle i,j\rangle}.$$

We know, from Theorem 2, that if $\gamma \leq 1$ our work is much simpler since $\hat{\theta}_C$ is the consensus alignment. Not surprisingly, when $\gamma > 1$ we can still derive a dynamic programming algorithm to efficiently find $\hat{\theta}_C$. We present an iterative, more graph theoretical, version of such an algorithm below. We note that centroid alignments were previously found by Miyazawa [**Miy94**], albeit in a different probabilistic model, and called reliable alignments.

We draw a line representing sequence R^1 with numbers from 1 to $|R^1|$, and below it another line representing sequence R^2 also with numbers from 1 to $|R^2|$. To represent θ we now make a diagram where we draw a line, which we call *string*, between position i in R^1 and position j in R^2 if $\theta_{\langle i,j\rangle} = 1$. Now, let $G = (V, E)$ be the intersection graph of the strings: $V = \{\langle i, j\rangle : i = 1, \ldots, |R^1|, j = 1, \ldots, |R^2|\}$ and $(u, v) \in E$ if u

and v intersect, that is, if they cross or meet at an endpoint. Clearly, if $(u, v) \in E$ then θ_u and θ_v violate one of the constraints defining Θ. Any stable set in G is a valid θ and if we assign to each $v \in V$ the weight $p_v = \pi_v - (1+\gamma)^{-1}$, the centroid estimator is a *maximum weight stable set* with respect to p. Figure 4.1 illustrates the matching diagram and the intersection graph.

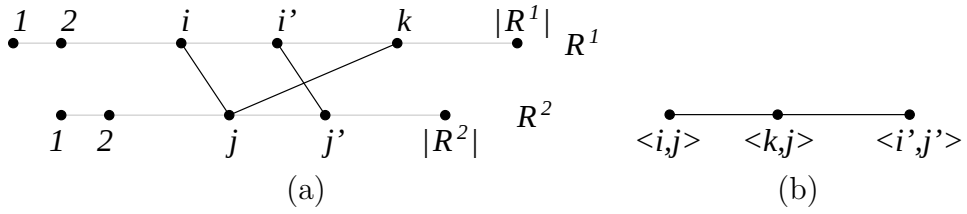

FIGURE 4.1. Sequence alignment (a) pairing diagram and (b) string intersection graph.

What makes finding a maximum weighted stable set in G tractable is the fact that G is a *cocomparability* graph, that is, the complement of G, $\bar{G}$, is transitively orientable and hence a comparability graph. A transitive orientation F of $\bar{G}$ can be obtained by defining the partial order $\prec$: for $u = \langle i, j\rangle$ and $v = \langle k, l\rangle$, $(u, v) \in F$ if and only if $u \prec v \doteq i < k$ and $j < l$. If $u \prec v$ we say that u precedes or is to the left of v in the alignment.

We can now translate our problem of finding a maximum weight stable set in G to finding a *maximum weight clique* in $\bar{G}$, for which we use Algorithm 4.1, adapted from Golumbic [**Gol04**]. The algorithm has two parts: it first updates partial maximum weights w and backpointers B in topological order in F with total complexity $O(|F|)$, and then builds a solution by backtracking from B in $O(|V|)$ time.

We note that, for our specific centroid alignment problem, there is no need to explicitly represent $\bar{G}$ since we can define $\textsc{Adj}_F(\langle i, j\rangle) = \{\langle k, l\rangle : k < i \text{ and } l < j\}$ and substitute the loop over the topological order σ_F by loops on $\langle i, j\rangle$ with $i = 1, \ldots, |R^1|$ and then $j = 1, \ldots, |R^2|$.

Input: Comparability graph $G = (V, F)$ and weights p on V.
Output: A maximum weight clique K of G w.r.t. p.
Data: Topological order σ_F.
function `MaxWeightClique`(V, F, p);
`% compute` $w(\cdot)$ `and` $B(\cdot)$
for $i \leftarrow 1, \ldots, |V|$ **do**
  $v \leftarrow \sigma_F^{-1}(i)$;
  if $\textsc{Adj}_F(v) = \emptyset$ **then**
    $w(v) \leftarrow p(v)$;
    $B(v) \leftarrow \Lambda$;
  else
    $y \leftarrow \arg\max_{x \in \textsc{Adj}_F(v)} w(x)$;
    $w(v) \leftarrow p(v) + w(y)$;
    $B(v) \leftarrow y$;
  end
end
`% build solution`
$y \leftarrow \arg\max_{v \in V} w(v)$;
$K \leftarrow \{y\}$;
$y \leftarrow B(y)$;
while $y \neq \Lambda$ **do**
  $K \leftarrow K \cup \{y\}$;
  $y \leftarrow B(y)$;
end
return K;

ALGORITHM 4.1: Maximum weight clique of a comparability graph.

2. RNA secondary structure prediction

Along with sequence alignment, prediction of RNA secondary structure is also one of most fundamental problems in computational biology. Also similarly to sequence alignment, defining the "best" secondary structure for a RNA sequence was for many years regarded as an optimization problem where the structure with "minimum free energy" was sought [**NPGK78, Zuk03**].

McCaskill [**McC90**] applied a probabilistic interpretation for free energy in RNA structures and was able to compute the partition function describing the ensemble of structures given a RNA sequence. Following McCaskill, Ding *et al.* [**DCL05**] developed a stochastic back-trace algorithm to draw samples directly from the Boltzmann weighted ensemble of secondary structures. Another probabilistic approach is

to design a stochastic context-free grammar (SCFG) with rules for left, right, pair and bifurcation productions [**DEKM99**]. Regardless of the (probabilistic) model, however, we can always represent the joint distribution of a RNA sequence X and its secondary structure θ as

$$P(X, \theta) \propto \exp\{-\beta E(X, \theta)\}$$

where β is the inverse temperature parameter and E is the "free energy" of X and θ. Before further discussing these models, let us give more details about the setup.

Our observed data X is a RNA sequence of length n. A point θ in our parameter space is a secondary structure represented by a binary vector of indicator variables: θ_v, $v = \langle i, j \rangle$, with $i = 1, \ldots, n-2$ and $j = i+2, \ldots, n$, indicates if position i in the X is paired to position j (consecutive positions do not form pairs). Each position can only be paired with at most one other position, and so we require that $\sum_{i=1}^{n-2} \theta_{\langle i,j \rangle} \leq 1$ for $j = i+2, \ldots, n$ and $\sum_{j=i+2}^{n} \theta_{\langle i,j \rangle} \leq 1$ for $i = 1, \ldots, n-2$. Even with these constraints there exists an exponentially large number of possible structures, and thus we need to further reduce Θ. Similarly to sequence alignment, we introduce *no-pseudoknot* constraints of the form $\theta_{\langle i,j \rangle} + \theta_{\langle k,l \rangle} \leq 1$ with $1 \leq i < k < j < l \leq n$ (that is, a pseudoknot occurs when both $\langle i, j \rangle$ and $\langle k, l \rangle$ are paired).

These exclusivity constraints can be more easily visualized if we draw a circle representing X with numbers from 1 to n and, for a given θ, we draw chords pairing positions i to positions j if $\theta_{\langle i,j \rangle} = 1$. Then, if two chords meet at an endpoint they violate the first set of constraints, while if two chords cross they violate a no-pseudoknot constraint. Furthermore, we can construct a graph $G = (V, A)$ where each vertex corresponds to a chord $v = \langle i, j \rangle$ and $(u, v) \in A$ if u and v intersect, that is, meet or cross. G is properly called a *circle* graph, and it is easy to see that circle graphs are equivalent to *overlap* graphs [**Gol04**]. Figure 4.2 illustrates the pairing diagram and the circle graph representing the exclusivity constraints.

Definition 8 (Overlap graph). Let $G = (V, A)$ be a graph and $\mathcal{I} = \{I_v\}_{v \in V}$ be a family of intervals on a line such that there exists a direct, one-to-one, correspondence

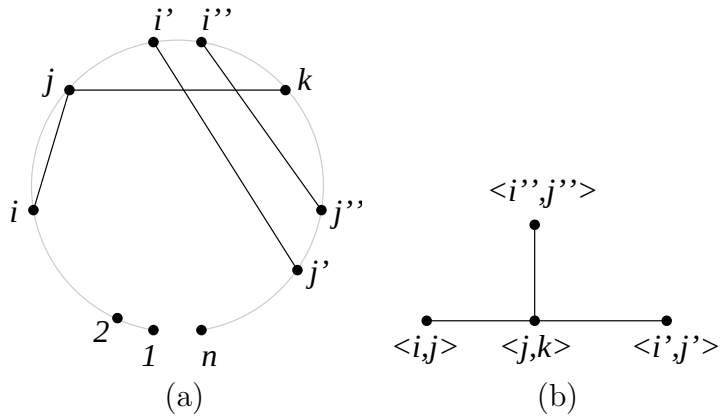

FIGURE 4.2. RNA secondary structure (a) pairing diagram and (b) constraint circle graph.

between each $v \in V$ and an interval in $\mathcal{I}$. G is an *overlap* graph if $(u, v) \in A$ if and only if $I_u \cap I_v \neq \emptyset$, $I_u \not\subset I_v$, and $I_v \not\subset I_u$, that is, I_u and I_v overlap, but one does not contain the other.

Any pair of distinct vertices $u, v \in V$ have to satisfy exactly one of the following relations, which we can use to define edge sets:

Overlap: $(u, v) \in A$ iff $I_u \cap I_v \neq \emptyset$, $I_u \not\subset I_v$, and $I_v \not\subset I_u$;

Containment: $(u, v) \in B$ iff either $I_u \subset I_v$ or $I_v \subset I_u$;

Disjointness: $(u, v) \in D$ iff $I_u \cap I_v = \emptyset$.

We note that the three relations are symmetric and they partition the set of all distinct pair intervals. Moreover, (V, A) is the overlap graph represented by $\mathcal{I}$ by definition, $(V, A + B)$ is an *interval* graph, and both (V, B) and (V, D) are *comparability* graphs since they admit transitive orientations given by $C = \{(u, v) \in V^2 : I_u \subset I_v\}$ and $F = \{(u, v) \in V^2 : I_u \prec I_v\}$, where $I_u \prec I_v$ denotes that I_u lies entirely to the left of I_v, respectively. In our circle graph representation, if $e = (\langle i, j\rangle, \langle k, l\rangle)$ then $e \in C$ if $k < i < j < l$ and $e \in F$ if $i < j < k < l$.

Under this overlap graph representation any point θ corresponds to a stable set $S = \{v \in V : \theta_v = 1\}$. It is interesting to note that each θ also corresponds to a *unique* parse tree for the following context-free grammar $\mathcal{G}$:

$$
\begin{aligned}
S &\rightarrow \texttt{A} \mid \texttt{C} \mid \texttt{G} \mid \texttt{U} \\
L &\rightarrow SL \mid SP \mid SB \mid \texttt{A} \mid \texttt{C} \mid \texttt{G} \mid \texttt{U} \\
P &\rightarrow SR \\
R &\rightarrow LS \mid PS \mid BS \\
B &\rightarrow PL \mid PP \mid PB
\end{aligned}
$$

The equivalence between Θ and $\mathcal{G}$ means that we just need to define position pairings in θ to fully specify a parse tree T_θ for X and vice-versa. This relation follows ultimately from the priority of left productions in $\mathcal{G}$: unpaired positions are left or single productions L or S, right productions R can only occur after a pair production P, and the left child of a bifurcation B can only be a pair production P.

If for each node N in a parse tree T we assign the subsequence with left and right endpoints $l(N)$ and $r(N)$ that the subtree rooted at N spans in X, then to obtain θ for T we just need to set $\theta_{\langle l(P),r(P)\rangle} = 1$ for all pair productions P in T. Figure 4.3 shows prototypical node labels for the production rules in $\mathcal{G}$; we define X as a generic rule that can be L, P, or B. Simple parse examples are also pictured in Figure 4.3. Given a post-order σ of T it is straightforward to generate a labeling of the nodes and the equivalent θ from Algorithm 4.2. For simplicity we assume that if a S (or L) node has only one child, a terminal t, then $\textsc{Left}(L) = \textsc{Right}(L) = t$.

To define a parse tree from some θ is more involved, but we can argue inductively as follows: the parse subtree that covers $X[i:j]$ can be constructed by first identifying the leftmost pair $\theta_{\langle k,l\rangle} = 1$; if no pair exists between i and j then $X[i:j]$ is fully generated by a series of L productions. Now, from i to $k-1$ (inclusive) we must have a series of L productions since only L rules might have no P as a production. If $l = j$ then we can recurse and define a subtree for $X[k+1:j-1]$; otherwise, we must have a bifurcation and we recurse in two branches to define subtrees for $X[k+1:l-1]$ and

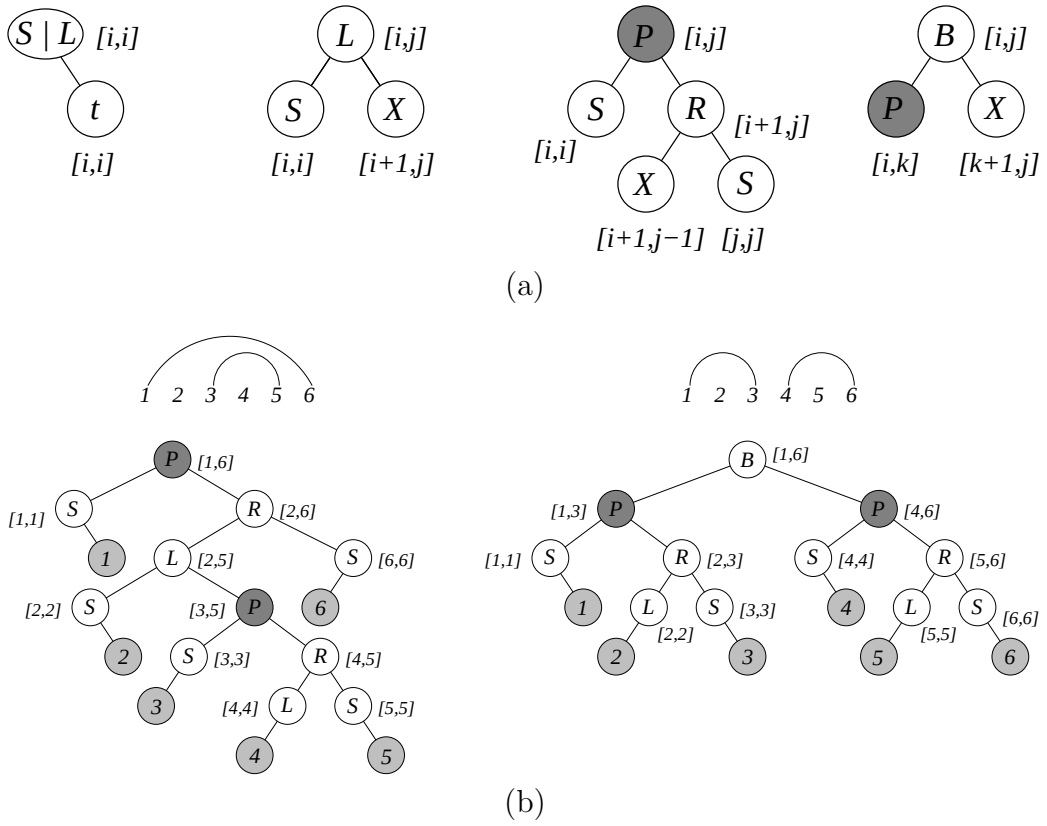

FIGURE 4.3. (a) Node labeling for production rules in context-free grammar $\mathcal{G}$. (b) Two examples of node labeled parse trees in $\mathcal{G}$: filaments above and below the sequences indicate pairs.

Input: Parse tree T.
Output: Vector θ equivalent to T.
Data: Post-order σ of T.
function `ParseTreeToVector(`T`)`;
$n \leftarrow 0$;
for $i \leftarrow 1, \ldots, |T|$ **do**
  $N \leftarrow \sigma^{-1}(i)$;
  if N *is terminal* **then**
    $n \leftarrow n + 1$; $l(N) \leftarrow r(N) \leftarrow n$;
  else
    $l(N) \leftarrow l(\text{LEFT}_T(N))$; $r(N) \leftarrow r(\text{RIGHT}_T(N))$;
    if $N = P$ **then** $\theta_{\langle l(N), r(N)\rangle} = 1$;
  end
end
return θ;

ALGORITHM 4.2: Parse tree T to vector θ.

$X[l+1:j]$. Figure 4.4 illustrates the two last cases. For completeness sake we present Algorithm 4.3 that is the converse, in recursive form, to Algorithm 4.2 (an iterative form can be obtained if we use a topological order on C, the containment relation on the overlap graph G). Since both algorithms produce the same subsequence labeling, the equivalence follows.

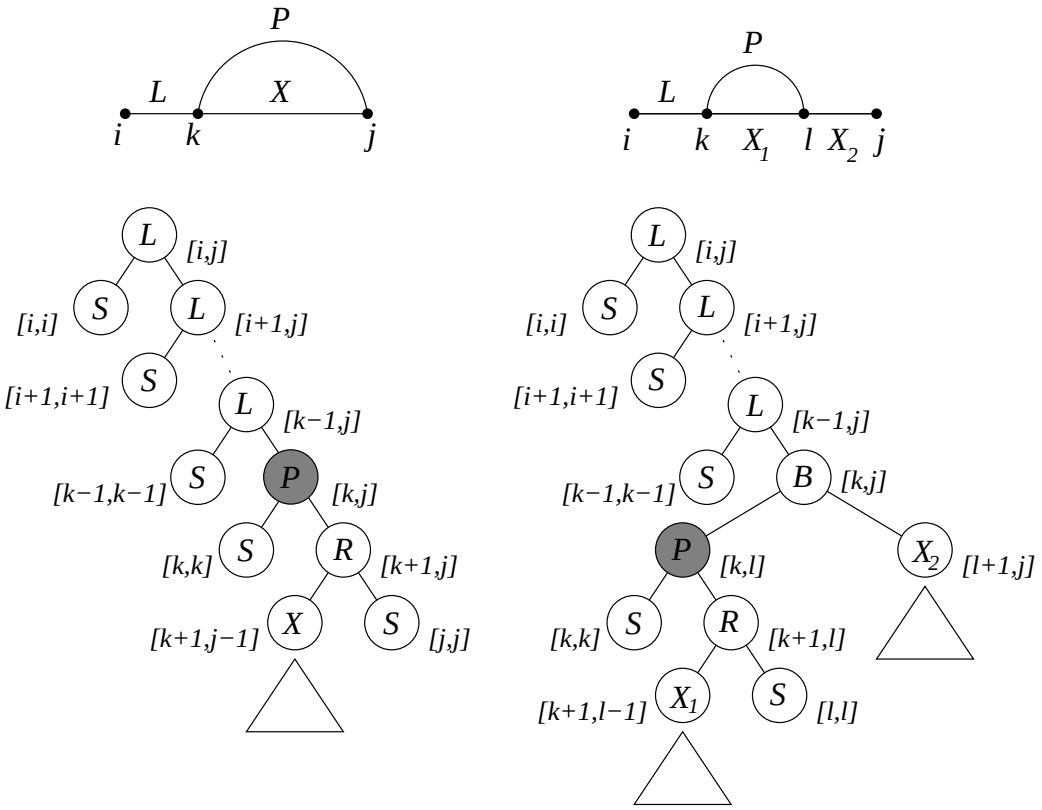

FIGURE 4.4. Recursively building a parse subtree for $X[i:j]$: the leftmost pair production P between positions k and l defines branch points with either P, if $l = j$, or B, if $l \neq j$, as subtree roots.

The equivalence between Θ and the yield of $\mathcal{G}$ is a powerful relation: it enables us to work on a much simpler space than the set of parse trees of a sequence. For instance, the centroid estimator can be easily defined in terms of generalized Hamming loss on indicator vectors of pairings θ. Moreover, as the equivalence of θ to a (parse) tree suggests, many computations can be performed efficiently due to the exclusivity constraints on Θ. In particular, given the joint distribution on X and θ, we can

Input: Vector θ, positions i and j, $i \leq j$.
Output: Parse subtree that spans $X[i:j]$.
function `VectorToParseTree(`θ, i, j`)`;
if $i = j$ **then**
  $N \leftarrow L$;
else
  if $\exists k : \theta\langle i, k\rangle = 1$ **then**
    if $k = j$ **then**
      $N \leftarrow P$; $C \leftarrow R$;
      $\textsc{Left}(N) \leftarrow S$; $\%\langle i, i\rangle$
      $\textsc{Right}(N) \leftarrow C$; $\%\langle i+1, j\rangle$
      $\textsc{Right}(C) \leftarrow S$; $\%\langle j, j\rangle$
      $\textsc{Left}(C) \leftarrow$ `VectorToParseTree(`θ, $i+1$, $j-1$`)`;
    else
      $N \leftarrow B$;
      $\textsc{Left}(N) \leftarrow$ `VectorToParseTree(`θ, i, k`)`;
      $\textsc{Right}(N) \leftarrow$ `VectorToParseTree(`θ, $k+1$, j`)`;
    end
  else
    $N \leftarrow L$;
    $\textsc{Left}(N) \leftarrow S$; $\%\langle i, i\rangle$
    $\textsc{Right}(N) \leftarrow$ `VectorToParseTree(`θ, $i+1$, j`)`;
  end
end
return N; $\%\langle i, j\rangle$

ALGORITHM 4.3: Vector θ to parse subtree for $X[i:j]$.

compute the centroid RNA structure with no worse complexity than computing the marginal posterior distributions $P(\theta_v = 1|X)$ [**DCL05, DCL06, DEKM99**].

2.1. Centroid estimation. Since θ is a binary vector restricted by exclusivity constraints, we can pursue a similar approach to sequence alignment with the following centroid estimator, secondary structure,

$$\hat{\theta}_C = \arg\max_{\hat{\theta}\in\Theta} \sum_{i=1}^{n-2} \sum_{j=i+2}^{n} \left[\pi_{\langle i,j\rangle} - (1+\gamma)^{-1}\right] \theta_{\langle i,j\rangle},$$

where $\pi_v = P(\theta_v = 1|X)$.

As in sequence alignment, when $\gamma \leq 1$ we do not need to resort to dynamic programming since Theorem 2 also applies here and it is enough to provide the consensus

structure. Ding *et al.* [**DCL05**] find the centroid using $\gamma = 1$ from posterior sampling θ using an stochastic backtrace routine. They also cluster the samples and find centroids for each group. Their results are usually better than MFE minimization methods when compared to golden standards. Mathews also reports centroids for a variety of γ values [**MDC$^+$04, Mat04**].

When $\gamma > 1$ we also need to find the maximum weight stable set on a graph G representing forbidden relations between vertices with respect to weights $p_v = \pi_v - (1+\gamma)^{-1}$ for each $v \in V(G)$, but now we have a more complex task than finding a centroid alignment since the overlap graph G is not a cocomparability graph. However, as expected, overlap graphs have efficient, polynomial time algorithms for finding maximum stable sets. We recall the interval representation of an overlap graph and, following the presentation in Golumbic [**Gol04**], define, for all $v \in V$, $U(v) = \{u \in V : (u, v) \in C\}$, the set of intervals that are contained in I_v.

Input: Interval set V, containment and left relations C and F, and weights p on V.
Output: A maximum weight stable set S of (V, A) w.r.t. p.
Data: Topological order σ_C.
function `MaxWeightStableSet`(V, C, F, p);
for $i \leftarrow 1, \ldots, |V|$ **do**
  $x \leftarrow \sigma_C^{-1}(i)$;
  if $U(x) = \emptyset$ **then**
    $S(x) \leftarrow \{x\}$;
    $w(x) \leftarrow p(x)$;
  else
    $T \leftarrow$ `MaxWeightClique`$(U(x),\ F[U(x)],\ w)$;
    $S(x) \leftarrow \{x\} \cup$ `MaxWeightStableSet`$(U(x), p)$;
    $w(x) \leftarrow \sum_{u \in S(x)} p(u)$;
  end
end
$T \leftarrow$`MaxWeightClique`(V, F, w);
return $\cup_{v \in T} S(v)$;

ALGORITHM 4.4: Maximum weight stable set of an overlap graph.

For each vertex x in the topological order σ_C on C, Algorithm 4.4 uses the maximum weight clique routine presented in the last section on the subgraph of F induced by the neighbors of x in C, $U(x)$, to build a larger maximum weight stable set S.

In our RNA structure problem, this means, for $x = \langle i, j \rangle$, finding non-overlapping pairings contained in x with maximum total weight. At the end of the loop we have computed weights $w(x) = p(x) + \max_{S \subset U(x)} \sum_{v \in S} p(v)$ where S are stable sets. Finally, we just need to compute the maximum weight clique on F w.r.t. w to find the maximum weight stable set. We note that for our specific RNA problem the loop over the topological order σ_C is implicitly given by a loop over $l = 2, \ldots, n-1$, $i = 1, \ldots, n-l$ and $x = \langle i, i+l \rangle$.

3. Phylogenetic motif search

Let T be a phylogeny relating s extant species at the leaves of T and $s-1$ species in the internal nodes of T. For each extant species j we observe a sequence $X_{\cdot,j}$ of length n. In phylogenetic motif search we have two objectives: similarly to reconstruction of ancestral states, we want to infer the sequence composition at the internal nodes in T, but we also seek to identify binding sites of a set of motifs at each sequence and species, including ancestral ones. As a typical example, the sequences X represent promoter regions of some gene of interest across species, and the motif represents binding affinities of some protein, like a transcription factor. The questions are then what are the ancestral sequences and where does the protein bind in each species promoter, including ancestral sequences.

To simplify our task, we only allow exactly one binding site for one fixed length motif at each sequence and species. Moreover, we further assume that the binding site is at the same position for all sequences. Before addressing phylogenetic motif search, we start by discussing pure motif search.

3.1. Motif search. Let us define, for our purposes, a motif $\mathcal{M}$ as a sequence composition of fixed length l, that is, a probability distribution over A^l where $A = \{a_j\}_{j=1}^m$ is our sequence alphabet. The higher the binding affinity of $\mathcal{M}$ to some sequence $M \subset A^l$, the higher the probability of M. It is usual [**LNL95**] to assume positional independence and represent a motif as a product multinomial distribution

with parameters α_i for the i-th position in $\mathcal{M}$,

$$P(M) = \prod_{i=1}^{l} \prod_{j=1}^{m} \alpha_{ij}^{I(m_i=a_j)},$$

where $M = (m_i)_{i=1}^{l}$ is a motif realization.

Let X be a sequence of length n in A. Our parameter θ, also of length n, represents motif binding sites in the following way: for position i, if $\theta_i = 0$ then no motif binding site overlaps i, otherwise if $\theta_i = k$ then motif k has a binding site covering i. Since in our simple model we are assuming only one motif and one binding site, θ has the characteristic pattern

$$00\cdots0\underbrace{11\cdots1}_{l}0\cdots0$$

with l consecutive ones marking the binding site. Now, since θ is a binary vector, we could set the gain factor γ and use our generalized centroid estimator. However, we can adopt a different, more concise, representation for θ if we observe that we only need the position where the site starts: $\theta' = \min\{i : \theta_i = 1\}$.

If the binding sites of θ and $\tilde{\theta}$ do not overlap, that is, if $|\theta' - \tilde{\theta}'| > l$, then $G(\theta, \tilde{\theta}) = n - 2l$ since both parameters agree only where there is no binding, on $n - 2l$ positions, and that warrants an unitary gain per position. If the sites of θ and $\tilde{\theta}$ do overlap, the parameters agree where the sites overlap, for $n - (l + |\theta' - \tilde{\theta}'|)$ positions, and where there is no binding, for $l - |\theta' - \tilde{\theta}'|$ positions, and then $G(\theta, \tilde{\theta}) = n - (l + |\theta' - \tilde{\theta}'|) + \gamma(l - |\theta' - \tilde{\theta}'|)$. Putting both conditions together, we have a gain function given by

$$G(\theta, \tilde{\theta}) = n - 2l + (1 + \gamma)\max\{0, l - |\theta' - \tilde{\theta}'|\}.$$

We can now fully adopt the more concise representation for our binding site location parameter. A common choice for the prior on Θ is the multinomial $MN(1, \psi)$, $\sum_{i=1}^{n} \psi_i = 1$, where we can either set a non-informative prior with $\psi_i = 1/n$ or further impose a hierarchical model by setting a Dirichlet distribution on ψ. To specify the

likelihood of θ we define the positional function ϕ

$$\phi_i(\theta) = \begin{cases} \text{if } 0 \leq i - \theta < l : i - \theta + 1 \\ \text{else} : 0 \end{cases}$$

that returns the relative position of i in the binding site; if the site does not overlap i, i is said to be a *background* position and has null relative position. We assume that, conditional on θ, X_i is independent of X_j, and define a multinomial distribution for the background positions in X (given θ) with parameter β. The likelihood is then given by

$$P(X|\theta) = \prod_{j=1}^{m} \prod_{i:\phi_i(\theta)=0} \beta_j^{I(X_i=a_j)} \prod_{i:k=\phi_i(\theta)>0} \alpha_{kj}^{I(X_i=a_j)}.$$

where the first product is over background positions and the second product is over positions in the motif binding site.

Intuitively, as we try to find the centroid estimator for the binding site, we should expect the centroid to overlap with positions with high posterior probability, affinity to the motif. In fact, the centroid can be easily seen to be

$$\hat{\theta}_C = \arg\max_{\tilde{\theta}\in\Theta} E_{\theta|X} G(\theta, \tilde{\theta}) = \arg\max_{\tilde{\theta}\in\Theta} \sum_{\theta:|\theta-\tilde{\theta}|<l} (l - |\theta - \tilde{\theta}|) P(\theta|X)$$

and so it is independent of γ, as expected. In addition, we note that since $G(\theta, \tilde{\theta})$ is proportional to a discrete convolution, the centroid can be obtained in $O(n \min\{l, \log n\})$ time.

3.2. Incorporating phylogenies. Back to our original problem, let us now accommodate the phylogeny T in the motif search model. We keep the conditional independence of positions given the binding site, but while in the motif model we have a product model for letters at each position, now we have phylogenies. Moreover, we assume that the multinomial distribution for letters at a position in the motif model now describes the distribution for letters at the *root* of the phylogeny at the

position. Given the distribution at the root, we can apply the distribution on Section 4.2 of Chapter 3 for letters in the remaining internal nodes and for the likelihood given the observed letters in the leaves of the phylogeny, for each position.

Our parameter space has the form $\theta = (\theta_0, \theta_1, \ldots, \theta_n)$, where θ_0 is the binding site position and the remaining θ_i are vectors where $\theta_{i,k}$ indicates the letter at position i and internal node k. We denote the root of T as r, the leaf set of T as $\mathcal{L}(T)$, and the parent of node k in T, as k^*. We adopt the same notation as in the previous section for the multinomial parameters α_i, for each motif position, but denote by α_0 the background parameter. Given our previous considerations, the joint distribution of X and θ is then

$$\begin{aligned} P(X,\theta) &= \prod_{i=1}^{n} P(X_i, \theta_i|\theta_0)P(\theta_0) \\ &= \prod_{i=1}^{n} \prod_{k\in\mathcal{L}(T)} P(X_{i,k}|\theta_{i,k^*}) \prod_{k\in T\setminus(\mathcal{L}(T)\cup\{r\})} P(\theta_{i,k}|\theta_{i,k^*})P(\theta_{i,r}|\theta_0)P(\theta_0) \end{aligned}$$

where $\theta_{i,r}|\theta_0 \sim MN(1, \alpha_{\phi_i(\theta_0)})$ and $P(\theta_{i,k}|\theta_{i,k^*})$ depends on the substitution matrix Q and the branch length between k and k^*, as in Section 4.2 of Chapter 3. Figure 4.5 presents the conditional independence graph for X and θ: a tree rooted at θ_0, whose children are phylogenies T_i, one for each position i.

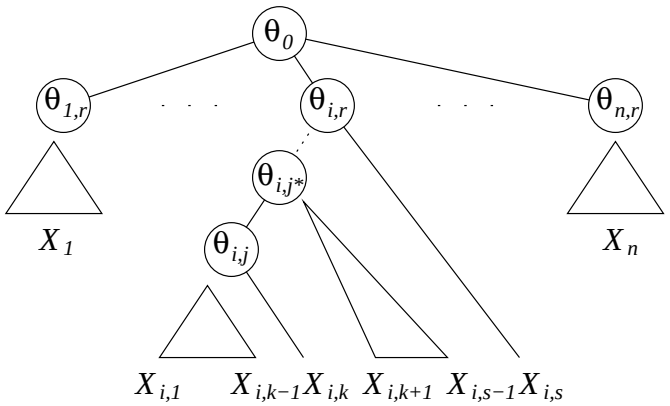

FIGURE 4.5. Joint conditional independence graph for phylogenetic motif model.

Using the first part of Algorithm 3.3 in Chapter 3 we can obtain $\pi_i(\theta_{i,r}) = P(X_i|\theta_{i,r})$ for each position i in T_i. The root likelihoods $\pi_i(\theta_{i,r})$ enable us to compute

the marginal posterior for the binding site position,

$$P(\theta_0|X) \propto \prod_{i=1}^{n} \sum_{j=1}^{m} P(X_i|\theta_{i,r} = a_j)P(\theta_{i,r} = a_j|\theta_0)P(\theta_0),$$

and the conditional joint

$$P(X, \theta_{i,r}|\theta_0) = P(X_i|\theta_{i,r})P(\theta_{i,r}|\theta_0) \prod_{k \neq i} \sum_{j=1}^{m} P(X_k|\theta_{k,r} = a_j)P(\theta_{k,r} = a_j|\theta_0)$$

from which we can further obtain $P(X, \theta_{i,r}, \theta_0)$, $P(\theta_0, \theta_{i,r}|X)$,

$$P(X, \theta_{i,r}) = \sum_{j=1}^{n} P(X, \theta_{i,r}|\theta_0 = j)P(\theta_0 = j),$$

and hence $P(\theta_{i,r}|X)$. Now we can continue, for each phylogeny T_i, with the second part of Algorithm 3.3 to obtain marginals $\pi_i'(\theta_{i,k}) = P(\theta_{i,k}|X)$ and joint marginals $P(\theta_{i,k}, \theta_{i,k^*}|X)$.

Since there are no constraints between the components of θ, it is straightforward to define the generalized centroid. For the binding site position we recall the result from the previous section,

$$(\hat{\theta}_C)_0 = \arg\max_{j=1,\ldots,n} \sum_{\theta_0 : |\theta_0 - j| < l} (l - |\theta_0 - j|)P(\theta_0|X)$$

while for the remaining ancestral state components we just need to take the marginal maximizer

$$(\hat{\theta}_C)_{i,k} = \arg\max_{a_j : j=1,\ldots,m} (G_{i,k}\mathbf{p}_{i,k})_j$$

where $G_{i,k}$ is a gain matrix for position i and internal node k and $\mathbf{p}_{i,k} = P(\theta_{i,k}|X)$.

Alternatively, to account for the conditional dependency structure among components in the parameter space we can derive the graph centroid estimator. The tree decomposition here is elementary since the graph is a tree, as depicted in Figure 4.5, and since all cliques are edges, we just need pair posterior joints $P(\theta_0, \theta_{i,r}|X)$ and $P(\theta_{i,k}, \theta_{i,k^*}|X)$ for all positions i and internal nodes k; all needed posteriors are given above. The procedure for computing the graph centroid is then direct and similar

to the one in Section 4.2 of Chapter 3: we just need to traverse the tree in bottom-up fashion, storing partial maximum sums and backtrack pointers for later centroid reconstruction.

CHAPTER 5

Conclusion

Maximum likelihood and MAP estimators have dominated discrete prediction and estimation for a long time. In this work we applied statistical decision theory with a Hamming loss function to present a new contender, the centroid estimator. We showed that the centroid estimator is also the minimum Bayes risk estimator for a family of loss functions, including the important case when the data being compared is not only categorical, but also ordinal and interval.

These estimators center themselves in the posterior weighted ensemble by balancing the forces of the members based on their posterior probabilities and their component-wise distances from the centroid estimator. Given the findings in three computational biology applications that centroid estimators substantially improve prediction of ground truth reference sets without modification of the underlying probabilistic model [**CL08**] and perhaps more importantly their principled foundation, centroid estimators offer a promising avenue for improved estimation and prediction in discrete high-dimensional inference problems.

Centroid estimators are formally characterized as solutions to discrete optimization problems having posterior marginal distributions as inputs. These optimization problems are intractable in general, but under certain constraints they become more amenable, attaining polynomial time complexity, or even trivially linear. For instance, an important case arises when the parameter space is multidimensional binary and shaped by exclusivity constraints, as seen in Section 3.1.

Centroid estimation is discussed and efficient algorithms are presented for a number of important, well known and ubiquitous models, including hidden Markov models, change point models, and graphical models. One of the main objectives of this research was to focus on efficient estimation; to this intent we provided, to the best of

our knowledge, the broadest characterization of models where the centroid estimator can be obtained in polynomial time, namely of graphical models defined on bounded tree-width graphs.

Finding the centroid estimation involves two phases: marginalization and optimization. Thus, the overall complexity of centroid estimation depends on the worst result from these steps. The first phase depends on the structure of the probability space, and so it is coupled to the model specification. We often are able to efficiently compute, in polynomial time, the marginal distributions or the mode of the distribution when the space shows a recursive structure, like a Markov property. The optimization step, on the other hand, depends on the constraints shaping the parameter space. However, the constraints are usually also defined by the probability structure of the model: we simply require a feasible solution to be a positive probability configuration. If this is the case, finding the centroid estimator requires no worse complexity than computing the marginal distributions, and hence the mode, MAP estimator.

Computational biology has recently been a fruitful source of high-dimensional discrete inference problems. We also present and discuss centroid estimation for the most classical and important problems in this field, some of them a direct application of the general results in centroid estimation. Another important contribution is the characterization of centroid estimation in sequence alignment and RNA secondary structure prediction problems in graph theoretical terms. This approach results in iterative versions of the usual dynamic programming recursive algorithms for these problems. The justification for such approach is twofold: first, as it reveals the underlying structure of the discrete optimization problem, it facilitates understanding and complexity analysis; second, the development of possible later extensions to our models benefits from this characterization—for instance, since the intersection graph of strings in the sequence alignment is a perfect graph, we immediately know that finding the chromatic number, for example, should require polynomial time.

If the posterior space is multimodal, the centroid estimator might not be reliable as it might be placed in-between and far from modes. The common approach in this case is to sample from the posterior space, cluster the samples to isolate the modes, and find restricted centroid estimates for each group. A more principled approach we intend to pursue is to perform an "implicit" clustering routine by defining centroids that should be centered in subspaces of high probability mass. This promising avenue is more akin to feature extraction as the subspaces could be specified by particular component realizations that induce high marginal joint probabilities. Another possible treatment is to define an estimator based on a thresholded Hamming loss function and thus restricting the subspace defining the centroid to a ball with radius given by the threshold. Such an estimator would be a compromise between the MAP estimator, when the radius is null, and the centroid, when the radius is the dimension of the space.

There are many problems in computational biology that would benefit from centroid estimation, like alternative splicing, simultaneous alignment and structure prediction of RNA, disease association studies, and genome rearrangement. These problems represent attractive and promising directions for future applied work.

Bibliography

[AGM+90] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, *Basic local alignment search tool*, Journal of Molecular Biology **215** (1990), 403–410.

[Att00] H. Attias, *A variational Bayesian framework for graphical models*, Adv. Neural Inf. Process. Syst. **12** (2000), 209–215.

[BB72] Umberto Bertele and Francesco Brioschi, *Nonserial dynamic programming*, Academic Press, Inc., Orlando, FL, USA, 1972.

[Bel57] Richard Ernest Bellman, *Dynamic programming*, Princeton University Press, New Jersey, USA, 1957.

[Bes86] Julien Besag, *On the statistical analysis of dirty pictures*, J. R. Stat. Soc. **48** (1986), no. 3, 259–302, with discussions.

[BG03] M. J. Beal and Z. Ghahramani, *The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures*, Bayesian Stat. **7** (2003), 453–464.

[Bod97] Hans L. Bodlaender, *Treewidth: Algorithmic techniques and results*, Proceedings 22nd International Symposium on Mathematical Foundations of Computer Science, MFCS'97, Lecture Notes in Computer Science (Berlin, Germany) (I. Privara and P. Ruzicka, eds.), vol. 1295, Springer-Verlag, 1997, pp. 29–36.

[CL00] Bradley P. Carlin and Thomas A. Louis, *Bayes and empirical Bayes methods for data analysis*, second ed., Chapman and Hall/CRC, New York, USA, 2000.

[CL08] Luis E. X. Carvalho and Charles E. Lawrence, *Centroid estimation in discrete high-dimensional spaces with applications in biology*, Proceedings of the National Academy of Sciences of the USA **105** (2008), 3209–3214.

[CM06] G. Casella and E. Moreno, *Objective bayesian variable selection*, J. Am. Stat. Assoc. **101** (2006), 157–167.

[DCL04] Ye Ding, Chi Yu Chan, and Charles E. Lawrence, *Sfold web server for statistical folding and rational design of nucleic acids*, Nucleic Acids Res. **32** (2004), W135–W141, Web Server issue.

[DCL05] ———, *RNA secondary structure prediction by centroids in a Boltzmann weighted ensemble*, RNA **11** (2005), 1157–1166.

[DCL06] ———, *Clustering of RNA secondary structures with application to messenger RNAs*, J. Mol. Biol. **359** (2006), 554–571.

[DEKM99] Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme Mitchison, *Biological sequence analysis: probabilistic models of proteins and nucleic acids*, Cambridge University press, Cambridge, UK, 1999.

[Die05] Reinhard Diestel, *Graph theory*, Springer-Verlag, 2005.

[DSO78] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, *A model of evolutionary change in proteins*, vol. 5, pp. 345–352, National Biomedical Research Foundation, Washington, D.C., USA, 1978.

[Fel81] J. Felsenstein, *Evolutionary trees from DNA sequences: a maximum likelihood approach*, Journal of Molecular Evolution **17** (1981), 368–376.

[GCSR03] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin, *Bayesian data analysis*, second ed., Chapman and Hall/CRC, New York, USA, 2003.

[GG84] Stuart Geman and Donald Geman, *Stochastic relaxation, gibbs distributions, and the bayesian restoration of images*, IEEE Transactions on Pattern Analysis and Machine Intelligence **6** (1984), 721–741.

[Gol04] Martin Charles Golumbic, *Algorithmic graph theory and perfect graphs*, Elsevier, Boston, MA, USA, 2004.

[Goo96] Joshua Goodman, *Parsing algorithms and metrics*, Proceedings of the 34th annual meeting on ACL, 1996, pp. 177–183.

[HC68] John M. Hammersley and Peter Clifford, *Markov fields on finite graphs and lattices*, Tech. report, University of California, Berkeley, CA, USA, 1968.

[HH92] S. Henikoff and J. G. Henikoff, *Amino acid substitution matrices from protein blocks*, Proceedings of the National Academy of Sciences of the USA **89** (1992), 10915–10919.

[IR05] H. Ishwaran and J.S. Rao, *Spike and slab variable selection: Frequentist and Bayesian strategies*, Ann. of Stat. **33** (2005), 730–773.

[JOSB95] Thomas M. Jermann, Jochen G. Opitz, Joseph Stackhouse, and Steven A. Benner, *Reconstructing the evolutionary history of the artiodactyl ribonuclease superfamily*, Nature **374** (1995), 57–59.

[KA93] S. Karlin and S. F. Altschul, *Applications and statistics for multiple high-scoring segments in molecular sequences*, Proceedings of National Academy of Sciences of the USA **90** (1993), 5873–5877.

[LC03] E. L. Lehmann and George Casella, *Theory of point estimation*, Springer, New York, USA, 2003.

[LL99] Jun S. Liu and Charles E. Lawrence, *Bayesian inference on biopolymer models*, Bioinformatics **15** (1999), 38–52.

[LNL95] Jun S. Liu, Andrew F. Neuwald, and Charles E. Lawrence, *Bayesian models for multiple local sequence alignment and gibbs sampling strategies*, Journal of the American Statistical Association **90** (1995), 1156–1170.

[LS88] Steffen L. Lauritzen and David J. Spiegelhalter, *Local computations with probabilities on graphical structures and their application to expert systems*, Journal of the Royal Statistical Society, Series B **50** (1988), 157–224.

[Mat04] D. H. Mathews, *Using an RNA secondary structure partition function to determine confidence in base pairs predicted by free energy minimization*, RNA **10** (2004), 1178–1190.

[McC90] J. S. McCaskill, *The equilibrium partition function and base pair binding probabilities for RNA secondary structure*, Biopolymers **29** (1990), 1105–1119.

[MDC+04] D. H. Mathews, M. D. Disney, J. L. Childs, S. J. Schroeder, M. Zuker, and D. H. Turner, *Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure*, Proc. Natl. Acad. Sci. **101** (2004), 7287–7292.

[MGTV05] N. Sha M. G. Tadesse and M. Vannucci, *Bayesian variable selection in clustering high-dimensional data*, J. Am. Stat. Assoc. **100** (2005), 602–617.

[Miy94] Sanzo Miyazawa, *A reliable sequence alignment method based on probabilities of residue correspondences*, Protein Eng. **8** (1994), no. 10, 999–1009.

[MT90] Silvano Martello and Paolo Toth, *Knapsack problems: Algorithms and computer implementations*, John Wiley & Sons, 1990.

[NPGK78] R. Nussinov, G. Pieczenik, J. R. Griggs, and D. J. Kleitman, *Algorithms for loop matchings*, SIAM Journal of Applied Mathematics **35** (1978), 68–82.

[NW70] S. B. Needleman and C. D. Wunsch, *A general method applicable to the search for similarities in the amino acid sequence of two proteins*, Journal of Molecular Biology **48** (1970), 443–453.

[PS98] Christos H. Papadimitriou and Ken Steiglitz, *Combinatorial optimization: algorithms and complexity*, Dover, 1998.

[Rab93] Lawrence R. Rabiner, *Fundamentals of speech recognition*, Prentice Hall, Englewood Cliffs, NJ, USA, 1993.

[San72] D. Sankoff, *Matching sequences under deletion/insertion constraints*, Proceedings of the National Academy of Sciences of the USA **69** (1972), 4–6.

[Sch95] Dolph Schluter, *Uncertainty in ancient phylogenies*, Nature **377** (1995), 108–109.

[SF07] M. Smith and L. Fahrmeir, *Spatial bayesian variable selection with application to functional magnetic resonance imaging*, J. Am. Stat. Assoc. **102** (2007), 417–431.

[SW81] T. F. Smith and M. S. Waterman, *Identification of common molecular subsequences*, Journal of Molecular Biology **147** (1981), 195–197.

[WLL02] Bobbie-Jo M. Webb, Jun S. Liu, and Charles E. Lawrence, *BALSA: Bayesian algorithm for local sequence alignment*, Nucleic Acids Res. **30** (2002), no. 5, 1268–1277.

[ZLL98] Jun Zhu, Jun S. Liu, and Charles E. Lawrence, *Bayesian adaptive sequence alignment algorithms*, Bioinformatics **14** (1998), 25–39.

[Zuk03] Michael Zuker, *Mfold web server for nucleic acid folding and hybridization prediction*, Nucleic Acids Res. **31** (2003), no. 13, 3406–3415.