Sum-Product Algorithms for the Analysis of Genetic Risks

Mikko Koivisto

To be presented, with the permission of the Faculty of Science of the University of Helsinki, for public criticism in Auditorium III, Porthania, on January 30th, 2004, at 12 o’clock noon.
Contact information

Postal address:
   Department of Computer Science
   P.O. Box 26 (Teollisuuskatu 23)
   FIN-00014 University of Helsinki
   Finland

Email address: postmaster@cs.Helsinki.FI

URL: http://www.cs.Helsinki.FI/

Telephone: +358 9 1911

Telefax: +358 9 191 44441
Sum-Product Algorithms for the Analysis of Genetic Risks

Mikko Koivisto

Department of Computer Science
P.O. Box 26, FIN-00014 University of Helsinki, Finland

Helsinki, January 2004, 155 pages
ISSN 1238-8645
ISBN 952-10-1577-2 (paperback)
ISBN 952-10-1578-0 (PDF)

Abstract

This work is motivated by a genetic data analysis task: recurrence risk analysis. Recurrence risks in various types of relatives, e.g., offspring and siblings, characterize the inheritance pattern of a genetic disorder. Given observations of the population prevalence and different recurrence risks, the task is to infer the number of underlying genes and the frequencies and effects of different variants of the genes. Due to a complex relationship of relatively simple data and a large number of model parameters, this problem is challenging both statistically and computationally. Straightforward application of existing techniques is not sufficient.

In the first part of this thesis, we study three general methodological issues. First, we review the Bayesian paradigm for statistical inference. Special emphasis is on certain fundamental difficulties of practical Bayesian methods. Second, we study the sum-product problem, that is, marginalization of a multidimensional function that factorizes into a product of low-dimensional functions. We introduce a novel algorithm that improves the well-known variable elimination method by employing fast matrix multiplication techniques. A special type of sum-product problems, called transformation problems, are studied. We generalize a technique known as the Yates algorithm and show how the Möbius transformation on a subset lattice can be computed efficiently. Third, we consider the problem of integrating a multidimensional function. We describe a sophisticated method based on a tempering technique and Metropolis Coupled Markov chain Monte Carlo simulation. We argue that this method is particularly suitable to computa-
tions required in Bayesian data analysis. Connections to related methods are also explored.

The second part is devoted to the genetic application. We introduce a genetic model called an epistatic Mendelian model. For computing recurrence risks under a fully specified model, we present a method that employs the Yates algorithm. Based on a different problem representation, we also give another algorithm that uses the fast Möbius transform. To integrate over the model parameters we present a version of the Metropolis-coupled Markov chain Monte Carlo method. Finally, we report experimental results that support two main conclusions. First, Bayesian analysis of recurrence risks is computationally feasible. Second, recurrence risk data provides interesting information concerning competitive genetic hypotheses, but the amount of information varies depending on the data set.

Computing Reviews (1998) Categories and Subject Descriptors:
F.2.1 Analysis of Algorithms and Problem Complexity: Numerical Algorithms and Problems
F.2.2 Analysis of Algorithms and Problem Complexity: Nonnumerical Algorithms and Problems
G.3 Probability and Statistics: Multivariate statistics, Probabilistic algorithms
J.3 Life and Medical Sciences: Biology and genetics

General Terms:
Algorithms, Experimentation, Theory

Additional Key Words and Phrases:
Bayesianism, Genetic model, Markov chain Monte Carlo, Möbius transform, Recurrence risk, Variable elimination, Yates algorithm
Acknowledgements

I would like to thank everybody who was directly or indirectly involved in the process of writing this thesis. First of all, I am grateful to my advisor, Prof. Heikki Mannila, for encouragement, guidance, and patience. His broad scientific experience and interest, as well as trust in my personal choices, have been invaluable. I would also like to thank Prof. Juha Kere for the collaboration that motivated this work. I highly appreciate communications with Prof. Esko Ukkonen, Prof. Elja Arjas, Prof. Henry Tirri, and Prof. Hannu Toivonen, which have broadened my understanding on things. The insightful comments from Prof. Pekka Orponen and Prof. Petri Myllymäki have helped me to improve this thesis.

When I was taking the first steps in statistical genetics, I was mentored by Dr. Vesa Ollikainen, Dr. Päivi Onkamo, and Dr. Mikko Sillanpää, for which I am grateful. On many aspects of Bayesianism, I have benefited from inspiring discussions with Teemu Roos. I would like to thank Kismat Sood, not only for commenting the manuscript, but also for our every-day chat on various matters. I am indebted to many friends and colleagues who have shared with me many of my problems, troubles, and concerns; Riika Kilpihari, Petteri Sevon, Kari Vasko, to name a few.

The Department of Computer Science at the University of Helsinki, currently headed by Prof. Jukka Paakki, has provided me with excellent working conditions. Financial support from the Helsinki Graduate School of Computer Science and Engineering (HeCSE) is gratefully acknowledged.

To my parents, Pirkko and Mauno, I would like to express my gratitude for their love and constant interest in the state of my studies. I also would like to thank my brother, Juhani, and my sisters, Anuikka and Marjukka, for reminding me that there are also other fascinating things in the world.

My deepest thanks to my wife, Heli, and my daughter, Ella, for being there.

Helsinki, December 2003
Mikko Koivisto
Contents

1 Introduction ........................................ 1

2 Bayesian modeling .................................. 7
   2.1 Pursuing good decisions ....................... 8
   2.2 Bayesian probabilities, models, and induction .. 10
   2.3 Practical versus ideal Bayesian modeling ... 17
   2.4 Discussion .................................. 22

3 Sum-products and the elimination algorithm ...... 23
   3.1 The problem of marginalizing a product ....... 23
   3.2 The variable elimination algorithm .......... 30
   3.3 Boosting the variable elimination algorithm ... 36
   3.4 Transformation problems ..................... 41
   3.5 On the limitations of the variable elimination algorithm . 50
   3.6 Concluding remarks .......................... 54

4 Integration by tempered importance sampling .... 57
   4.1 Integration problem .......................... 58
   4.2 Monte Carlo importance sampling ............ 59
   4.3 Tempered integration ........................ 62
   4.4 Metropolis-coupled Markov chain Monte Carlo ... 65
   4.5 Related methods ............................ 68
   4.6 Concluding remarks .......................... 72

5 Population risks and genetic models ............. 73
   5.1 Population risks of binary traits ............ 73
   5.2 Sample-based risk statistics ................. 76
   5.3 General genetic model ....................... 77
   5.4 Epistatic Mendelian model .................... 80
      5.4.1 Ingredients of genotypes ............... 80
      5.4.2 Genotype frequencies ................... 81
5.4.3 Transmission model ........................................ 82
5.4.4 Relation of genotypes and phenotypes ................. 83
5.4.5 Summary ........................................................ 84
5.5 Direct and inverse inference problem ..................... 85

6 Computation of population risks .............................. 87
6.1 The problem ....................................................... 87
  6.1.1 Relationship types: coefficients of relatedness .... 88
  6.1.2 The problem of computing a recurrence risk ..... 92
  6.1.3 Why are advanced methods needed? ................. 93
  6.1.4 Preliminary observations ................................. 94
6.2 Matrix decomposition and the Yates algorithm ........... 96
  6.2.1 Quadratic form ............................................. 96
  6.2.2 Locus-wise decomposition ............................... 98
  6.2.3 Computation of locus-wise matrices ................. 99
  6.2.4 Applying the Yates algorithm ......................... 102
6.3 Variance decomposition and fast Möbius transform ....... 103
  6.3.1 Genotypic value as a function of multiple variables 104
  6.3.2 Orthogonalization and decomposition of variance . 105
  6.3.3 Computation of variance components ................. 108
  6.3.4 Covariance representation of recurrence risks .... 112
  6.3.5 Computation of recurrence risks ...................... 113
6.4 An inequality for recurrence risks ......................... 121
6.5 Concluding remarks ........................................... 124

7 Exploration of genetic models by sampling .................. 127
7.1 Data and model revisited .................................... 127
7.2 Implementing tempered Markov chain Monte Carlo ....... 129
7.3 Experiments on synthetic and real data ................... 132
  7.3.1 An analysis on small data sets ....................... 132
  7.3.2 A simulation study ....................................... 134
  7.3.3 Selected synthetic and real data sets ............... 136
7.4 Conclusions ..................................................... 141

8 Discussion .......................................................... 145

References ........................................................... 147
Chapter 1

Introduction

Scientific data analysis aims at updating knowledge via learning from observations. The goal is often the validation of a theory or comparison of competitive hypotheses. In Hume's sense, however, this type of inductive reasoning cannot be justified solely by logic. Unavoidably, one has to also deal with speculative elements, whose structure and nature are specified by a scientific paradigm. A paradigm determines the criteria according to which one selects and defines problems for enquiry, and defines the rules of rational reasoning to approach the problems. The role of data analysis is to follow such general principles and postulates in a rigorous manner, usually in some restricted context, say, in the prediction of tomorrow's weather based on recent measurements. Thus, scientific data analysis ought to combine domain specific, often informal, knowledge representations, and formal, exact methods, developed mainly in mathematics, statistics, and computer science.

This work is motivated by a specific genetic data analysis task: recurrence risk analysis. The problem setting is relatively simple. We consider a certain trait, say disease, and are interested in the genetic nature of its inheritance. We assume that as a result of an epidemiological study we have some observations concerning certain population risk characteristics of the trait. In addition to the population prevalence, i.e., the general risk, this type of data may include estimates for various recurrence risks, i.e., frequencies with which the trait occurs among the various types of relatives of affected individuals, e.g., offspring, siblings, and cousins. By analyzing such risk data we may obtain information about the number of genes responsible for the trait as well as about the frequencies and effects of different variants of the genes. Such information has its direct, inductive use in genetic counseling. But also, knowledge about the plausibility of different genetic hypotheses may help in designing more complex and expensive further stud-
yses like linkage, association, and gene expression analysis. The analysis of recurrence risk data serves as a preliminary step in the sequential genetic analysis that aims at positional cloning of the genes, understanding the etiology of the trait, and perhaps ultimately, at drug discovery.

The analysis of recurrence risks is methodologically challenging. First we need a paradigm to approach the following loosely stated question:

What can be said about the plausibility of different genetic hypotheses given observations about recurrence risks?

In this thesis we will follow the Bayesian paradigm for statistical modeling and inference. Accordingly, any answer to the above problem is supposed to take the form of a probability distribution. This poses two types of challenges. One is the building of a probability model, which essentially is a problem of genetics and statistics. We have to extract the relevant quantities subject to modeling, and study the dependencies of these quantities. Fortunately, in these issues we can borrow from the existing literature of statistical genetics, e.g., [Fis18, Kem57, ES71, Bul80]. What has not been studied so well is the amount of information that one can obtain from recurrence risk data. This lack of knowledge is understandable as most of the related research has been carried out under the classical statistical paradigm that cannot directly answer this type of questions. The other challenge is the computational handling of the model, which essentially is an algorithmic issue. Basically, the question is about automatization of probability calculus under a specified probability model. This involves computational subtasks that can be solved exactly as well as tasks where we have to resort to approximation methods. Both types of problems are nontrivial—sophisticated algorithms are needed.

The methodological questions arising from the genetic application are generic by nature, and thus interesting and important also in their own right. Acknowledging that, the focus of this work is actually not on the specific genetic data analysis task, but rather on a few general themes that might have useful applications beyond what is described in this thesis. Accordingly, the first part of the thesis contributes to the general methodological aspects. The second half is then devoted to the specific genetic data analysis problem.

We start in Chapter 2 by reviewing the Bayesian paradigm with two major goals. The primary goal is to describe some appealing characteristics of the Bayesian paradigm, and thus support the Bayesian choice in the genetic data analysis application. In particular, we focus on questions of model complexity and identifiability that, being notoriously uncomfortable
for classical methods, most transparently show the elegance and applicability of the Bayesian methods. Unfortunately, when trying to apply the Bayesian paradigm in practice one often faces severe difficulties, including the building of an appropriate probability model and handling computationally infeasible expressions. It is a secondary goal of Chapter 2 to contribute to the discussion of these problematic issues that a practitioner cannot neglect.

Inherent in Bayesian inference is a well-known computational problem: marginalization of a function of many variables. When the distribution decomposes into a product of low dimensional functions, as often is the case, the problem shows a combinatorial aspect. Namely, the computational complexity of the marginalization problem can be expressed via the properties of an appropriately defined graph. Dynamic programming type algorithms can be designed to exploit the graph structure. The topic has been studied extensively in the context of Bayesian networks (for different aspects, see e.g., [Pea88, Coo90, JLO90, DL93, BG01], and references therein). Interestingly, a great deal of related work had been done already before in the specific context of genetic pedigree analysis [ES71, LE75, CTS78, LB83, LG87]. It was not until about a decade ago that people recognized that the marginalization algorithms are in fact applicable also to other pairs of operations than the conventional addition "+" and multiplication "·", provided that certain algebraic properties are satisfied [Dec96, SH96, LJ97, BMR97, AM00, KFL01]. Thereby, the marginalization problem, also called the sum-product problem [KFL01], in its generalized form provides a deep connection to certain theoretical questions of computing [SH96, Ste03].

We consider the exact computation of multidimensional summations in Chapter 3. First we review the basic algebraic problem formulation and describe a dynamic programming method known as the variable elimination algorithm. We then consider the suboptimality of the elimination algorithm and point out that in many cases the algorithm can be accelerated by fast matrix multiplication methods [Str69, Pan84, CW90]. For example, extending the techniques of Boolean matrix multiplication, non-trivial applications of fast matrix multiplication are found in the all-pairs-shortest-paths problem [AGM97, Zwi02]. We contribute by considering certain more generic and somewhat complementary settings where the elimination algorithm can be boosted by fast matrix multiplication. While these findings may be theoretically interesting, the resulting algorithms are not practical. As a topic more tightly connected to the recurrence risk analysis task, we introduce a subclass of sum-product problems called transformation problems. Examples of transformation problems include the Fourier transform,
Yates transform, and certain types of Möbius transforms, of which the latter two will be adapted and tuned in the application context in Chapter 6. In Chapter 3, we approach transformation problems from the general and more abstract perspective. The goal of the chapter is to emphasize the role of finding an appropriate sum-product formulation in the first place, whereas, once a formulation is found and fixed, the variable elimination method is almost trivial to apply as such. A minor contribution, related to this, is an improvement on Kontkanen’s et al. [KBM+03] recent algorithm for exact computation of the normalizing maximum likelihood function for the multinomial distribution.

Computing marginal functions exactly is not always feasible. Monte Carlo techniques for approximate evaluation of multidimensional integrals are described in Chapter 4. The idea of Monte Carlo methods is simple: to approximate a population average (i.e., the integral) by a sample average. However, drawing a representative sample from a large-dimensional space can be extremely difficult. Therefore, after introducing the basic Monte Carlo method, we quickly move to a sophisticated technique that we call the tempered integration method. The basic idea in tempered integration is to decompose the difficult target integral into a sequence of easier integrals. The idea has occurred in, e.g., [Nea93, GM98].

Chapter 5 starts the application specific part of the thesis. We first consider the nature of recurrence risk data. Then we introduce a rather expressive class of genetic hypotheses, the epistatic Mendelian model, which has been extensively studied in both simplified and generalized forms in the literature [Kem57, ES71, Bul80]. In Chapter 5 we also separate two computational tasks: the direct problem is to compute theoretical recurrence risks under a specified genetic model; the inversion problem is to compute summaries about the genetic models given recurrence risk data. The main technical contribution of the thesis is Chapter 6, where algorithms for the direct problem, the computation of recurrence risks, are developed and analyzed. We argue that the existing methods designed for handling large pedigrees are not optimal. Fast algorithms are derived by applying the general Yates and Möbius transforms introduced in Chapter 3. Based on the mathematics developed we also show an inequality stating that under the epistatic Mendelian model, the offspring risk is always at least as high as the sibling risk; this is a revision of the version published in [KM01]. Finally, in Chapter 7, we consider the actual data analysis task, the inversion problem. Tempered integration techniques described in Chapter 4 are employed to explore the space of genetic hypotheses. By experiments on synthetic and selected real data sets we validate the methodology and
implementation. The results serve also as evidence for the conclusion that the Bayesian paradigm is applicable for updating knowledge.

A summarizing discussion of the general methodological themes and their application in recurrence risk data analysis is provided in Chapter 8.
Chapter 2
Bayesian modeling

Finding descriptive and normative rules for rational behavior is an attractive and important goal of many scientific branches, including philosophy, psychology, cognitive science, mathematics, statistics, and computer science. No consensus has been reached so far, and different “rational systems” have been derived from different principles and postulates.

One approach is Bayesian decision theory, which formulates rational behavior as the maximization of subjective expected utility. The Bayesian way of combining personal preferences and beliefs has turned out to be suitable in numerous contexts in artificial intelligence and statistics. Despite its generality and elegance, the Bayesian paradigm is somewhat controversial. One reason for this is perhaps the relativism that arises unavoidably from the subjectivity—the paradigm does not aim at “scientific objectivity” in any absolute sense. Another reason may be certain practical difficulties concerning the building and handling of complex models.

In this chapter, we describe the Bayesian approach with special emphasis on statistical modeling and inference. Throughout the chapter Bayesian methods are occasionally contrasted with frequentist and information theoretic methods. We start, in Section 2.1, by briefly recalling the decision theoretical foundations of Bayesian reasoning under uncertainty. In Section 2.2 we continue by discussing Bayesian probability models. In particular, we discuss the concepts of complexity, identifiability, and power of models, that are especially relevant in data analysis. Section 2.3 addresses some problems of practical Bayesian modeling. Finally, Section 2.4 summarizes the chapter.

Connections to the rest of this work are wide. Chapter 5 considers genetic models from the Bayesian perspective. Studies of computing marginals, in chapters 3 and 4, are motivated by Bayesian inference. Also, the experiments, described in Chapter 7, follow the Bayesian paradigm.
2.1 Pursuing good decisions

Many real-life problems can be naturally formulated as decision problems. In a decision problem a subject has a set of possibilities actions that she can, or has to, choose from. The objective of the subject is to choose one of the actions that are most beneficial to her. There are a number of slightly different but essentially equivalent more precise formulations of a decision problem, of which we present one.

We let \( \mathcal{A} \) denote the set of possible actions, \( \mathcal{W} \) the set of possible states of the world, or simply worlds, in which the subject makes her decision, and \( U \) a utility function that is a mapping from \( \mathcal{W} \times \mathcal{A} \) to the real numbers. Now the decision problem in a world \( w \in \mathcal{W} \) is to choose an action \( a \in \mathcal{A} \) that maximizes \( U(w,a) \). It is usually supposed that the subject knows the set \( \mathcal{A} \) and the function \( U \). So, the nature of the problem depends on what the subject knows about \( w \). Most importantly, the decision problem, as described above, is a well-defined optimization problem if and only if the world \( w \) is known. Otherwise the problem is, in general, ill-defined, for the utility depends on an unknown entity, \( w \). In that case, one must introduce additional postulates and principles. Different approaches can be seen as definitions of a marginal utility. Three competitive principles are summarized below.

The principle of worst-case utility says that the subject should choose an action that maximizes the utility in the worst possible scenario. Thus, an optimal action maximizes

\[
U_W(a) = \min_{w \in \mathcal{W}} U(w,a).
\]

An obvious problem of the principle is that it may lead to very conservative decisions. Usually this happens if \( \mathcal{W} \) is taken to be “too large” so that it includes a world \( w \) that is logically possible yet highly unlikely. Despite this drawback, once the set \( \mathcal{W} \) is fixed, the decision problem becomes well defined under the worst-case principle. The worst-case principle is often followed in the complexity analysis of algorithms (e.g., [AHU74]).

The principle of expected utility says that the subject should choose an action that maximizes the expected utility under a suitably specified probability distribution on \( \mathcal{W} \). Thus, an optimal action maximizes

\[
U_p(a) = \sum_{w \in \mathcal{W}} p(w)U(w,a),
\]

where \( p(w) \) is the probability of \( w \). (Note that here the present world is assumed to be independent of the action to be chosen.) A problem of this
2.1 Pursuing good decisions

principle is that the specification of the probability distribution may be difficult and even philosophically questionable—what is the interpretation of the probability? In any case, if the distribution $p$ is given (for arbitrary large $\mathbb{W}$), then the decision problem is well defined under the principle of expected utility. In the context of algorithm analysis, the analogous approach is called average-case analysis (e.g., [AHU74]).

*The principle of worst-case relative utility* says that the subject should choose an action that maximizes the worst-case *gain*. We define the gain as the difference of the obtained utility $U(w, a)$ and the maximum utility that one could obtain if one knew $w$ and was allowed to choose from a set $\mathbb{B}$ of actions. Thus, an optimal action maximizes

$$U_B(a) = \min_{w \in \mathbb{W}} \{U(w, a) - \max_{b \in \mathbb{B}} U(w, b)\}.$$  

(We note this principle can be also formulated as minimization of the worst-case *regret*, where the regret is defined as the difference $(\max_{b \in \mathbb{B}} U(w, b)) - U(w, a)$, i.e., the negative of the gain.) An advantage of this principle is that the set $\mathbb{W}$ may well be very large including unlikely worlds, yet no probability distribution is required. The cost of this comfortability is the requirement for a reasonable specification of the set $\mathbb{B}$. Putting $\mathbb{B} = \mathbb{A}$ would agree with a convenient, and perhaps the most natural, game-theoretic interpretation. However, this would usually yield a too large set of competitors, and hence, a trivial or useless decision rule. Consequently, usually $\mathbb{B}$ is taken to be a proper subset of $\mathbb{A}$. The best known applications of the principle of worst-case relative utility are perhaps the ones studied in the context of online or sequential prediction (see e.g. [HKW98], and references therein). A simple, yet technical, setting is described in the next example, where we briefly compare the above described three principles.

**Example 2.1** Let us consider sequential probabilistic prediction of binary values against logarithmic loss. We let $\mathbb{W} = \{0, 1\}^n$ be the set of possible binary sequences of length $n$. An action is a probabilistic strategy from the set $\mathbb{A} = \{a : a$ is a probability distribution on $\mathbb{W}\}$. The utility of a strategy $a \in \mathbb{A}$ for a sequence $w = (w_1, \ldots, w_n) \in \mathbb{W}$ is defined as $U(w, a) = \log_2 a(w) = \sum_{t=1}^n \log_2 a(w_t | w_1, \ldots, w_{t-1})$. For a sequence $(w_1, \ldots, w_t)$ we let $s_t$ denote the number of 1's in it; for $t = n$ we may drop the subscript.

Different principles lead to different optimal actions. It can be shown that according to the worst-case principle the uniform distribution on $\mathbb{W}$ is the unique optimal choice.

To consider the worst-case relative utility we need to specify a subset of $\mathbb{A}$. We let $\mathbb{B}$ be the set of Bernoulli distributions on $\mathbb{W}$. That is, for
each $b \in \mathbb{B}$ we have $b(w) = \theta^s(1 - \theta)^{n-s}$ for some $\theta$ specific to $b$. It can be shown (originally by Shtarkov, 1987; see, e.g., [CBL01]) that the worst-case relative utility is uniquely minimized by the normalized maximum likelihood distribution $a(w)$ defined by

$$a(w) = \frac{(s/n)^s(1 - s/n)^{n-s}}{\sum_{w' \in \mathcal{W}} (s'/n)^{s'}(1 - s'/n)^{n-s'}},$$

where $s'$ is the number of 1's in $w'$. Note that the optimal choice is not a member of the set $\mathbb{B}$.

To consider the expected utility we must specify a probability distribution. We let $p$ be the distribution that satisfies $p(w_t \mid w_1, \ldots, w_{t-1}) = (1 + s_{t-1})/(1 + t)$. This $p$ is the optimal strategy under the principle of expected utility. This choice of $p$ is known as Laplace’s rule of succession. It can be obtained as a mixture of Bernoulli distributions.

In what follows, we operate under the principle of expected utility. There are foundational arguments that favor the principle of expected utility over the alternatives. It can be shown that certain intuitively appealing axioms of rational decision making unavoidably lead to the conclusion that a rational decision maximizes the expected utility for some probability measure (see, e.g., [BS94, Ber03]). This, in fact, says that all uncertainty about the possible states of the world should be represented by a probability distribution, thus forcing the Bayesian paradigm, that will be reviewed in the next sections.

2.2 Bayesian probabilities, models, and induction

The principle of expected utility assumes that the subject can assign a probability distribution on the set of possible worlds. How such an assignment could and should be made depends on the interpretation of the concept of probability. Also, the semantics of the principle of expected utility remains unclear until that interpretation is fixed.

Usually two different interpretations of probability are distinguished: the classical, “objective” one and the Bayesian, “subjective” one. In the classical interpretation the probability of an event is taken to be the limiting relative frequency of occurrences of the event under a suitably specified infinite random trial.\footnote{The idea is that there is a probability measure (perhaps unknown) under which the relative frequencies tend to a limiting value almost surely. This limiting value is the probability. As we see, this is a sort of circular definition since the underlying probability measure as a real-world entity is assumed to have some interpretation already.} Under this interpretation probabilities are usually
considered as “objective”, real entities of the world that are independent of
the subject. This is in sharp contrast to the Bayesian subjective interpre-
tation. Under the Bayesian interpretation probabilities can be assigned to
any meaningful statement about the world. The probability of a statement
measures a subject’s uncertainty, or degree of belief, about the truth of the
statement. (Ultimately, the semantics of “meaningful” and “truth” are, of
course, also subjective, being dependent on the subject’s ontological and
epistemological views.)

Mathematically Bayesian probability calculus equals probability theory.
Sometimes Bayesian probability calculus is defined as an extension to pro-
positional logic [Jay03]. Consequently, probabilities are assigned to state-
ments, and the basic rules of probability are derived from a set of reasonable
axioms (e.g., the desiderata of Jaynes [Jay03]). In this sense the mathe-
matical formulation of Bayesian probability calculus differs significantly
from the set and measure theoretic formulation of Kolmogorov that is used as
the standard of mathematical probability theory. Interestingly, but perhaps
not surprisingly, both axiomatizations can be shown to yield essentially the
same mathematical system. However, there are also minor differences, per-
haps one of the most important being Jaynes’s finite set policy [Jay03]: Do
all modeling and calculations with finite objects and do not consider limits
until finally.

Bayesian modeling is about specifying a subjective probability for all
statements relevant to the decision problem in question. It is often con-
venient to express statements via random variables. Henceforth we let
$x_1, \ldots, x_n$ be random quantities that take values from domains $A_1, \ldots, A_n$.

We will apply indexing by subsets in a generic way. For example, if $S = \{s_1, \ldots, s_k\}$
with $s_1 < \cdots < s_k$, then $x_S$ denotes the vector $(x_{s_1}, \ldots, x_{s_k})$
that is an element of $A_S = A_{s_1} \times \cdots \times A_{s_k}$. We call any joint distri-
bution over the variables a Bayesian model. All probability functions will be
denoted by $p$ in a generic manner. For example, $p(x_n \mid x_1)$ denotes the
conditional probability of $x_n$ given $x_1$; this probability is obtained from the
joint model $p(x_1, \ldots, x_n)$ by conditioning and marginalization. In Bayesian
statistical modeling, different variables usually play different roles. Some of
the variables are observable while some others are latent (or, unobservable,
hidden, missing). Also, it may be that only some of the variables are actu-
ally relevant in the decision problem while the rest are treated as ancillary
(or nuisance) variables that help in specifying the probabilities but are not
of primary interest. Sometimes it is also technically advisable to introduce
variables that are difficult to interpret as entities of the real world. An
example of such a variable is the bias of a coin in the usual coin tossing
context; see Example 2.2 below.

A popular way to build a Bayesian model is to "extend" a frequentist model. In a frequentist model some of the quantities $x_1, \ldots, x_n$ (that a Bayesian treats as random variables) are treated as fixed parameters, and hence not as random variables, while the others are considered outcomes of a "truly random" process, and hence random variables. Disregarding the difference in the probability interpretation, frequentist models of random variables for given parameters are often principal components of Bayesian models. To clarify terminology we call the frequentist variables observables, though, in general, some (frequentist) variables may well be latent (unobservable). For a while we denote the parameters and the observables by $\theta$ and $y$, and the associated domains by $\Theta$ and $\mathcal{Y}$, respectively. For simplicity suppose that $\theta$ and $y$ form a partition of the (Bayesian) variables. A frequentist model specifies the probability distribution of the observables for any fixed parameter values. In Bayesian notation this corresponds to the set $\{p(y \mid \theta) : \theta \in \Theta\}$. For a fixed $y$ the mapping $p(y \mid \theta)$ is usually called a likelihood function of $\theta$. We will use term likelihood model for this type of conditional distribution. A full Bayesian model is obtained by specifying a distribution of the parameters and putting simply

$$p(x_1, \ldots, x_n) = p(\theta, y) = p(\theta)p(y \mid \theta).$$

In many statistical settings the parameters $\theta$ are relevant but latent while the observations $y$ are observable but ancillary. In Bayesian methods the interest is then in the distribution of $\theta$. Bayes's theorem tells how to update the prior $p(\theta)$ to the posterior $p(\theta \mid y)$ via the likelihood $p(y \mid \theta)$:

$$p(\theta \mid y) = \frac{p(\theta)p(y \mid \theta)}{\int_{\Theta} p(\theta)p(y \mid \theta) d\theta}.$$  

Mathematically, Bayes’ theorem states nothing but the definition of conditional probability. Its significance lies in the way it relates the concepts of causality, learning, and induction. Namely, the likelihood part usually expresses the causal order: often $\theta$ may be thought of as a cause of $y$. For example, $\theta$ may parametrize a “random generating process” that results in $y$. Learning is present in that the prior beliefs about $\theta$ are updated based on an “example” $y$. Induction will take place once the subject considers a new instance $y'$ that is in some sense similar to $y$; this idea will become clearer soon.

A slightly different perspective on the prior–likelihood decomposition is expressed by the celebrated representation theorem, due to de Finetti (see, e.g., [Ber96] and references therein). The result is purely mathematical
but it gives a deep insight into the relationship of Bayesian and frequentist models. To state a basic version of the representation theorem we need the concept of exchangeability. A finite sequence $y_1, y_2, \ldots, y_k$ of variables is exchangeable under a probability distribution $p$ if their joint distribution is invariant under permutations. An infinite sequence of variables is exchangeable if all of its finite subsequences is exchangeable. For example, any sequence of independent random variables is obviously exchangeable.

The representation theorem states that if $y_1, y_2, \ldots$ is an infinite, exchangeable sequence of variables, then for any $k$,

$$p(y_1, y_2, \ldots, y_k) = \int_{\Theta} p(\theta) \prod_{i=1}^{k} p(y_i \mid \theta) d\theta,$$

for some $\Theta$ and some joint distribution $p(y_i, \theta) = p(y_i \mid \theta)p(\theta)$ that is the same for all variables $y_i$. In words, the representation theorem says that there is a prior distribution for additional variables $\theta$ and a likelihood model. Note that the representation theorem is an existence result. It generally does not specify the probability model. However, it encourages the modeler to consider augmenting the set of variables and the probability model as $p(\theta, y_1, \ldots, y_k) = p(\theta) \prod_{i=1}^{k} p(y_i \mid \theta)$. The concept of exchangeability formulates a Bayesian counterpart for the frequentist i.i.d. (i.e., independently and identically distributed) assumption. Note that calling variables $y_1, \ldots, y_k$ i.i.d. would be somewhat ridiculous when trying to predict, say $y_k$ after seeing $y_1, \ldots, y_{k-1}$.

**Example 2.2** Let us consider sequential prediction of binary valued variables $y_1, y_2, \ldots$. Suppose that these variable are outcomes from a coin tossing experiment. Then it is natural (though by no means necessary) to judge that the sequence is infinitely exchangeable under the subjective model $p$. The representation theorem now states that for any $k$

$$p(y_1, y_2, \ldots, y_k) = \int_{0}^{1} p(\theta)\theta^{s_k}(1 - \theta)^{k-s_k} d\theta,$$

where $s_k = y_1 + \cdots + y_k$ and $p(\theta)$ some prior distribution. Furthermore, the “new” variable $\theta$ is defined independently of $k$ and $p$ as the limit $\lim_{n \to \infty} s_n/n$ (see, e.g., [Ber03]). When predicting $y_k$ given the earlier outcomes, the principle of expected utility suggests to consider the posterior distribution, which by Bayes’ theorem can be expressed as

$$p(y_k \mid y_1, \ldots, y_{k-1}) = \int_{0}^{1} p(\theta \mid y_1, \ldots, y_{k-1})\theta^{y_k}(1 - \theta)^{1-y_k} d\theta.$$
Thus, learning about \( \theta \) is here the base of induction. Recall from Example 2.1 that for a particular choice of the prior \( p(\theta) \) we get Laplace’s rule of succession.

Due to some features of the representation theorem it may be difficult to adopt this result as a starting point of subjective Bayesian modeling. First, one is supposed to be able and willing to imagine an infinite sequence of similar variables, which is inconvenient if one is actually modeling a finite sequence, say 10 tosses of a coin. Not surprisingly, there are results [DF80] that say that if the sequence can be embedded into a much larger finite, exchangeable sequence, then no important distortion will occur in quantifying uncertainties when assuming infinite exchangeability. Unfortunately the assumption of a much larger covering sequence is crucial for obtaining satisfactory approximations [DF80]. Second, the theorem implies existence of a variable—for concreteness consider the bias \( \theta \) of a coin—whose semantics is somewhat unclear. Although \( \theta \) is said to be defined as a limiting value of a function of the variables \( y_i \) [Ber03], such a limit does not always exist (though a limit may exist in a \( p \)-almost sure sense). Even in the case when the limit exists it may be difficult to think of \( \theta \) as an (unknown) real-world entity that, in principle, could be observed somehow. That said, we may regard the representation theorem as a purely mathematical result that loosely supports subjective Bayesian modeling.

It is an advantage of the Bayesian paradigm for statistical inference that the model can be arbitrarily complex. It does not matter how many parameters are included in the model, provided that one can assign a reasonable (“informative”) prior on them. The opposite is the case with frequentist approaches that rely on point estimators and asymptotics. In order to get sensible inferential results by frequentist methods one is typically forced to use overly simple models. Sometimes a complex (likelihood) model may be overparametrized in the sense that different parameter values yield the same distribution of the observable variables. In that case the parameters, and the model, are said to be unidentifiable. The lack of identifiability has dramatic consequences in the popular maximum likelihood procedure. In contrast, unidentifiability causes no real difficulty in the Bayesian approach. This is because the posterior distribution will anyway correctly represent the posterior uncertainty about the parameters. Note also that unidentifiability does not assert that there is no Bayesian learning. That is, it does not imply that the posterior will equal the prior (see, e.g., [GS99]). However, it is true that unidentifiability affects some aspects of learning. Especially, if the prior probabilities (or densities) for two different values of unidentifiable parameters are equal, then also their posterior probabilities
(or densities) will be equal. We may conclude by saying that the Bayesian paradigm encourages the modeler to faithfully express her beliefs, and to be open minded regarding the possible states of the world.

In the sequel we will not separate “parameters” and “variables”. All unknown quantities are treated as random quantities. The focus is on the quantities of interest, $x_R$, and on the quantities that the modeler expects to be able to observe, $x_S$. Here $R$ are $S$ refer to two subsets of $\{1, \ldots, n\}$. Note that we do not need to assume that $R$ and $S$ are disjoint nor that their union covers all the quantities included in the model.

How much information can one obtain from data? How useful is the model? These questions are related to the notion of expected utility as we will see below. Consider a decision problem with a set $\mathcal{A}$ of possible actions. Suppose that $x_R$ includes the quantities relevant to the utility, so that $U(x_R, a)$ is the utility of choosing $a \in \mathcal{A}$ when $x_R$ characterizes the state of the world. Suppose further that the decision problem takes place after the subject has observed values for another set of variables $x_S$. According to the principle of expected utility, the optimal action is then given by

$$\hat{a}(x_S) = \arg \max_{a \in \mathcal{A}} \sum_{x_R \in \mathcal{A}_R} p(x_R \mid x_S) U(x_R, a).$$

In order to consider the expected consequences of using a model, we define the expected posterior utility in terms of a utility function, a probability model, and a set of observable variables, as

$$U_p(S) = \sum_{x \in \mathcal{A}} p(x) U(x_R, \hat{a}(x_S)).$$

The expected posterior utility is a flexible measure of properties of a model. It can be employed for different purposes with different choices of the action set and the utility function.

In particular, being a subjective measure of preference, the utility may well be defined in terms of personal probabilities. When designing an experiment one may want to maximize the expected information gain [Lin56, Ber79a]. This is seen to be a special case of maximizing the expected posterior utility when we let $\mathcal{A}$ be the set of probability distributions on $\mathcal{A}_R$ and put $U(x_R, a) = \log_2 a(x_R)$, the logarithmic utility function.\(^2\) Then, by

\(^2\)Actually, the information gain is defined as the relative entropy, given by $\sum_{x_R} p(x_R \mid x_S) \log_2 [p(x_R \mid x_S)/p(x_R)]$, so that the straightforward utility function would be $U(x_R, a) = \log_2 [a(x_R)/p(x_R)]$. However, this modification has no effect when optimizing $S$, as the model $p$ is considered fixed.
optimizing $S$ one minimizes the expected (Shannon or differential) entropy of the posterior distribution of the quantities of interest, $x_R$.

Our second example seems to have received little attention in the literature. We define the Bayesian power of inferring $x_R$ based on $x_S$ as a special case of the expected posterior utility by putting $A = A_R$ and $U(x_R, a) = 1(x_R = a)$, the indicator utility function. The term “power” is here appropriate since in this case the expected posterior utility gives the probability that the posterior guess for $x_R$ is correct, thus resembling the frequentist notion of statistical power. The motivation for this concept is obvious. Namely, like its cousin, the expected information gain, Bayesian power can be used in the design of experiments. Further, when the experiments are fixed, Bayesian power quantifies the prior risk of ending up with incorrect posterior conclusions. Here it is worth noting that “simple” models are more powerful than “complex” models—actually this is a way to define model complexity.

An important instance of Bayesian decision making is often considered under the names of model comparison and model selection (or model choice). We consider a class $\mathcal{M}$ of hypotheses, such that each hypothesis $M \in \mathcal{M}$ specifies a probability model $p(x_S \mid M)$ on the observables, $x_S$. We allow the number and semantics of the latent quantities depend on the hypothesis, and therefore we do not explicitly consider the modeling of the latent quantities here. Many common settings in statistical modeling have a structure of this type. For example, the hypothesis $M$ may specify the number of components in a mixture model, the lag in a time series model, the subset of covariates (predictors) in a classification model, or the distributional form of “noise” in a regression model. Now model comparison is about quantifying the plausibility or utility of different hypotheses, whereas model selection is about choosing the most plausible or useful hypothesis.

Let us consider the ideal case, where the modeler believes that one of the hypotheses in $\mathcal{M}$ is true. Bernardo and Smith [BS94] call this the $\mathcal{M}$-closed case; we discuss the opposite, $\mathcal{M}$-open case, in Section 2.3. In the $\mathcal{M}$-closed case, there exists a prior on $\mathcal{M}$, and hence, a joint model $p(x_S, M) = p(M) p(x_S \mid M)$. Thus, the unknown true hypothesis, $M$, acts as a random quantity, and can be treated like any other quantity included in the model. Consequently, for any preferences associated to the comparison and selection task, the necessary and sufficient quantification of the beliefs is given by the posterior distribution $p(M \mid x_S)$. In general, the answers regarding model comparison and selection, of course, depend on the associated utilities. For model comparison the posterior as such might be a sufficient answer, corresponding to the logarithmic utility function.
2.3 Practical versus ideal Bayesian modeling

For model selection a hypothesis that maximizes the posterior might be an appropriate answer, corresponding to the indicator utility function.

2.3 Practical versus ideal Bayesian modeling

Unfortunately, the ideal Bayesian paradigm, reviewed in the previous section, may be difficult to follow in practice. There are three main reasons for this. First, building a probability model that is an honest description of the decision-maker’s personal beliefs may be a very tedious task, if not impossible. We refer to this difficulty as the elicitation problem. Second, computations required to, e.g., handle posterior distributions or, ultimately, to find an optimal action, may be intractable due to lack of analytic solutions. We refer to this difficulty as the resource problem. The third challenge is in scientific reporting and argumentation based on subjective modeling. We refer to this difficulty as the consensus problem. In this section we consider these important problems of practical Bayesian modeling in more detail.

Many Bayesian statisticians seem to agree with Box [Box76] in the frequently quoted viewpoint that

“all models are wrong, but some are useful”

(e.g., [BS94, p. 238], [SBCvdL02, O’H03]). How that assertion should be interpreted is, however, somewhat unclear. Under the subjective Bayesian paradigm a model $p$ is “true” or “correct” if it “truly” expresses a subject’s personal degree of belief concerning all possible statements about the world. This criterion of truth does not make any connection between the subjective beliefs and the external “objective world”. Accordingly, different models can be correct, but only one per subject. Thus, a model can be wrong only in the presence of uncareful elicitation. A less radical definition would be that $p$ is “true” or “correct” if all “objectively true statements” about the world have positive probability under $p$. Accordingly, models in practice tend to be wrong, not only due to the elicitation difficulty, but also due to the fundamental difficulty (or even, impossibility) of constructing meaningful statements about the world (including the truth). Even if the modeler could elicitate her correct model, simplifications to the model may be required due to computational complexities. This is another source of wrong models. However, we include also these cases in the elicitation problem and not in the resource problem. Recall that the resource problem refers to challenges arising when one has to approximate computations under a fixed target model, which may be the correct model, or a simplification or proxy of it.
It is a fundamental problem of Bayesian modeling that one has to deal with approximations to the correct model. Let us pursue this challenge further. It is convenient to assume that one can handle models that are “conditionally correct”. That is, given that certain modeling assumptions—which we call a hypothesis and denote by \( M \)—are true, the distribution \( p(x \mid M) \) is available. Here \( p \) refers to the correct subjective probability model, and \( M \) is treated as a variable that takes values from a set \( \mathcal{M} \). It is also convenient to denote the correct hypothesis by \( M^* \) so that \( p(x \mid M^*) \) denotes the correct model. Thus, we have assumed that \( M^* \notin \mathcal{M} \), and that \( p(M^*) = 1 \) implying \( \sum_{M \in \mathcal{M}} p(M) = 0 \). In words, one is supposed to use probability models that are conditional on (certainly) false hypotheses. Bernardo and Smith [BS94] call this the \( \mathcal{M} \)-open case. The question is now: can the Bayesian methods be used in such an uncomfortable situation and still provide reasonable quantifications of uncertainty? One popular approach is to complete the set \( \{p(x|M) : M \in \mathcal{M}\} \) to a joint model over \( x \) and \( M \) by introducing a technical probability distribution on the set of available hypotheses. Then the completed model can be formally used as if it was the correct model. This is perhaps the most common way to handle the issue [BS94, p. 384]. However, this violates the Bayesian paradigm, since the probabilities no more represent the modeler’s beliefs. Consequently, maximization of expected utility will not yield optimal decisions (with respect to modeler’s beliefs).

To overcome this problem, Valpola [Val00, p. 16] informally suggests that the statement “the hypothesis \( M \) is true”, denoted briefly by \( M \), should be replaced by “the hypothesis \( M \) is the best for modeling \( x \)”. It seems that this reinterpretation fails to solve the problem. To see this, denote the latter version briefly by \( \tilde{M} \). We notice that a proper prior distribution \( p(\tilde{M}) \) can be assigned. But we also notice that, in general, the likelihood \( p(x \mid \tilde{M}) \) does not coincide with the available one, \( p(x \mid M) \). In fact, the elicitation of \( p(x \mid \tilde{M}) \) may be difficult.

There are also other attempts to handle the problem of wrong models. One approach is to select a single best model \( p(x \mid M) \) based on some appropriate criterion. In the prediction context, for example, a popular technique is the cross-validation estimation of expected utility [BS94, Ch. 6], [VI.03]. However, that method can be justified only by asymptotical properties under certain assumptions of the generating process. Furthermore, when \( \mathcal{M} \) already contains a single element \( M \), there is, of course, nothing to select. O’Hagan [O’H03] proposes model criticism where the idea is to evaluate the adequacy of a model as a representation of the data, without reference to explicit alternative models. Various techniques can be used to check the ad-
2.3 Practical versus ideal Bayesian modeling

equacy, including global predictive diagnostic techniques, and local conflict
diagnostics (see [O'H03]). The main problem in these approaches is that
they cannot adhere to the Bayesian paradigm. Rather, they are exploratory
tools.

Does there exist any satisfactory solution to the fundamental problem?
It seems that the problem of wrong models might be fruitfully approached
from the perspective of approximation, as we now briefly sketch. The idea
is that if the modeler employs a model that she believes to be close to
her personal model, then no important distortion will occur in quantifying
uncertainties. An outline of a more formal treatment could be as follows.
First, represent the uncertainty about some appropriately defined distance
of the collection \( \{ p(x \mid M) : M \in \mathcal{M} \} \) and the unavailable correct model
\( p(x \mid M^*) \). Let this distance be random, since the subject may not know (or
cannot compute) the value due to the intractability of the correct model.
Then, combine the elements in \( \{ p(x \mid M) : M \in \mathcal{M} \} \) into a model \( p(x \mid \mathcal{M}) \),
e.g., by the usual averaging over a technical prior. Continue by showing that
the obtained proxy distribution \( p(x \mid \mathcal{M}) \) is close to the correct unavailable
distribution \( p(x \mid M^*) \). Finally, conclude that the approximation quality is
effectively preserved under probability calculus and when maximizing the
utility function. It seems that no formal results have been formulated and
proved on this basis. Yet, according to intuition this type of approximation
viewpoint might justify the use of wrong models, provided that they are
believed (or known) to be close to the correct model in some appropriate
sense. Pursuing this issue further is beyond the scope of this work; this
type of analysis might be worth developing further.

To consider the resource problem we assume for simplicity that the tar-
get model is the correct model. A representative example of the resource
problem is the computation of the posterior mean of a variable of interest.
It is often the case that there is no efficient exact way to compute the mean.
Consequently, one has to resort to deterministic or randomized approxima-
tion methods (if not willing to simplify the original model). A reasonable
assumption is that the target model can be efficiently evaluated at some,
relatively small, number of points, but that it cannot be handled efficiently
as a whole. The major problem in the conventional approximation meth-
ods, e.g., the Monte Carlo methods, is the way they measure the quality
of the approximation. The guaranteed absolute or relative errors may be
sufficient for a decision-maker only if the errors are negligible. More impor-
tantly, randomized approximation algorithms are guaranteed to work only
with some confidence, which is entirely a frequentist concept. This fact is
pointed out by O'Hagan [O'H87] in the context of Monte Carlo methods for
Bayesian inference. We discuss this issue more in Chapter 4.

The consensus problem concerns the nature of objectivity in scientific inquiry. There have been serious attempts to define "objective" Bayesian methods. The motivation is clear. As Bernardo [Ber03] states it,

"...in many important situations, including scientific reporting and public decision making, the results must exclusively depend on documented data which might be subject to independent scrutiny."

In these attempts the likelihood model is taken as given. The goal is to define the prior in an objective way, free of all subjective aspects. Perhaps the most advanced "objective" procedure so far is called reference analysis [Ber79b, BB92]. Many other famous approaches, such as maximum entropy and (univariate) Jeffreys's rule, are found as special cases of the general reference prior. In reference analysis a reference prior is defined based on information theoretical considerations. Among a set of candidate priors, the reference prior maximizes the expected amount of information that is obtained from observations under the assumed likelihood model. In that sense the reference prior is the most noninformative prior. The set of candidates can be the set of all probability distributions or it can be restricted by some "generally accepted" assumptions. A more precise definition can be found, e.g., in [Ber03].

An objection to the reference method lies in the fact that the reference prior, and hence also the posterior, does not enjoy the subjective interpretation. Consequently, the reference posterior cannot directly support the decision making. So, the role of reference methods is indirect. Reference analysis gives answers to a what-if question, namely "what could be said about the quantity of interest given the data, if one's prior knowledge were dominated by the data?" [Ber98]. The reference method may be criticized also because it explicitly separates the prior and likelihood parts of the model and assumes that the likelihood model is generally accepted as the model of collective beliefs. What if there is a wide class of possible likelihood models, should not one complete the reference prior by a reference likelihood into a joint reference model? Although logically justified, it seems that the results would then not be useful anymore. Unfortunately, this issue is not discussed in [BS94, Ber98, Ber03].

Based on the concept of Kolmogorov complexity, stronger versions of objective Bayesian-like approaches have been proposed. Roughly, the Kolmogorov complexity of a string $x$ is the length of the shortest computer program that outputs $x$; a more precise definition is given in terms of universal Turing machines and asymptotic function classes (see [LV97]). Since
any sequence of finite objects, observable and latent quantities, can be represented as a string \( x \), the probability of \( x \) can be assigned via its Kolmogorov complexity. This way one gets a model \( p(x) \), where \( p \) is called a universal distribution. The universal model assigns the probability distribution according to “objective regularities” or “absolute information content” on \( x \), not based on the modeler’s subjective beliefs. Unfortunately, this seeming objectivity hides some crucial subjective elements. Namely, the Kolmogorov complexity measure becomes well defined only after fixing a (universal) Turing machine. While for infinite strings the choice of the Turing machine is not important, for finite strings the Kolmogorov complexities crucially depend on the machine. The unavoidable conclusion is that the approach, known as the ideal minimum description length induction [VL00], provides no added value in statistics—fixing a Turing machine corresponds to a subjective choice of \( p \), and thus to a subjective Bayesian model. This view seems to be in sharp contrast to those presented in, e.g., [VL00, GTV01].

A minor issue is that the Kolmogorov complexity (in any form) is a non-computable function. Also motivated by the idea of data compression, Rissanen [Ris76, Ris87, Ris96] develops computable universal models that are applicable also in practically interesting finite cases. The most recent formulations, however, follow the principle of worst-case relative utility, and should not be considered as Bayesian-like methods.

If objective Bayesian methods do not exist, then how can Bayesian data analysis contribute to scientific inquiry and decision making? To answer this, Bernardo and Smith [BS94, p. 102] separate two different contexts: public reporting and cohesive-small-group decision making. When it comes to public scientific reporting, the Bayesian paradigm does not force one to report results obtained under a single subjective model. Instead, it is recommended to report a rich range of the possible belief mappings induced by a data set, a practice sometimes called sensitivity analysis. In a group decision making situation individual contemplation is not sufficient when there is a need for group belief and decision. How to combine individual probabilities and utilities depends on the “rules of the game”: whether the aim is at information sharing, democracy, negotiation, or competition. Once the rules and conventions are fixed, the group decision will emerge in the game where each group member tries to maximize her personal utility under her personal belief model. Note, however, that the Bayesian paradigm does not tell which rules or commitments lead to the most rational way to act as a group. We may conclude by stating that consensus is not the primary goal of the Bayesian paradigm, though consensus may be reached, up to some degree, among people who share similar preferences, initial beliefs,
and experience.

2.4 Discussion

The Bayesian paradigm provides an elegant approach to pursue rational decisions under uncertainty. As its principal ingredient the approach requires the decision-maker to elicitate her subjective probability model, possibly for a large number of variables. Once a model is built, various questions of inference and prediction (induction) can be answered from the unifying, decision theoretic perspective.

Due to practical constraints, however, one often has to resort to simplified models and approximative computation, regarding the elicitation and resource problems, respectively. If one adheres to the Bayesian paradigm, then both types of relaxations would require an assignment of two-level probabilities. That is, one has to consider a subjective probability distribution on functions of subjective probabilities. Unfortunately, this kind of orthodox analysis seems to be too complicated to carry out in practice. Integrals required for Bayesian reasoning, for example, are usually handled by frequentist Monte Carlo procedures. The Bayesian alternatives [O’H91, O’H92, RG04], though more efficient in a certain sense, tend to be computationally demanding.

From a purely practical point of view, the “semi-Bayesian” methods, involving slightly wrong models and frequentist Monte Carlo, can be justified. In a typical case, a rigorous Bayesian analysis would probably give essentially the same results. That said, while certain consistency issues of practical Bayesianism are fundamental and theoretical, their importance to applications is somewhat questionable.

A largely open question seems to be, how useful the Bayesian paradigm is for the analysis of computational problems in general. In particular, the Bayesian approach might give fruitful insight into problems that are usually solved under an essentially frequentist framework by “stochastic” or “randomized” algorithms (see, e.g., [MR95]).
Chapter 3

Sum-products and the elimination algorithm

In this chapter we study the problem of computing a marginal of a function that decomposes into a product of lower dimensional functions. We specify this sum-product problem formally in Section 3.1. In Section 3.2, we review a well-known, general algorithmic solution, the variable elimination algorithm. Using the results on fast matrix multiplication, we show in Section 3.3 that the conventional variable elimination algorithm is not optimal in general. An important special class of sum-product problems is formed by transformation problems introduced in Section 3.4. Some special transforms and related algorithms, including the Yates algorithm, are derived. We also show how the Möbius transform on a subset lattice can be mapped to a sum-product form. In Section 3.5, we emphasize the importance of finding appropriate sum-product representations and discuss the limitations of the variable elimination algorithm. Finally, Section 3.6 provides some remarks and open problems.

This chapter offers a generic, algorithmic perspective on certain issues studied elsewhere in this work. The study of computing marginals is, in general, motivated by the Bayesian paradigm reviewed in Chapter 2, and especially by the genetic application problem described in Chapter 5. Results on the Yates algorithm and Möbius transform, given in this chapter, are applied in Chapter 6 for the computation of recurrence risks. Chapter 4 considers a related problem of marginalization over infinite domains.

3.1 The problem of marginalizing a product

We start by considering an example of a sum-product problem.
Example 3.1 Consider a 4-dimensional marginal of an 8-dimensional function:

\[ g(x_1, x_2, x_3, x_4) = \sum_{x_5=1}^{m} \sum_{x_6=1}^{m} \sum_{x_7=1}^{m} \sum_{x_8=1}^{m} f(x). \]

Here \( x = (x_1, \ldots, x_8) \). Suppose further that \( f \) decomposes into a product of four “local” functions:

\[ f(x) = a(x_1, x_5, x_8) b(x_2, x_5, x_6) c(x_3, x_6, x_7) d(x_4, x_7, x_8). \]

How fast can the marginal function \( g \) be computed given \( a, b, c, \) and \( d \)? Clearly, the straightforward method requires \( O(m^8) \) arithmetic operations. However, by exploiting the assumed problem structure, it is possible to derive an algorithm that uses only \( O(m^4) \), or even \( O(m^3) \), operations; such algorithms will be given in examples 3.16 and 3.23.

In order to cover a wider class of computational problems similar to that of Example 3.1, we use a more abstract formulation. Following Aji and McEliece [AM00] we rely on certain general properties of addition and multiplication. The appropriate framework is the commutative semiring.

Definition 3.2 A commutative semiring is a set \( \mathcal{R} \), together with two binary operations called “+” and “\( \cdot \)”, which satisfy the following axioms:

S1. The operation “+” is associative and commutative, and there is an additive identity element called “0” such that \( r + 0 = r \) for all \( r \in \mathcal{R} \).

S2. The operation “\( \cdot \)” is associative and commutative, and there is a multiplicative identity element called “1” such that \( r \cdot 1 = r \) for all \( r \in \mathcal{R} \).

S3. The distributive law holds, i.e., for all \( a, b, c, \in \mathcal{R} \),

\[ (a \cdot b) + (a \cdot c) = a \cdot (b + c). \]

The semiring is denoted as a triple \( (\mathcal{R}, +, \cdot) \).
Table 3.1: Some Commutative Semirings

<table>
<thead>
<tr>
<th>( \mathcal{R} )</th>
<th>‘(+, 0)’</th>
<th>‘(, 1)’</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-(\infty), (\infty))</td>
<td>(+, 0)</td>
<td>(, 1)</td>
<td>Also a ring</td>
</tr>
<tr>
<td>(0, (\infty])</td>
<td>(min, (\infty))</td>
<td>(, 1)</td>
<td></td>
</tr>
<tr>
<td>[0, (\infty))</td>
<td>(max, 0)</td>
<td>(, 1)</td>
<td></td>
</tr>
<tr>
<td>(-(\infty), (\infty))</td>
<td>(min, (\infty))</td>
<td>(+, 0)</td>
<td></td>
</tr>
<tr>
<td>[-(\infty), (\infty))</td>
<td>(max, (-\infty))</td>
<td>(+, 0)</td>
<td></td>
</tr>
<tr>
<td>[-(\infty), (\infty])</td>
<td>(max, (-\infty))</td>
<td>(min, (\infty))</td>
<td></td>
</tr>
<tr>
<td>{0, 1}</td>
<td>(+, 0)</td>
<td>(, 1)</td>
<td>Modulo 2; also a ring</td>
</tr>
<tr>
<td>{0, 1}</td>
<td>(OR, 0)</td>
<td>(AND, 1)</td>
<td>Boolean</td>
</tr>
<tr>
<td>(\mathcal{P}(S))</td>
<td>((\cup), (\emptyset))</td>
<td>((\cap), (S))</td>
<td>(S) a finite set</td>
</tr>
<tr>
<td>(\Lambda)</td>
<td>((\vee), 0)</td>
<td>((\land), 1)</td>
<td>(\Lambda) a distributive lattice</td>
</tr>
<tr>
<td>(\Lambda)</td>
<td>((\land), 1)</td>
<td>((\lor), 0)</td>
<td>(\Lambda) a distributive lattice</td>
</tr>
<tr>
<td>(R[x, y, \ldots])</td>
<td>(+, 0)</td>
<td>(, 1)</td>
<td>(R) a commutative ring; also a ring</td>
</tr>
</tbody>
</table>

Perhaps interesting, but not included in [AM00], is the maximin-semiring (line 6 in Table 3.1); one can easily verify the distributive law, for all \(a, b, c\),

\[
\max\{\min\{a, b\}, \min\{a, c\}\} = \min\{a, \max\{b, c\}\}.
\]

A more important fact, yet not included in Table 3.1, is that the semiring properties hold for a set of mappings to a semiring when the addition and multiplication of two mappings are defined in a point-wise manner; additionally it is required that the set contains the 0- and 1-identity maps and that it is closed under addition and multiplication. For instance, the set of polynomial maps, where the coefficients and the variables take values from a commutative semiring, forms a commutative semiring. Note also that the polynomials in one or more indeterminates over any commutative semiring, as a formal system of coefficients, also forms a commutative semiring (the last line in Table 3.1).

Sometimes stronger properties hold for the algebraic structure in question. An important restriction of the commutative semiring is the commutative ring.

**Definition 3.3** A commutative semiring \((\mathcal{R}, +, \cdot)\) is a **commutative ring** if every element \(a\) in \(\mathcal{R}\) has an additive inverse in \(\mathcal{R}\), i.e., an element \(b\) such that \(a + b = 0\).

An example of commutative ring is the set of integers together with ordinary addition and multiplication. If we drop the requirement of commuta-
tive multiplication, we get a semiring or a ring. For example, the set of square matrices over a (semi)ring, with ordinary matrix addition and multiplication, forms a (semi)ring, but not a commutative (semi)ring. Also, if for every nonzero element in a ring there is a multiplicative inverse, then the ring is called a field.

The existence of additive inverses in commutative ring has mostly been neglected in related studies of sum-product algorithms [SH96, LJ97, BMR97, AM00, KFL01]. We show in Section 3.3 that in some cases this extra property can be used to devise (asymptotically) faster algorithms. This is interesting, not only because one often operates with commutative rings, but also because in some cases a commutative semiring that is not a commutative ring as such, can be embedded into a commutative ring with little computational overhead. Until Section 3.3 we, however, work only on the general case of commutative semirings.

We are now ready to describe the generalized sum-product problem. First, let us introduce some notation. Throughout this chapter, we let $x_1, \ldots, x_n$ be variables taking values in the finite sets $A_1, \ldots, A_n$. We use $N$ as a shorthand for the set $\{1, \ldots, n\}$. If $S = \{i_1, \ldots, i_r\}$ where $i_1 < \ldots < i_r$, is a subset of $N$, we denote the product $A_{i_1} \times \cdots \times A_{i_r}$ by $A_S$, and the variable list $(x_{i_1}, \ldots, x_{i_r})$ similarly by $x_S$. We call $S$ the scope of $x_S$ and $A_S$ the domain of $x_S$. We may drop the subscript when it refers to the whole universe $N$. The complement of $S$ (relative to the universe $N$) is denoted by $S^c$. We need also the following definition.

**Definition 3.4** Let $T$ be a collection of subsets of a set $V$. We say that $T$ is a cover of $V$ if the union of the elements in $T$ equals $V$, i.e., $\bigcup T = \bigcup_{T \in T} T = V$. □

The generalized sum-product problem is now defined as follows.

**Definition 3.5** Let $A_1, \ldots, A_n$ be finite sets. Let $(\mathcal{R}, +, \cdot)$ be a semiring. Let $T$ be a cover on $N$ and let $S$ be a subset of $N$. The tuple $(A, \mathcal{R}, T, S)$ defines an evaluation problem: Given functions $f_T : A_T \rightarrow \mathcal{R}$ for $T \in T$, evaluate for all $x_S \in A_S$ the value $g_S(x_S)$ of the marginal function $g_S : A_S \rightarrow \mathcal{R}$ defined by

$$g_S(x_S) = \sum_{x_S \in A_S} \prod_{T \in T} f_T(x_T).$$

The problem is called the single marginalization of a product function problem, and is denoted by SMPF$(A, \mathcal{R}, T, S)$. A problem instance is a set $\mathcal{F} = \{f_T : T \in T\}$ of fully specified functions. □

\[^1\]In the literature a ring that contains a multiplicative identity element is sometimes called a unit ring. Here we do not make this distinction.
3.1 The problem of marginalizing a product

We note that an SMPF problem essentially specifies an operator (or functional), i.e., a mapping from the function space specified by $\mathcal{A}$, $\mathcal{R}$, and $\mathcal{T}$ onto the function space specified by $\mathcal{A}$, $\mathcal{R}$, and $S$. This view suggests and supports a more compact notation for the expressions. For instance, we may write the marginal function simply as

$$g_S = \sum_{S^c} \prod_{T \in \mathcal{T}} f_T.$$

Here the addition and multiplication of functions are defined in a point-wise manner. This requires that we understand a function $f_T : \mathcal{A}_T \rightarrow \mathcal{R}$ as a mapping from $\mathcal{A}$ onto $\mathcal{R}$ for which only the arguments in $\mathcal{A}_T$ are relevant. More precisely, for any $x, x' \in \mathcal{A}$ we have $f_T(x) = f_T(x')$ whenever $x_T = x'_T$. In this case we say that $f_T$ is a function over $T$. This way the sum $f_S + f_T$ and the product $f_S f_T$ become well defined, the result being a function over $S \cup T$.

The following example is to clarify the SMPF problem representation and notation.

**Example 3.6** We continue Example 3.1. We represent the sum-product problem described in Example 3.1 in the form of Definition 3.5. First, we have $n = 8$ variables with domains $\mathcal{A}_i = \{1, \ldots, m\}$ for $i = 1, \ldots, 8$. Second, we operate in the commutative semiring $\mathcal{R}$ of real numbers equipped with ordinary addition and multiplication. We notice that this semiring is also a commutative ring. Third, the set of local scopes is $\mathcal{T} = \{T_1, T_2, T_3, T_4\}$, where $T_1 = \{1, 5, 8\}$, $T_2 = \{2, 5, 6\}$, $T_3 = \{3, 6, 7\}$, and $T_4 = \{4, 7, 8\}$. The correspondence is obvious: $a = f_{T_1}$, $b = f_{T_2}$, $c = f_{T_3}$, and $d = f_{T_4}$. Clearly, $\mathcal{T}$ is a cover on $N = \{1, \ldots, n\} = \{1, \ldots, 8\}$. Fourth, the marginal over $S = \{1, 2, 3, 4\}$ is to be computed, implying $S^c = \{5, 6, 7, 8\}$. With this specification the problem of Example 3.1 is an instance $\mathcal{F} = \{f_T : T \in \mathcal{T}\}$ of SMPF($\mathcal{A}, \mathcal{R}, \mathcal{T}, S$). A notationally convenient way to write the marginal function is

$$g_{1234} = \sum_{5678} f_{158} f_{256} f_{367} f_{478}.$$

Here 158, for example, is a shorthand for the set $T_1 = \{1, 5, 8\}$. □

To describe SMPF as an algorithmic problem we need to fix the input and output representations. Since a problem instantiation is specified by a set of functions, it is natural to let the input be a list of function representations. Similarly the output is a single function representation. As the representation we assume a basic, search tree data structure. More
precisely, for a scope $T = \{i_1, \ldots, i_r\}$ where $i_1 < \ldots < i_r$, we represent a function $f_T : \mathcal{A}_T \to \mathcal{R}$ by a tree of height $r$. At level $(k-1)$ of the tree every node has $|\mathcal{A}_{i_k}|$ child nodes, one per each possible value for variable $x_{i_k}$. Consequently, the tree has $|\mathcal{A}_T|$ leafs, one per each vector $x_T \in \mathcal{A}_T$. Note that we treat the problem structure $(\mathcal{A}, \mathcal{R}, \mathcal{T}, S)$ as a part of the problem specification which is the same for all instances of the SMPF problem.

This brings us to the next issue. How do we measure the computational complexity of SMPF problems? For a single SMPF problem we do this mainly in terms of arithmetic complexity. By arithmetic complexity we mean here the total number of arithmetic operations performed by an algorithm. However, when the nature of the operations so requires, or when table-lookups or similar operations play an important role in the algorithm, or when space complexity is of interest, then these features are incorporated into complexity considerations. For instance, in general we cannot assume that the evaluation of a function $f_T$ at a fixed $x_T$ takes a constant time. Instead, the time is proportional to $|T|$ when using the tree representation. For an algorithm for an SMPF problem we get in this way the arithmetic or time complexity.

To consider asymptotic complexities meaningfully, we need to consider infinite problem classes. In general, any set of SMPF instances forms a problem class. However, we consider mainly classes that are unions of SMPF problems. For example, by letting $m = 1, 2, 3, \ldots$ for the problem described in Example 3.6, we get an infinite class of SMPF problems. Similarly we may consider problems on different structures (covers) and different number of variables.

When considering problem classes we face an extended algorithmic problem. The input is not just a set of functions on a fixed structure, but also the structure itself is taken as a part of the input. By structure we mean here the $\mathcal{A}$, $\mathcal{T}$, $S$ (using the notation of Definition 3.5), whereas the commutative semiring is assumed to be fixed. We will see in the next section that the problem is naturally solved in two phases. In the first phase the structure of a given problem is analyzed, with no reference to the actual instance of the problem (i.e., the local functions). That is, one operates with the graphical structure determined by $\mathcal{T}$ and $S$:

**Definition 3.7** The graph structure\(^2\) of a problem SMPF$(\mathcal{A}, \mathcal{R}, \mathcal{T}, S)$ is a pair $(G_T, S)$, where $G_T = (N, R)$ is an undirected graph with $(i, j) \in R$ if and only if $i \neq j$ and $\{i, j\} \subseteq T$ for some $T \in \mathcal{T}$.

\(^2\)We note that the graph structure does not preserve all information present in the cover. We also note that a cover is a hypergraph, where each member in the cover corresponds to a hyperedge.
Figure 3.1: Two graphical visualizations of the structure of the problem described in Example 3.6. (a) A graph where a node corresponds to a variable or a local function (boxes, not labeled). There is an edge between a variable node and a function node if the variable is an argument of the function. (b) A graph where a node corresponds to a variable. There is an edge between two nodes if some local function has the corresponding variables as arguments. In both visualizations the marginal variables are indicated by circles.

For an illustration see Figure 3.1. In the second phase the required marginal is evaluated utilizing the results of the structure analysis made in the first phase.

Many interesting and important computational problems can be interpreted as SMPF problems (or problem classes) in a natural way. For example, in signal processing, coding theory, and belief networks, see [AM00, KFL01]. Classical “combinatorial problems”, including SAT and MAXSAT, are included in [SH96]. According to Stearns and Hunt [SH96], the representations for these problems are “direct and structure-preserving”.

Stearns [Ste03] suggests analyzing sum-product problems in order to establish lower bounds for the problem complexity. For the discrimination of hard and easy problems, he suggests focusing on cases where the primitive variable domains are small, say $|A_i| = 2$, and where the local scopes are small, say $|T| \leq 3$ for all $T \in T$. This proposal is understandable when one is interested in analyzing how hard problems may arise from a seemingly simple system, or which constraints are needed to ensure that there exists an efficient (polynomial) algorithm for the problem. In this work we do not follow this idea, since we are also interested in cases where the input size may be very large, even exponential in $n$. For such problems we seek fast polynomial algorithms with respect to the input size, though often exponential in $n$. 
Even more general sum-product problem formulations than that of SMPF have been presented. The problem of computing multiple marginals, called “marginalize-a-product-function” (MPF) is introduced in [AM00]. That problem has important applications and can be solved, using so called “junction trees”, more efficiently than by simply solving multiple SMPF problems. In different guises this problem and method appear also in [SH96, KFL01]. From the perspective of this chapter, however, the difference to the SMPF problem defined above is not important. Yet, we reserve the abbreviation MPF for that problem, and consequently we use the modifier “single” for our restricted problem. Still slightly more general than MPF are the formulations in [Dec96, LJ97]. A different, interesting extension is that of Stearns and Hunt [SH02] where more than two operations, e.g., ordinary “max”, “+”, and “·” on $\mathbb{R}$, are considered simultaneously.

### 3.2 The variable elimination algorithm

Since important special cases of the SMPF problem have appeared for a long time in different contexts, it is not a surprise that the same solution has been discovered many times independently. Various versions of essentially the same technique have appeared under many names: Pearl’s algorithm [Pea88], bucket elimination [Dec96], generalized distributive law [AM00]. The same method appears in different guises also in [SH96, LJ97, KFL01]. Here we call the method the **variable elimination algorithm**.

As hinted by its name, the method proceeds iteratively marginalizing the variables one by one. Accordingly, the method follows some order of the variables.

**Definition 3.8** An *elimination order* on a set $U$ is a linear order on $U$. $\square$

A marginalization eliminates a variable. What is left after eliminating a variable is again an instance of an SMPF problem, but over a smaller number of variables.

A formal description of the variable elimination algorithm is given below (Algorithm 3.9). This description of the variable elimination algorithm borrows much from that of Dechter [Dec96]. The main difference is that here we use a rigorous set representation whereas Dechter somewhat loosely mixes (ordered) lists and (unordered) sets. A consequence of this choice is that duplicates of local scopes are forbidden. That is, for each local scope $T$ we can have exactly one local function $f_T$ in the collection $\mathcal{F}$. To satisfy this, at each elimination step of the algorithm, a special routine is performed (line 4 of Algorithm 3.9). Note that the intermediate functions
3.2 The variable elimination algorithm

computed by the algorithm are denoted by $f_{S'}$, not, e.g., by $g_{S'}$. With this notational choice we emphasize the iterative nature of the algorithm, that is, the fact that these intermediate functions will serve as local functions in the subsequent iterations. Moreover, we reserve the letter $g$ for the ultimate target marginal. This distinction is convenient when we proving the correctness of the algorithm (Theorem 3.11).

**Algorithm 3.9** (the variable elimination algorithm)

**Input** an instance $\mathcal{F} = \{f_T : T \in \mathcal{T}\}$ of SMPF($A, R, \mathcal{T}, S$) and an elimination order $\prec$ on $S^c$

**Output** the marginal function $g$ as described in Definition 3.5

**Method**

1. for each $i \in S^c$ in the order $\prec$ do
2. $\mathcal{T}' \leftarrow \{T \in \mathcal{T} : i \in T\}$ % local scopes that cover $i$
3. $S' \leftarrow \bigcup \mathcal{T}' - \{i\}$ % covered variables not to be eliminated
4. $\mathcal{T}' \leftarrow \mathcal{T}' \cup \{T \in \mathcal{T} : T \subseteq S'\}$ % to avoid duplicates
5. $f_{S'} \leftarrow \sum_i \prod_{T \in \mathcal{T}}, f_T$ % eliminate $i$th variable
6. $\mathcal{T} \leftarrow \mathcal{T} \cup \{S'\} - \mathcal{T}'$ % update local scopes
7. $\mathcal{F} \leftarrow \mathcal{F} \cup \{f_{S'}\} - \{f_T : T \in \mathcal{T}'\}$ % update functions, used implicitly
8. return $\prod_{T \in \mathcal{T}} f_T$

The next example shows a run of the variable elimination algorithm.

**Example 3.10** We continue examples 3.1 and 3.6. Recall that the problem is to compute the marginal function

$$g_{1234} = \sum_{5678} f_{158} f_{256} f_{367} f_{478}. $$

We list the elimination steps of the variable elimination algorithm when run on the elimination order $8 \prec 7 \prec 6 \prec 5$. At the first iteration we have $S' = 1457$ and the elimination step takes the form

$$f_{1457} \leftarrow \sum_{8} f_{158} f_{478}. $$

Note that if the problem was slightly different so that there was already a local function $f_{1457}$, then this function would be also included in the summation on the right-hand side. This is to avoid duplicated local scopes and to adhere to the set representation. The collection of local scopes (the cover) and local functions are updated to $\mathcal{T} = \{256, 367, 1457\}$ and $\mathcal{F} = \{f_{256}, f_{367}, f_{1457}\}$, respectively. The subsequent three iterations correspond
to elimination steps

\[ f_{13456} \leftarrow \sum_{7} f_{367} f_{1457}, \]
\[ f_{12345} \leftarrow \sum_{6} f_{256} f_{13456}, \]
\[ f_{1234} \leftarrow \sum_{5} f_{12345}. \]

This \( f_{1234} \) is finally returned by the algorithm. We observe that this \( f_{1234} \) coincides with the required marginal \( g_{1234} \). \qed

**Theorem 3.11** Algorithm 3.9 is correct.

**Proof** Let \( g_S \) be the marginal function to be computed. Let \( S^c = \{i_1, \ldots, i_r\} \) with \( i_1 < i_2 < \cdots < i_r \). We prove by induction on \( k = 1, 2, \ldots, r \) that after line 7 in \( k \)th iteration we have

\[ g_S = \sum_{C_k} \prod_{T \in T} f_T, \tag{3.1} \]

where \( C_k = S^c - \{i_1, \ldots, i_k\} \).

Consider the general case, \( 1 \leq k \leq r \). We suppose that in the beginning of the \( k \)th iteration the program variable \( T \) is such that the induction assumption (3.1) holds for \( k - 1 \). Note that for \( k = 1 \) this holds trivially. Consider the collection \( T \) when the algorithm performs lines 2 to 5 in the \( k \)th iteration. Divide \( T \) into two parts: \( T' \) as specified on lines 2–4 and the remaining part \( T'' = T - T' \). Now we may write

\[ g_S = \sum_{C_{k-1}} \prod_{T \in T} f_T = \sum_{C_k} \left( \prod_{T \in T''} f_T \sum_{i_k} \prod_{T \in T'} f_T \right), \]

by the commutativity of the addition and multiplication, and by the distributive law. We note that \( \sum_{i_k} \prod_{T \in T'} f_T \) is a function over \( \bigcup T' = S' \), where \( S' \) is as defined on line 3. Thus, by the substitutions on lines 5–6 we get that

\[ g_S = \sum_{C_k} \prod_{T \in T''} f_T f_{S'} = \sum_{C_k} \prod_{T \in T} f_T. \]

Recall that \( T \) refers here to the updated set of local scopes, not to the original one. This completes the induction.
3.2 The variable elimination algorithm

Finally, note that after the last, rth iteration we have \( C_r = \emptyset \), and hence 
\[ g_S = \prod_{T \in T} f_T. \] 
This is precisely what the algorithm outputs on line 8, thus completing the proof.

The complexity of the variable elimination algorithm is determined by the size of the local scopes that are generated along the way. This in turn depends on the elimination order that is an input of the algorithm. To give a more precise characterization we need to define some quantities induced by the variable elimination algorithm.

**Definition 3.12** Consider a run of Algorithm 3.9 on a problem \( \text{SMPF}(A, \mathcal{R}, \mathcal{T}, S) \) and an elimination order \( \prec \) on \( S^c \). For each \( i \in S^c \) denote by \( S'_i \) the corresponding set \( S' \) defined on line 3. The *induced maximum width*, *induced maximum size*, and *induced total size* are defined as

\[
w_{\text{max}} = \max_{i \in S^c} |S'_i \cup \{i\}|, \quad s_{\text{max}} = \max_{i \in S^c} |\mathcal{A}_{S'_i \cup \{i\}}|, \quad s_{\text{tot}} = \sum_{i \in S^c} |\mathcal{A}_{S'_i \cup \{i\}}|,
\]

in the respective order.

These induced quantities play a central role in the complexity of the variable elimination algorithm. A minor contribution to the complexity is due to the need of computing a product of the local functions. The following result shows that local functions can be multiplied, added, and marginalized with a relatively low cost, in linear time with respect to the size of the input and output. This observation does not appear in related studies (e.g., [SH96, AM00, KFL01]) where a constant time access to an array representation of the functions is assumed either implicitly or explicitly.

**Proposition 3.13** Suppose that the arithmetic operations on a semiring \( \mathcal{R} \) can be performed in a constant time. Then, given tree representations of two functions \( f_S : \mathcal{A}_S \to \mathcal{R} \) and \( f_T : \mathcal{A}_T \to \mathcal{R} \), the tree representations of the sum \( f_S + f_T \) and the product \( f_S f_T \) can be formed in time \( \mathcal{O}(|\mathcal{A}_{S \cup T}|) \). Similarly, the tree representation of the marginal \( \sum_T f_{S \cup T} \) of a function \( f_{S \cup T} : \mathcal{A}_{S \cup T} \to \mathcal{R} \) can be formed in time \( \mathcal{O}(|\mathcal{A}_{S \cup T}|) \).

**Proof** We first sketch a simple algorithm for the summation problem—the multiplication problem can be handled similarly. First form a tree for a function \( f_{S \cup T} \) over \( S \cup T \). Then traverse the three trees simultaneously in a straightforward depth-first manner so that for every \( x_{S \cup T} \in \mathcal{A}_{S \cup T} \) the corresponding leaves of the three trees can be pointed to in a constant time. While traversing, for each \( x_{S \cup T} \) put \( f_{S \cup T}(x_{S \cup T}) \leftarrow f_S(x_S) + f_T(x_T) \) in a constant time.
Consider then the computation of the marginal. Let $R = S \cup T - T$ so that $R$ and $T$ are disjoint and $f_R = \sum_T f_{R \cup T}$ denotes the marginal function. We sketch an algorithm similar to that above. First form a tree for the function $f_R$ and initialize the values to zero. Then traverse the two trees of $f_R$ and $f_{R \cup T}$ simultaneously depth-first so that for every $x_{R \cup T} \in \mathcal{A}_{R \cup T}$ the corresponding leaves of the two trees can be pointed to in a constant time. While traversing, for each $x_{R \cup T}$ update $f_R(x_R) \leftarrow f_R(x_R) + f_{R \cup T}(x_{R \cup T})$ in a constant time. □

For the complexity of the variable elimination algorithm we obtain an upper bound easily. The following bound involves the number of local functions, that is, the size of the cover.

**Theorem 3.14** The variable elimination algorithm runs in time $O((n + |\mathcal{T}|)s_{\text{max}})$, provided that the arithmetic operations can be performed in a constant time.

**Proof** Consider first the number of additions. The total number of additions performed by the algorithm is clearly $O(s_{\text{tot}})$, and hence $O(ns_{\text{max}})$. Then consider the number of multiplications. Let the order be $i_1 \prec i_2 \prec \cdots \prec i_r$. Let $t_h$ denote the size of $\mathcal{T}'$ at the $h$th iteration. Similarly, let $t_0$ denote the number of local scopes after the $r$ iterations. Since every elimination step combines $t_h$ functions into a single function, we have

$$t_0 + \sum_{h=1}^{r} (t_h - 1) = |\mathcal{T}|.$$

Thus the total number of multiplications is bounded by $|\mathcal{T}|s_{\text{max}}$. The total number of arithmetic operations performed by the algorithm is thus $O((n + |\mathcal{T}|)s_{\text{max}})$. By Proposition 3.13 this bound holds also for the running time, completing the proof. □

An apparent shortcoming of the above analysis is that it is conditional on the elimination order and a single SMPF problem. When considering a class of SMPF problems, a preliminary algorithmic phase for finding a good, if not optimal, elimination order for a given SMPF problem needs to be introduced. Running Algorithm 3.9 is then the second phase. This way, handling SMPF problems can be naturally divided into a preliminary phase that operates only on the problem structure, and a subsequent phase that performs arithmetic computations on a semiring.

An optimal elimination order can be defined with respect to any of the above defined three complexity measures. An optimal elimination order in turn induces the optimal value of the corresponding complexity measure.
3.2 The variable elimination algorithm

Definition 3.15 For an SMPF(\(\mathcal{A}, \mathcal{R}, \mathcal{T}, S\)) define the maximum width, denoted by \(w^*_\text{max}\), as the minimum of the maximum widths induced by different elimination order on \(S^c\). Similarly define maximum size and total size, denoted by \(s^*_\text{max}\) and \(s^*_\text{tot}\), respectively.

Let it be noted that the maximum width appears under many names. For example, Becker and Geiger [BG01] uses term “cliquewidth”. A more commonly used quantity is the “treewidth” which is defined as the maximum width minus 1 [SH96]. Stearns and Hunt [SH96] also use terms “weighted depth” and ‘channelwidth’ to refer similarly defined quantities.

Finding an optimal elimination order is hard. In the class of all SMPF problems, finding an optimal elimination order is NP-complete [ACP87]. Interestingly, however, for a case where the maximum width is \(w^*\) there exits an \(O(n^{w^*+1})\) algorithm that finds an optimal order with respect to the maximum width [ACP87]. This bound is polynomial in \(n\) if the maximum width is bounded by a constant. But this is, of course, not always the case. Nevertheless, Shoikhet and Geiger [SG97] demonstrate that a related triangulation problem on typical (random) graphs can be solved optimally up to 100 nodes (variables) in a reasonable time frame. On the other hand, several approximation algorithms have also been designed to find close to optimal elimination order (see, e.g., [BG01], and references therein). We do not consider these methods here.

An interesting question is posed by Becker and Geiger [BG01]: does there exist an algorithm that finds the optimal elimination order (say, with respect to the maximum width) as fast as the variable elimination algorithm runs on the optimal elimination order? The requirement becomes \(O(n2^{w^*})\), since we may assume binary primitive domains and simple semiring operations. This question can be regarded as a highly relevant open problem. Some hope for a positive answer is provided in [Bod96] where an \(O(nf(w^*))\) algorithm is given, where \(f\), however, is a superexponential function of \(w^*\).

We conclude this section by analyzing the complexity of the variable elimination algorithm for the problem considered in the earlier examples.

Example 3.16 We continue Example 3.10. Recall that the problem is to compute the marginal function

\[ g_{1234} = \sum_{5678} f_{158} f_{256} f_{367} f_{478}. \]

Consider the elimination algorithm on different elimination orders. We note that there are \(4! = 24\) different orders. However, we can easily find an optimal elimination order, since there is plenty of symmetry in the structure. By the symmetry of the variable labels, we may assume that the first
element in the order is 8. Now in Algorithm 3.9 we have $S' = 1457$ and the elimination step takes the form

$$f_{1457} \leftarrow \sum_{8}^{158} f_{478}.$$ 

We see that the computation of $f_{1457}$ requires roughly $m^5$ arithmetic operations. The remaining problem may be written as

$$g_{1234} = \sum_{567}^{123456} f_{256} f_{367} f_{1457}.$$ 

For the second step we have three variables where to choose from. However, by the symmetry of 5 and 7 it is sufficient to consider two cases: $x_7$ next or $x_6$ next. Consider first continuing by elimination of $x_7$. Now $S' = 13456$ and the elimination step results in a function $f_{13456}$. The remaining problem is written as

$$g_{1234} = \sum_{56}^{123456} f_{256} f_{13456}.$$ 

Now 5 and 6 are symmetric in the sense that their elimination order does not affect the required number of operations. Eliminating $x_6$ first results in a function $f_{12345}$. Finally, eliminating $x_5$ result in the function $f_{1234} = g_{1234}$. We conclude that the elimination order $8 \prec 7 \prec 6 \prec 5$ yields the induced total size $m^5 + m^6 + m^6 + m^5$.

On the other hand, eliminating $x_6$ as the second step results in a function $f_{2357}$ for $S' = 2357$. The remaining problem is then written as

$$g_{1234} = \sum_{57}^{2357} f_{2357} f_{1457}.$$ 

Now 5 and 7 are symmetric. Eliminating $x_7$ first results in a function $f_{12345}$. Finally, eliminating $x_5$ results in the function $f_{1234} = g_{1234}$. We conclude that the elimination order $8 \prec 6 \prec 7 \prec 5$ yields the induced total size $m^5 + m^5 + m^6 + m^5$ which is slightly less than what we got for the order we considered first.

\[\square\]

### 3.3 Boosting the variable elimination algorithm

Is the variable elimination algorithm optimal when it is run with an optimal elimination order? For commutative semirings in general this seems to be a largely open problem. Interestingly, for commutative rings the answer is
negative. To see this, just notice that the ordinary multiplication of two $m \times m$ matrices is an SMPF problem (class). On one hand, it is easy to see that for the matrix multiplication problem the maximum width equals three, and hence, the total size and the arithmetic complexity of the algorithm is $\Omega(m^3)$. On the other hand, the fast matrix multiplication methods require only $O(m^\omega)$ arithmetic operations, where $\omega < 3$ due to Strassen’s [Str69] ingenious recursive algorithm.

This suggests that the efficiency of the variable elimination can be boosted by applying the fast matrix multiplication methods. In fact, one might exploit a relatively wide repertoire of fast matrix multiplication results. Below we recall some most recent and interesting asymptotic results; their practical significance is not clear.

**Theorem 3.17** ([CW90]) There is an algorithm for multiplying two $m \times m$ square matrices with complexity $O(m^{2.38})$. \hfill \Box

**Theorem 3.18** ([Cop97]) Let $\tau < 0.294$. For arbitrary $\varepsilon > 0$ there is an algorithm for multiplying an $m \times m$ square matrix by an $m \times m^\tau$ rectangular matrix with complexity $O(m^{2+\varepsilon})$. \hfill \Box

**Theorem 3.19** ([HP98]) There is an algorithm for multiplying an $m \times m$ square matrix by an $m \times m^2$ rectangular matrix with complexity $O(m^{3.34})$. \hfill \Box

In this section we discuss how these results can be applied to boost the variable elimination algorithm in the context of commutative rings. As far as the author knows, the general connections of rectangular matrix multiplication and sum-product problems have not been studied earlier. Thus, the conceptual and technical development in the rest of this section, may present some novel, yet fairly straightforward, findings.

How much can the elimination algorithm be accelerated by the fast matrix multiplication methods? An obvious way to (asymptotically) accelerate the elimination algorithm is to replace the single-variable marginalization steps by Strassen-type fast matrix multiplication algorithms. This approach is possible, though nontrivial, even when the SMPF problem is defined on a semiring that is not itself a ring. In that case, one may be able to embed the semiring into a ring with a little computational overhead. This is the idea, for instance, in the fast Boolean matrix multiplication and its extensions recently developed for the all-pairs-shortest-paths problem [AGM97, Zwi02]. However, by this approach one may hope to get only a marginal improvement in the variable elimination algorithm when the primitive domains $\mathcal{A}_i$ are large. Especially, if the variables are binary-valued, i.e., $|\mathcal{A}_i| = 2$ for all $i$, then no acceleration is possible.
We next show that based on a different technique, the ordinary variable elimination algorithm may sometimes be significantly accelerated. The trick we use is extremely simple. It is based on the observation that in some cases the sequential elimination of variables requires no fewer arithmetic operations than if eliminating the variables as a block; see Example 3.23 below. But the opposite may hold: a block of variables can be eliminated efficiently using fast matrix multiplication. A more precise description of the method follows.

**Definition 3.20** A blocked elimination order on a set \( S \) is a pair \((B, \prec)\), where \( B \) is a partition of \( S \) and \( \prec \) is a linear order on \( B \).

The algorithm is a straightforward extension of the ordinary variable elimination algorithm (Algorithm 3.9).

**Algorithm 3.21** (the boosted variable elimination algorithm)

**Input** an instance \( \mathcal{F} = \{ f_T : T \in \mathcal{T} \} \) of \( \text{SMPF}(A, R, T, S) \) and a blocked elimination order \((B, \prec)\) on \( S^c \)

**Output** the marginal function as described in Definition 3.5

**Method**

1. for each \( B \in B \) in the order \( \prec \) do
2. \( \mathcal{T}' \leftarrow \{ T \in \mathcal{T} : B \cap T \neq \emptyset \} \) % local scopes that overlap \( B \)
3. \( S' \leftarrow \bigcup \mathcal{T}' - B\) % covered variables not to be eliminated
4. \( \mathcal{T}' \leftarrow \mathcal{T}' \cup \{ T \in \mathcal{T} : T \subseteq S' \} \) % to avoid duplicates
5. \( f_{S'} \leftarrow \sum_B \prod_{T \in \mathcal{T}} f_T \) % eliminate \( x_B \), possibly via fast MM
6. \( \mathcal{T} \leftarrow \mathcal{T} \cup \{ S' \} \) % update local scopes
7. \( \mathcal{F} \leftarrow \mathcal{F} \cup \{ f_{S'} \} - \{ f_T : T \in \mathcal{T}' \} \) % update functions, used implicitly
8. \( \text{return } \prod_{T \in \mathcal{T}} f_T \)

**Theorem 3.22** Algorithm 3.21 is correct.

**Proof** Similar to the proof of correctness of the variable elimination algorithm (Theorem 3.11) and therefore omitted.

When is the above algorithm more efficient than the ordinary variable elimination algorithm? The possible gain depends on the structure of the subproblems formed and solved at each elimination round. To analyze this, we have to consider in more detail the elimination step on line 5 of Algorithm 3.21. Before doing this, let us revisit our concrete example. In that example problem we get a considerable reduction in the arithmetic complexity.
Example 3.23 We continue Example 3.16. Recall that the problem is to compute the marginal function

\[ g_{1234} = \sum_{5678} f_{158} f_{256} f_{367} f_{478}. \]

Also recall that in Example 3.16 we showed that the variable elimination algorithm requires \( O(m^6) \) arithmetic operations.

Consider now the boosted elimination algorithm on the blocked elimination order \( 8 \prec 6 \prec 57 \). Recall, again from Example 3.16, that the elimination of the variables \( x_8 \) and \( x_6 \) can be done using \( O(m^5) \) arithmetic operations, and that the remaining subproblem is to compute

\[ g_{1234} = \sum_{57} f_{1457} f_{2357}. \]

Now treating the block 57 as a “super variable” is advantageous, since by the fast matrix multiplication this last elimination step requires only \( O((m^2)^\omega) \) arithmetic operations. Thus, by Theorem 3.17 the total arithmetic complexity of the boosted elimination algorithm is \( O(m^5) \), which is less than the complexity \( O(m^6) \) of the ordinary algorithm.

In fact, the complexity is even better. Namely, the elimination of the variable \( x_8 \), similarly \( x_6 \), can be viewed as a multiplication of an \( m^2 \times m \) matrix by an \( m \times m^2 \) matrix. Hence, by Theorem 3.19 the first two elimination steps together take \( O(m^{4.34}) \) operations. Thus, the last block elimination dominates, yielding a total complexity of \( O(m^{4.76}) \).

We now consider the elimination step of the boosted variable elimination algorithm in more detail. Consider first an ideal case where maximum acceleration is possible. This is the case if \( S' \) happens to have a partition \( \{A, C\} \) such that the product of local functions decomposes as

\[ \prod_{T \in \mathcal{T}'} f_T = \left( \prod_{T \in \mathcal{T}' : T \subseteq A} f_T \right) \left( \prod_{T \in \mathcal{T}' : T \subseteq C} f_T \right). \]

This gives a matrix multiplication representation to the elimination step, and fast square or rectangular multiplication methods can be applied. Unfortunately, in general, there need not exist such a partition, in which case the situation is more complicated. Consider, for example, an elimination step of the form

\[ f_{123} \leftarrow \sum_{4} f_{124} f_{134} f_{234}. \quad (3.2) \]
In this example no acceleration seems to be possible. One might first think that it is the graph structure of the problem that determines whether boosting by fast matrix multiplication is possible or not.\(^3\) However, this view is incorrect. For example, the graph structure of

\[
f_{123} \leftarrow \sum_{4} f_{12} f_{13} f_{23} f_{14} f_{24} f_{34}
\]

is the same as the structure of the above problem (3.2). Yet, the elimination step can be reorganized as

\[
f_{123} \leftarrow f_{12} f_{13} f_{23} \sum_{4} (f_{14} f_{24})(f_{34}),
\]

which essentially corresponds to a rectangular matrix multiplication.

We conclude this discussion of the boosted elimination step with some summarizing remarks. The pattern how the fast matrix multiplication is most efficiently applied to a particular subproblem can be complicated. It is determined by the problem structure (a hyper graph), but not solely by the graph structure. Accordingly, finding the way to apply the matrix multiplication, if possible, is rather a matter of the structure analysis, than an issue to be solved by the elimination algorithm for a fixed SMPF problem. Consequently, finding the most efficient matrix multiplication patterns, if any, should be incorporated into the preliminary phase, where problem structure is analyzed in order to find a good blocked elimination order. Also, note that the best blocked elimination order need not be a “segmentation” of an optimal variable elimination order.

We next sketch a simple algorithm for finding an optimal blocked elimination order. Let \(S \subseteq N = \{1, \ldots, n\}\) be the marginal scope. For any \(A \subseteq S^c\) let \(g(A)\) denote the minimum number of arithmetic operations required by the boosted elimination algorithm for eliminating the variables over \(A\) when run on a blocked order whose first \(|A|\) elements form the set \(A\). Furthermore, for any \(B \subseteq S^c - A\) let \(f(B, A)\) denote the number of operations required at the subsequent elimination step when the variables over \(B\) are eliminated as a block. We note that the number \(f(B, A)\) does not depend on the specific blocked order chosen for the preceding variables over \(A\). Therefore, for all \(A \subseteq S^c\) we have the recurrence

\[
g(A) = \min_{B \subseteq A} \{f(B, A - B) + g(A - B)\}.
\]

\(^3\)Note that optimal elimination orders for the ordinary variable elimination algorithm are determined solely by the graph structure (and domain sizes).
3.4 Transformation problems

On the boundary, of course, $g(\emptyset) = 0$ by the definition of $g$. Assume now that the function $f$ is precomputed. Then the above recurrence for $g(S^c)$ can be solved and an optimal blocked elimination order can be extracted by standard dynamic programming techniques. The running time can be shown to be $O(3^{S^c})$ and hence $O(3^n)$. However, the total complexity is dominated by the complexity of computing the function $f$, since there are about $3^{S^c}$ values to be computed. It seems that the values can be computed in a time that is a low-degree polynomial in the number $n$ of variables and in the number $|\mathcal{T}|$ of local scopes.

Note that the above sketch remains valid with few changes if the size of a variable block is bounded by some parameter, say $\ell$. Specially in the case $\ell = 1$, we obtain an algorithm that finds an optimal elimination order for the ordinary variable elimination algorithm. Assuming $|\mathcal{T}| = O(n)$ a calculation suggests the total time complexity of $O(n2^n)$ for this special case.

Let us finally consider how much fast matrix multiplications can boost the elimination algorithm when all variables are binary-valued. Not surprisingly, it is easy to find an infinite sequence of SMPF problems on binary variables so that the reduction in arithmetic complexity is significant. Just treat the multiplication of two $2^k \times 2^k$ matrices as a marginalization of a product of two $2k$-dimensional functions. The improvement is from $O(2^{3k})$ to $O(2^{2k})$. This construction is, of course, trivial, and just a matter of problem representation. Namely, one could, in the first place, represent the problem in terms of three variables, each with $2^k$ possible values. However, this argument is less valid if the matrix product representation, though present, is hidden due to many smaller local scopes.

3.4 Transformation problems

We next consider a special type of sum-product problems we call transformation problems. Examples of transformation problems are the discrete Fourier transform, the Yates transform, and the Möbius transform on subset lattice. We discuss these applications after defining the general transformation problem in which a given function is to be transformed to another similar function by a linear operator.

However, we are here interested in transformations that enjoy a structure that can be exploited when computing the transform. This leads us to consider an SMPF problem whose structure is characterized by a “sliding” sequence of local scopes. A feature of this formulation is that the function to be transformed will be included in a function with a slightly larger scope;
Figure 3.2: Two graphical visualizations of the structure of a sliding transform. (a) A graph where a node corresponds to a variable or a local function (boxes, not labeled). There is an edge between a variable node and a function node if the variable is an argument of the function. (b) A graph where a node corresponds to a variable. There is an edge between two nodes if some local function has the corresponding variables as arguments. In both visualizations the marginal variables are indicated by circles.

see Example 3.26 below. Many interesting well-known transformations, including those mentioned above, can be seen as refinements of a coarser, general transformation problem.

It is convenient to define the general transformation problem (class) in a canonical form, i.e., for a specific labeling of the variables ignoring the isomorphic structures.

**Definition 3.24** A problem $\text{SMPF}(\mathcal{A}, \mathcal{R}, \mathcal{T}, S)$ is said to be a sliding transform if the number variables is even, $n = 2k$, and $\mathcal{T} = \{\{i, i+1, \ldots, i+k\} : i = 1, \ldots, k\}$ and $S = \{1, \ldots, k\}$.

We may call an SMPF problem a sliding transform if it matches to the above definition for some permutation of the variable indexes. See Figure 3.2 for the graph structure of a sliding transform on $n = 8$ variables.

How fast can one compute a sliding transform? The following result says that by taking advantage of the problem structure one obtains a linear algorithm. Note that the size of the input in a sliding transform on $2k$ variables is of size $\Theta(k m^{k+1})$, provided that each primitive domain has $m$ elements.
3.4 Transformation problems

**Theorem 3.25** The arithmetic complexity of sliding transforms is $O(k m^{k+1})$, provided $|A_i| \leq m$.

**Proof** We consider running the variable elimination on the elimination order $n \prec n-1 \prec \cdots \prec k+1$. Clearly, the elimination of variable $x_n$ corresponds to the elimination step

$$f_{\{n-k,n-k+1,\ldots,n-1\}} \leftarrow \sum_{n} f_{\{n-k,n-k+1,\ldots,n-1,n\}},$$

since $n$ is included in exactly one local scope. Iteratively, for $j < n$, the elimination of $x_j$ corresponds to

$$f_{\{j-k,j-k+1,\ldots,j-1\}} \leftarrow \sum_{j} f_{\{j-j-k,j-k+1,\ldots,j-1,j\}} f_{\{j-k+1,j-k+2,\ldots,j\}}.$$

Thus, by Proposition 3.13, the arithmetic complexity is bounded by $O(\sum_{j=k+1}^{n} |A_{\{j-k,j-k+1,\ldots,j-1,j\}}|)$, which is $O(k m^{k+1})$ since $|A_i| \leq m$. $\square$

Many important computational problems have the structure of a sliding transform and can be solved by the variable elimination algorithm. However, the appropriate SMPF problem representation is not always apparent. A well-known example is the discrete Fourier transform and the fast Fourier transform.

**Example 3.26** The discrete Fourier transform (DFT) of a sequence $f(0), \ldots, f(r-1)$ of complex numbers is the sequence

$$g(s) = \sum_{t=0}^{r-1} f(t) \alpha_r(s t), \quad s = 0, 1, \ldots, r-1,$$

where $\alpha_r(q) = e^{q 2\pi i / r}$, i.e., the $r$th complex root of unity to the power of $q$. The direct computation takes $O(r^2)$ time. However, the fast Fourier transform algorithm (FFT) solves the problem in $O(r \log r)$ time.

We next show that the DFT can be viewed as a special case of a sliding transform, and respectively, the FFT can be viewed as a variable elimination algorithm; a slightly different derivation (for the case $k = 3$) is given in [KFL01]. For simplicity, we let $r = 2^k$ for some $k$. We put $n = 2k$, and introduce binary variables $x_1, \ldots, x_n$ such that $(x_k, \ldots, x_1)$ and $(x_{k+1}, \ldots, x_n)$ are the bit representations of the indices $s$ and $t$ respectively. That is, $s = \sum_{h=1}^{k} 2^{k-1} x_{k+1-h}$ and $t = \sum_{h=1}^{k} 2^{k-1} x_{k+h}$. The key fact used in the
FFT is that $\alpha_r(q) = \alpha_r(q \mod r)$. We observe that

$$s \cdot t \mod r = \left( \sum_{h=1}^{k} 2^{h-1} x_{k+1-h} \right) \left( \sum_{h=1}^{k} 2^{h-1} x_{k+h} \right) \mod 2^k$$

$$= \sum_{h=1}^{k} 2^{h-1} x_{k+1-h} \sum_{h'=1}^{k+1-h} 2^{h'-1} x_{k+h'}.$$ 

Hence,

$$\alpha_r(s \cdot t) = \prod_{h=1}^{k} \alpha_r(2^{h-1} x_{k+1-h} \sum_{h'=1}^{k+1-h} 2^{h'-1} x_{k+h'}).$$

This suggests defining $k$ local scopes and functions as follows. We let $T_j = \{j, j+1, \ldots, j+k\}$ for $j = 1, \ldots, k$. For $j = 1, \ldots, k - 1$ we define

$$f_{T_j}(x_{T_j}) = \alpha_r(2^{k-j} x_j \sum_{h=1}^{j} 2^{h-1} x_{h+k}).$$

Note that only some of the variables appearing on the left hand side appear also on the right hand side. For $j = k$ we include also the function $f$ by setting

$$f_{T_k}(x_{T_k}) = f(x_{k+1}, \ldots, x_n) \alpha_r(x_k \sum_{h=1}^{k} 2^{h-1} x_{h+k}),$$

where $f(x_{k+1}, \ldots, x_n) = f(t)$. Note that in this case all the variables appearing on the left hand side appear also on the right hand side.

It is now easy to verify that

$$g = \sum_{\{k+1, \ldots, n\}} \prod_{j=1}^{k} f_{T_j}.$$ 

Furthermore, we see that the DFT of any function $f$ can be represented as an instance of a sliding transform. By Theorem 3.25 the DFT can be computed in time $O(k \cdot 2^{k+1})$ which is $O(r \log r).$

In the above example we saw that it is already the coarser structure of a sliding transform that yields the computational efficiency. In the case of DFT, for instance, the finer problem structure does not give further reduction in the complexity.
However, there are also important transformation problems which enjoy such a finer structure that makes it possible to reduce the computational complexity. The following definition formulates a problem structure where the linear transform operator decomposes into a product of pairwise factors. In matrix algebra such a linear mapping corresponds to a square matrix that is a tensor (or Kronecker) product of smaller matrices. We borrow the name for this type of transform from a related computational technique known as the Yates algorithm due to Yates [Yat37].

**Definition 3.27** A problem SMPF($\mathcal{A}, \mathcal{R}, \mathcal{T}, S$) is said to be a *Yates transform* if the number of variables is even, $n = 2k$, and if $\mathcal{T} = \{\{i, i+k\} : i = 1, \ldots, k\} \cup \{\{k+1, \ldots, 2k\}\}$ and $S = \{1, \ldots, k\}$. That is, the marginal function, to be computed, decomposes as

$$g_{\{1,\ldots,k\}}(x_1, \ldots, x_k) = \sum_{x_{k+1}, \ldots, x_n} f_{\{k+1,\ldots,n\}}(x_{k+1}, \ldots, x_n) \prod_{i=1}^{k} f_{\{i,i+k\}}(x_i, x_{i+k}),$$

for all values of variables $x_1, \ldots, x_n$.

We may call an SMPF problem a Yates transform if it matches to the above definition for some permutation of the variable indexes.

**Example 3.28** Let $x_1, \ldots, x_k$ and $y_1, \ldots, y_k$ be variables with the binary domain $\{0,1\}$. The *Hadamard transform* of a function $f$ of variables $y_1, \ldots, y_k$ is the function $g$ defined by

$$g(x_1, \ldots, x_k) = \sum_{y_1, \ldots, y_k} f(y_1, \ldots, y_k)(-1)^{\sum_{i=1}^{k} x_i y_i},$$

with ordinary addition and multiplication operations. Clearly this is a special case of a Yates transform. The Hadamard transform has applications, for instance, in digital signal and image processing (see, e.g., [YH97]).

We also note that a Yates transform can be represented nicely in a matrix form:

**Remark 3.29** For $i = 1, \ldots, k$ let $A_i$ be an $m_i \times m_i'$ matrix. Denote by $A$ their tensor product $A_1 \otimes \cdots \otimes A_k$ (also called direct or Kronecker product; for definition and properties see, e.g., [MM92]). In addition, let $u$ be a row vector of length $m = m_1 \cdots m_k$. Then the matrix–vector multiplication $uA$ is a Yates transform. Likewise, if $v$ is a column vector of length $m' = m_1' \cdots m_k'$, then $Av$ is a Yates transform. Conversely, any instance of a Yates transform can be viewed as a matrix multiplication of this type.
It is obvious that any instance of a Yates transform can be (efficiently) represented as an instance of a sliding transform. Thus, the time complexity of a Yates transform is at most that of the corresponding sliding transform. For example, computing a Hadamard transform takes $O(k 2^k)$ time. Interestingly, for large primitive domains a Yates transform can be slightly accelerated by the fast matrix multiplication techniques. In the proof of the following result we derive the Yates algorithm.

**Theorem 3.30** The arithmetic complexity of Yates transforms is $O(k m^{k+1})$, provided that the cardinality of the primitive domains is at most $m$. In a ring the complexity reduces to $O(k m^{k+0.30})$ for $k \geq 3$ and to $O(m^{2.38})$ for $k = 2$.

**Proof** The first statement is obvious, since any problem instance can be efficiently mapped to an instance of a sliding transform problem with the same number of variables and the same primitive domains.

Suppose then that the arithmetic operations are performed on a ring. We consider running the boosted variable elimination algorithm on the (blocked) elimination order $n \prec n-1 \prec \cdots \prec k+1$. Clearly, the elimination of variable $x_n$ corresponds to the elimination step

$$f_{\{n-k,\ldots,n-1\}} \leftarrow \sum_{n} f_{\{n-k,n\}} f_{\{n-k+1,\ldots,n\}}.$$  

Iteratively, for $j < n$, the elimination of $x_j$ corresponds similarly to

$$f_{\{j-k,\ldots,j-1\}} \leftarrow \sum_{j} f_{\{j-k,j\}} f_{\{j-k+1,\ldots,j\}}.$$  

Thus, each of the $k$ elimination steps corresponds to a rectangular matrix multiplication. The sizes of the matrices are at most $m \times m$ and $m \times m^{k-1}$. By Theorems 3.17 and 3.19 the claimed complexity bound follows. This special case of the variable elimination algorithm is known as the Yates algorithm [Yat37], originally designed for binary variables (i.e., $m = 2$).

The third generic type of transform we consider in this section is the Möbius transform. We start by studying certain key features of rather generally formulated Möbius transforms. Similarly to a sliding transform or a Yates transform, a Möbius transform maps a function to another function by a linear operator. The difference is that in a Möbius transform the operator is defined by a graph. After briefly studying an abstract form, we move to a special case where the graph is a subset lattice (definition is given below). An interesting and somewhat unobvious connection to the Yates
transform is found. This observation leads to the fast M"{o}bius transform (FMT) algorithm. The algorithm itself is not new. A similar, yet much heavier and somewhat nontransparent presentation of the FMT is given in [KS91, Ken91]. Here we give a simplified and concise description.

An abstract formulation of the M"{o}bius transform is as follows. Let $V$ be a set and $R \subseteq V \times V$ a relation on $V$. Let $W$ be an arbitrary linear space over $\mathbb{R}$ (the field of real numbers).

**Definition 3.31** The M"{o}bius transform of a function $f : V \to W$ on a graph $G = (V, R)$ is the function $f^G : V \to W$ defined by

$$f^G(s) = \sum_{t \in R^{-1}(s)} f(t) \quad \text{for all } s \in V. \quad (3.5)$$

The above definition is sometimes given only for graphs that are partially ordered sets (posets), possibly because then we have the following important result known as the M"{o}bius inversion formula; we omit the proof (see [Aig79, Ch. 4]).

**Theorem 3.32** For any poset $G = (V, R)$ there exists a unique function $\mu_G : V \times V \to \mathbb{R}$ such that for all $f$,

$$f(s) = \sum_{t \in R^{-1}(s)} \mu_G(t, s) f^G(t) \quad \text{for all } s \in V. \quad (3.6)$$

The function $\mu_G$ above is called the M"{o}bius function of $G$. Note that by M"{o}bius transform and M"{o}bius inversion we refer to the two functionals defined respectively in (3.5) and (3.6). Yet, in order to keep the notation relatively simple, we avoid introducing any explicit notation for these two mappings.

The above defined abstract M"{o}bius transform has important special cases. Perhaps the best-known instance is the M"{o}bius transform in number theory. In this case, the set $V$ is the set of integers, the relation $R$ corresponds to "divisibility", and the linear space $W$ is simply the set of complex numbers. Another well-known instance is the M"{o}bius transform on a subset lattice, mainly used in the context of belief functions (see, e.g., [KS91]. We now focus on the M"{o}bius transforms on subset lattice.

**Definition 3.33** Let $N$ be a finite set. The subset lattice on $N$ is the graph $(V, R)$, where $V = \mathcal{P}(N)$ is the collection of subsets of $N$ and $R = \{(T, S) : T \subseteq S \subseteq N\}$ is the inclusion relation.
Obviously a subset lattice is a poset. The following well-known result gives the associated Möbius function (see, e.g., [GW77, p. 330], [Aig79, pp. 148–153]).

**Proposition 3.34** If $G$ is a subset lattice on $N$, then the Möbius function of $G$ is given by $\mu_G(T, S) = (-1)^{|S-T|}$ for all $T \subseteq S \subseteq N$. □

There is an efficient way to compute a Möbius transform of a given function, known as the *fast Möbius transform*. The key idea is to represent the subsets of $N$ as binary vectors. That is, a subset $S \subseteq N$ is understood as a binary vector $(S_1, \ldots, S_k) \in \{0, 1\}^k$, where $S_i = 1$ iff $i \in S$. (Note that here $S$ and $T$ should not be confused with the marginal and local scopes introduced in the previous sections. Yet, this notation anticipates that these $S$ and $T$ will also serve as variable scopes in the context of Möbius transforms; see below and Section 6.3.)

**Algorithm 3.35** (the fast Möbius transform)

**Input** A function $f : \mathcal{P}(N) \to W$, where $N = \{1, \ldots, k\}$ and $W$ is a linear space.

**Output** The transformed function $f^G$.

**Method**

1. $f_k \leftarrow f$
2. for $j = k, k-1, \ldots, 1$ do
3.   for each $S \subseteq N$ do
4.     $f_{j-1}(S_1, \ldots, S_k) \leftarrow \sum_{T_j \subseteq S_j} f_j(S_1, \ldots, S_{j-1}, T_j, S_{j+1}, \ldots, S_k)$
5. return $f_0$

**Theorem 3.36** Algorithm 3.35 works correctly.

**Proof** Straightforward induction gives that the functions $f_j$ computed by the algorithm satisfy

$$f_{j-1}(S_1, \ldots, S_k) = \sum_{T_j \leq S_j} \cdots \sum_{T_k \leq S_k} f(S_1, \ldots, S_{j-1}, T_j, \ldots, T_k)$$

for all binary vectors $(S_1, \ldots, S_k)$. Hence, in particular,

$$f_0(S) = f_0(S_1, \ldots, S_k) = \sum_{T_1 \leq S_1} \cdots \sum_{T_k \leq S_k} f(T_1, \ldots, T_k) = \sum_{T \subseteq S} f(T),$$

completing the proof. □

We consider the complexity of Möbius transforms and Möbius inversions with respect to two measures. One is the number of arithmetic operations on the linear space $W$. The second is the total cost of the arithmetic
operations when allowing the cost of an addition of \( f(S) \) and \( f(T) \) to depend exponentially on the cardinalities of the sets \( S \) and \( T \); this makes sense when \( f(S) \) and \( f(T) \) are functions over scopes \( S \) and \( T \), respectively.

**Theorem 3.37** Let \( N = \{1, \ldots, k\} \). Möbius transforms and inversions on a subset lattice \( (\mathcal{P}(N), \subseteq) \) can be computed using \( \mathcal{O}(k \, 2^k) \) arithmetic operations on the space \( W \).

If the time complexity of an addition \( f(S) + f(T) \) is bounded by \( m^{|S \cup T|} \) for some \( m \) for all \( S, T \subseteq X \), then Möbius transformation and inversion can be computed in time \( \mathcal{O}(k(m+1)^k) \).

**Proof** We note that the arithmetic complexity follows from the time complexity in the special case of \( m = 1 \). Therefore, it is sufficient to consider the case where the cost of an addition \( f(S) + f(T) \) is bounded by \( m^{|S \cup T|} \).

We first consider the complexity of a Möbius transform. The analysis of Algorithm 3.35 is straightforward. At each of the \( k \) iterations the time requirement is bounded by

\[
\sum_{S \subseteq N} m^{|S|} = \sum_{h=0}^{k} \binom{k}{h} m^h = (m + 1)^k,
\]

up to a constant factor. Namely, the set represented by the binary vector \( (S_1, \ldots, S_{j-1}, T_j, S_{j+1}, \ldots, S_k) \) must be a subset of \( S \) (not necessarily a proper subset). Thus, we get the claimed bound \( \mathcal{O}(k(m+1)^k) \).

Consider then the complexity of a Möbius inversion. It is easy to see that Algorithm 3.35 can be easily modified so that it computes the inversion instead of the transform. Namely, replace line 4 by

\[
f_{j-1}(S_1, \ldots, S_k) \leftarrow \sum_{T_j \subseteq S_j} (-1)^{T_j} f_j(S_1, \ldots, S_{j-1}, T_j; S_{j+1}, \ldots, S_k).
\]

Then, by a simple induction,

\[
f_0(S) = \sum_{T \subseteq S} (-1)^{\sum_{i=1}^{k} S_i - T_i} f^G(T) = \sum_{T \subseteq S} (-1)^{|S \setminus T|} f^G(T) = f(S),
\]

assuming that \( f^G \) (not \( f \)) is given as an input to the algorithm. The complexity is obtained similarly as for the Möbius transform.

The binary vector representation of the subsets reveals that any Möbius transform and inversion essentially have the structure of a Yates transform. Why did we not derive the fast Möbius transform algorithm and the complexity results based on Yates transform? The reason lies in the fact that
we defined Yates transforms as SMPF problems for semirings. In contrast, Möbius transforms are defined for a linear space over a field of scalars, and furthermore, not in the SMPF form. Thus, some work is needed to formally match these different representations. An alternative to the above derivation is to construct an explicit mapping from an Möbius transform or inversion instance to a Yates transform instance. For Möbius transforms this is straightforward as the (vector or function) elements of a linear space can be treated as elements of a semiring. For Möbius inversions, however, an annoying fact is that the scalar coefficients, as well, need to be converted to semiring elements, with appropriate definitions of semiring addition and multiplication. That said, the above given rather separate descriptions of the Yates algorithm and the fast Möbius transform (and inversion) can be arguably justified.

3.5 On the limitations of the variable elimination algorithm

It is important to note that the SMPF formalism can express only certain types of structural features of computational problems. In applications there is often more structure available. Below we discuss some most important structural features that cannot be naturally handled solely by the SMPF formalism and the variable elimination algorithm.

(a) In SMPF problems the marginalization is supposed to be taken over a domain that is a Cartesian product of primitive domains, \( A_{S^c} = A_{i_1} \times \cdots \times A_{i_r} \) for \( S^c = \{i_1, \ldots, i_r\} \). A natural generalization asks for the marginalization over a subset of the Cartesian product, \( B_{S^c} \subset A_{S^c} \). We note, however, that any instance of such a problem can be represented as an instance of an SMPF problem by introducing an extra local function, the indicator function \( f_{S^c}(x_{S^c}) = 1_{\{x_{S^c} \in B_{S^c}\}} \). Such an indicator function may or may not decompose into a product of similar, local functions. In order to represent a problem in the SMPF form this type of analysis should be done beforehand.

(b) In SMPF problems the local functions are considered indeterminates, that is, arbitrary functions specified by an instance of the problem. However, in many cases some of the local functions are fixed and only some are considered indeterminates. This is the case, for example, with the discrete Fourier transform and the Hadamard transform. Note that item (a) above can be viewed as a special case of this type of additional structure due to “fixed local functions”.
3.5 On the limitations of the variable elimination algorithm

(c) It is not possible to express intrinsic structure of local functions in the SMPF formalism. Here we mean functions that are not necessarily totally fixed but that hold some structure, e.g., exchangeability of the arguments. Note that item (b) above can be viewed as a special case of this type of additional structure. This representational lack may be crucial. Namely, employing the variable elimination algorithm ignoring the intrinsic structure may be significantly suboptimal. Some examples are given below.

(d) In SMPF problems the local functions are considered independently. However, in some applications, one function may be associated with many local scopes. For example, in the Hadamard transform the functions $\alpha(x_i, y_i) = (-1)^{x_i y_i}$ are the same for all pairs $(x_i, y_i)$ of variables. Employing the variable elimination algorithm but ignoring this type of structure may be suboptimal. As an example consider the computation of the $k$th power of an $m \times m$ matrix (over a semiring). The obvious algorithm takes $O(m^3 k)$ time, but we know that a doubling technique reduces this to $O(m^3 \log k)$. See also Example 3.40 below.

The above remarks (a–d) recognize that a sum-product type problem, arising in some application, should be analyzed carefully to find an appropriate SMPF formulation. Such an analysis may lead to introduction of new variables, as we did to get the fast Fourier transform and the fast Möbius transform. It may also be that the whole problem is better solved in some “dual space”. A good example of this is the fast convolution algorithm for the discrete convolution problem.

Example 3.38 The discrete convolution of two sequences $a(0), \ldots, a(r-1)$ and $b(0), \ldots, b(r-1)$ of complex numbers is the sequence

$$c(s) = \sum_{t=0}^{s} a(t) b(s-t), \quad s = 0, 1, \ldots, r-1.$$ 

This can be trivially mapped to an SMPF problem, but the variable elimination algorithm would take $O(r^2)$ time. The variable elimination algorithm is not able to exploit the intrinsic structure of the local function $b$. However, there exists a well-known more efficient algorithm that uses the fast Fourier transform. Namely, the convolution theorem (see, e.g., [CLR96, p. 790]) states that

$$c = \text{DFT}^{-1}(\text{DFT}(a) \cdot \text{DFT}(b)),$$
where $a, b$, and $c$ are padded with 0's to length $2r$. Here DFT denotes the discrete Fourier transform mapping and "·" denotes the inner product. Using the fast Fourier transform (and inversion) the convolution can be computed in time $O(r \log r)$.

Another example of the limitations of the SMPF formalism and the variable elimination algorithm is provided by the matrix determinant and permanent.

**Example 3.39** Let $A = (a_{ij})$ be an $n \times n$ matrix. The **determinant** of $A$ is defined as

$$
\text{det}(A) = \sum_{\sigma \in S_n} \text{sgn}(\sigma) a_{1\sigma(1)} a_{2\sigma(2)} \cdots a_{n\sigma(n)},
$$

where $S_n$ is the set of permutations of the numbers $\{1, \ldots, n\}$ and $\text{sgn}(\sigma)$ is the sign of the permutation $\sigma$. The **permanent** of $A$ is defined as

$$
\text{per}(A) = \sum_{\sigma \in S_n} a_{1\sigma(1)} a_{2\sigma(2)} \cdots a_{n\sigma(n)}.
$$

The computation of the determinant, e.g., via Gaussian elimination, takes polynomial time. In contrast, the fastest algorithm developed for the computation of the permanent takes $O(n 2^n)$ time [Knu98, p. 499]. At first glance this difference may be surprising, since the definition of the determinant is seemingly more complex than that of the permanent.

An algorithm that computes the permanent in time $O(n 2^n)$ is sketched as follows. For nonempty subsets $S \subseteq N = \{1, \ldots, n\}$ we define recursively

$$
g(S) = \sum_{j \in S} a_{|S|j} g(S - \{j\}),
$$

where on the boundary we define $g(\emptyset) = 1$. Note that here $j$ corresponds to $\sigma(|S|)$. It is not difficult to verify that $g(N) = \text{per}(A)$. A simple calculation yields the claimed time complexity for a straightforward dynamic programming algorithm.

We note that this algorithm performs variable elimination, first eliminating $\sigma(1)$, then $\sigma(2)$, and so forth. Yet, there is no immediate way to define local functions to get an SMPF instance. Even if this was possible, the variable elimination algorithm would not be able to exploit the intrinsic structure of the local functions. The fact that the similarly defined determinant can be computed very efficiently shows that the intrinsic structure may sometimes play a surprisingly important role.
3.5 On the limitations of the variable elimination algorithm

Our last example in this section considers a sum-product type computational problem where various techniques are applied to derive an efficient algorithm. Here we describe and improve an algorithm presented recently in [KBM+03].

**Example 3.40** Kontkanen et al. [KBM+03] consider the computational problem of evaluating

\[
R_{K,N} = \sum_{x_1 + \cdots + x_K = N} \frac{N!}{x_1! \cdots x_K!} \prod_{i=1}^{K} \left( \frac{x_i}{N} \right)^{x_i},
\]

where \(K\) and \(N\) are positive integers. In a special context of prediction under the principle of worst case relative utility (see Section 2.1), this expression has an interpretation as the normalizing constant of the maximum likelihood distribution for the multinomial distribution, called the *regret* in [KBM+03]. They give an algorithm that achieves the time complexity of \(O(N^2 \log K)\) assuming that certain operations with rational numbers can be performed in a constant (amortized) time.

Here we derive Kontkanen’s et al. [KBM+03] algorithm from a slightly more generic perspective. We also improve the algorithm significantly using the fast convolution algorithm. Define first

\[
f(y) = \frac{(N!)^{1/K}}{y!} \left( \frac{y}{N} \right)^y \quad \text{for all } y = 0, 1, \ldots, N.
\]

Also, for all \(j = 1, 2, \ldots, K\) define the function \(g_j\) by

\[
g_j(m) = \sum_{x_1 + \cdots + x_j = m} \prod_{i=1}^{j} f(x_i) \quad \text{for all } m = 0, 1, \ldots, N.
\]

It is now easy to verify that for all integers \(K_1, K_2 \geq 1\) such that \(K_1 + K_2 = K\), we have recursively,

\[
R_{K,N} = g_K(N) = \sum_{N_1 + N_2 = N} g_{K_1}(N_1)g_{K_2}(N_2),
\]

with the boundary \(g_1(m) = f(m)\).

How fast can we compute \(R_{K,N}\)? Suppose first that \(K\) is a power of two, say \(2^r\). Then it is clearly sufficient to compute the values \(g_{2^s}(m)\) for all \(m = 0, 1, \ldots, N\) and \(s = 0, \ldots, r\). By the above recursion equation, this can be done in \(O(N^2 \log K)\) steps, since the computation of a function \(g_{2^s}\) given \(g_{2^{s-1}}\) can obviously be done in \(O(N^2)\) steps. If \(K\) is not a power of
two, it is easy to show that less than \( \log K \) additional functions \( g_j \) need to be computed, thus yielding the same asymptotic complexity \( \mathcal{O}(N^2 \log K) \).

But a significant improvement is possible. We recognize that for any integers \( j, j_1, j_2 \geq 1 \) such that \( j_1 + j_2 = j \) the function \( g_j \) is the convolution of the functions \( g_{j_1} \) and \( g_{j_2} \). Therefore, given \( g_{j_1} \) and \( g_{j_2} \), the function \( g_j \) can be computed via the fast convolution algorithm (see Example 3.38 above) in \( \mathcal{O}(N \log N) \) steps. Thus, the complexity of computing \( R_{K,N} \) is \( \mathcal{O}(N \log N \log K) \).

We conclude this section by noting that the hard part in solving sum-product type problems is finding an appropriate problem representation. Once a good representation has been found, running the variable elimination algorithm, or its variant, is relatively easy.

### 3.6 Concluding remarks

The general sum-product problem formulation, SMPF, has many appealing features. Various “standard” problems can be represented as sum-product problems in a “direct and structure-preserving” way [SH96]. The generalized variable elimination algorithm provides a unifying approach to tackle these problems, often leading to efficient methods.

But there are also shortcomings. As we discussed in Section 3.5, it is not always apparent how a particular computational task is best represented as an SMPF problem. To get maximal gain of the sum-product framework in a fixed problem, one may have to alternate between different representations in a dynamic fashion. Though SMPF representations together with the variable elimination algorithm are effective building blocks, it seems that plenty of creativity may be needed when designing algorithms for a particular application. There are some potentially fruitful extensions of the sum-product framework (e.g., [SH02]) that we did not include in this review.

In this chapter we only studied exact methods. Since the intermediate functions computed by the elimination algorithm may involve many variables, exact computations are not always feasible. Approximate methods can offer a practical solution. However, it seems that a unifying approach for various approximative methods on different algebraic structures is difficult to find. For example, a complicating issue is how one should quantify approximation errors on different instances of the abstract semiring. In Chapter 4, a special family of Monte Carlo methods on the ordinary real number system is studied.

We observed that fast matrix multiplication methods can be used to reduce the complexity of the variable elimination algorithm. Since the
asymptotic analysis hides large constant factors, the fast methods do not have practical value until computers can handle very large matrices, say with 10,000 rows and columns. More importantly, the simple observation presented here can, at best, only lead to a polynomial complexity reduction, from $O(\gamma(n, m))$ to $O(\gamma(n, m)^{\omega/3})$, where $\omega$ is the matrix multiplication exponent ($\omega < 2.38$) and $\gamma(n, m)$ is the complexity of the ordinary variable elimination algorithm on a problem parametrized by the number of variables $n$ and the size of the primitive domains $m$. An open question is, can other algebraic techniques lead to more dramatic reductions. We formulate a related open problem below. In this chapter we only presented the simple observation; more general and detailed analysis is a topic of future research.

We finally formulate two concrete open problems. First consider a problem related to the application of fast matrix multiplication methods. Let $\{R, S, T\}$ be a partition of $\{1, \ldots, n\}$. Recall that the fast matrix multiplication methods apply ideally, when the task is to compute the marginal

$$g_{R \cup T} = \sum_S f_{R \cup S} f_{S \cup T},$$

for given functions $f_{R \cup S}$ and $f_{S \cup T}$. An open problem is to find an asymptotically efficient way to carry out the marginalization, given that the two functions factorize as

$$f_{R \cup S} = \prod_{r \in R} \prod_{s \in S} f_{\{r, s\}} \quad \text{and} \quad f_{S \cup T} = \prod_{s \in S} \prod_{t \in T} f_{\{s, t\}}.$$

We see that the fast (rectangular) matrix multiplication is more efficient than the naive variable elimination. Do more efficient (algebraic) methods exist?

The second problem is as follows: For all subsets $S$ of $\{1, \ldots, n\}$, compute

$$c(S) = \sum_{T \subseteq S} a(T) b(S - T),$$

where $a$ and $b$ are two given functions. We note that the Möbius transform and inversion are of this form for a special function $b$. Also, we note a seeming similarity to the discrete convolution of two vectors. Do there exist algorithms faster than the straightforward one?
Chapter 4

Integration by tempered importance sampling

Integration of a function is a key task in many scientific applications. Sometimes the integrand (the function to be integrated) is so complex that there exists no closed form solution or that analytic treatment is intractable. In such cases, one has to resort to approximations.

In this chapter, we study a family of stochastic approximation algorithms. Whereas the presentation of methods is fairly general, examples are taken from Bayesian statistics. We start, in Section 4.1, by defining the integration problem and discussing its relationship to sum-product problems (see Chapter 3). In Section 4.2, we review the fundamental Monte Carlo importance sampling procedure. A key difficulty in importance sampling is the finding of an efficient sampling distribution. Motivated by this we, in Section 4.3, introduce a sophisticated technique called tempered integration method. In Section 4.4, we show that this method is ideal to implement in conjunction with the Metropolis coupled Markov chain Monte Carlo sampling algorithm [Gey93]. In Section 4.5, we explore related methods. Concluding remarks are made in Section 4.6.

Connections to other parts of this thesis are tight. Tools presented here are, in a sense, complementary to the exact sum-product algorithms studied in Chapter 3. Strong motivation comes from the calculus needed for Bayesian reasoning. In Chapter 7, we apply the tempered integration method in the context of genetic models and also discuss some implementation details.
4.1 Integration problem

Loosely speaking, integration means summation of the values of a function over a set of arguments. The summation yields the integral of the function. Every definition of an integral is based on a particular measure. For instance, the Lebesgue integral is based on Lebesgue measure (an extension of length and area to more complicated sets). The ordinal arithmetic summation is based on the counting measure. In the following definition of the integration problem we allow mixing these two measures.

**Definition 4.1** Let \( n \) be a natural number. For each \( i = 1, \ldots, n \) let \( \mu_i \) be counting or Lebesgue measure and \( \mathcal{A}_i \) a \( \mu_i \)-measurable subset of \( \mathbb{R} \). The Cartesian product \( \mathcal{A} = \times_{i=1}^{n} \mathcal{A}_i \) together with the product measure \( \mu = \times_{i=1}^{n} \mu_i \) specify an evaluation problem: Given a function \( f : \mathcal{A} \to \mathbb{R} \), evaluate the integral \( I(f) \) defined by

\[
I(f) = \int_{\mathcal{A}} f(x) \mu(dx).
\]

A problem instance is specified by specifying the integrand \( f \). We assume that the sets \( \mathcal{A}_i \) are bounded and that \( f \) is nonnegative, i.e., \( f(x) \geq 0 \) for all \( x \in \mathcal{A} \). Further, we require that \( f \) is integrable, i.e., \( I(f) \) is finite. For convenience we also require that \( I(f) \neq 0 \).

Illustrative applications of integration are provided by Bayesian statistics. In the two examples below we use the notation and concepts introduced in Chapter 2.

**Example 4.2** Consider a Bayesian model \( p(x) \) where \( x \) consists of observable variables \( x_S \), continuous latent variables \( x_C \), and discrete latent variables \( x_D \), with domains \( \mathcal{A}_S, \mathcal{A}_C, \) and \( \mathcal{A}_D \), respectively. Then the posterior expectation of a nonnegative function \( g(x_{CD}) \) is given by

\[
E_{\text{post}}(g) = \sum_{x_D \in \mathcal{A}_D} \int_{\mathcal{A}_C} p(x_{CD} \mid x_S) g(x_{CD}) dx_C.
\]

Clearly, this equals \( I(f) \) for \( f(x_{CD}) = p(x_{CD} \mid x_S) g(x_{CD}) \), and for an appropriate mixture \( \mu \) of counting and Lebesgue measures.

**Example 4.3** If we in Example 4.2 replace the posterior by the likelihood function and the function \( g(x_{CD}) \) by the prior (density and mass function), we get the marginal likelihood,

\[
p(x_S) = \sum_{x_D \in \mathcal{A}_D} \int_{\mathcal{A}_C} p(x_S \mid x_{CD}) p(x_{CD}) dx_C.
\]
This equals $I(f)$ for $f(x_{C,U,D}) = p(x_S \mid x_{C,U,D}) p(x_{C,U,D})$, and for the same mixture $\mu$ of counting and Lebesgue measures as in Example 4.2. \qed

Integration problems are similar to the sum-product problems considered in Chapter 3. Both problems deal with marginalization of basically arbitrary multidimensional functions. However, there are differences between these marginalization tasks. First, here we allow for uncountable variable domains, which makes the exact algorithmic evaluation impossible in general. Second, the marginal to be evaluated is here one-dimensional (scalar). Third, the possible product structure of the integrand is omitted in the problem formulation. Fourth, the algebraic structure is fixed to the real numbers with ordinal addition. Regarding this representational lack, possible extensions of the integration problem will be discussed in Section 4.6.

What type of algorithms are suitable for an integration problem? Since exact integration is infeasible in general, we must resort to approximations. There are three main methodologies proposed in literature (see e.g., [PTVF92, Jor99]). Numeric methods (e.g., Newton–Cotes formulas, Gaussian quadrature) control a worst case approximation error based on assumed smoothness of the integrand. It is known that numeric methods perform well in low dimensional problems but are not applicable when the number of dimensions is higher (greater than three). Analytic approximations (e.g., Laplace’s approximation, variational approximation) are based on fitting a relatively simple parametric function to the integrand. Integration of the fitted function then gives an approximation to the original integral. How well this technique performs obviously depends on the relationship of the integrand and the parametric function. Opposite to numeric and analytic integration, stochastic Monte Carlo methods are nondeterministic. The idea is to draw a large sample of arguments for the integrand, and then estimate the integral by the sample average. Probabilistic notions are needed to characterize the approximation quality. Perhaps due to its genericity and efficiency, variants of computer intensive Monte Carlo methods have become very popular for handling complex multidimensional integration. We discuss Monte Carlo methods in more detail in the rest of this chapter.

### 4.2 Monte Carlo importance sampling

A fundamental method for approximating a large-dimensional integral is the generic Monte Carlo importance sampling method. Any particular implementation of the method uses an importance sampling distribution $\pi$. We let $\pi(x)$ denote the probability (or probability density) of $x$. The method is as follows. First a random sample $\{x^{(t)} : t = 1, \ldots, T\} \subseteq A$ is drawn
according to the distribution \( \pi \). Then the integral \( I(f) \) is estimated by a Monte Carlo average as

\[
\hat{I}(f, \pi) = \frac{1}{T} \sum_{t=1}^{T} \frac{f(x^{(t)})}{\pi(x^{(t)})}.
\]

It is well known that under certain regularity conditions, this gives a good approximation to \( I(f) \) with a probability that tends to 1 as the sample size \( T \) grows.

More precisely, assume that \( \pi(x) > 0 \) whenever \( f(x) > 0 \), and that \( \{x^{(t)} : t = 1, \ldots, T\} \) is an ergodic\(^1\) sample from \( \pi \). Then, for any \( \varepsilon > 0 \), we have

\[
\lim_{T \to \infty} P\{|\hat{I}(f, \pi) - I(f)| < \varepsilon\} = 1,
\]

where \( P \) denotes the probability measure induced by random sampling from \( \pi \); this type of convergence is briefly written as \( \hat{I}(f, \pi) \to I(f) \). Moreover, the estimator is unbiased, i.e., the expected value of \( \hat{I}(f, \pi) \) equals \( I(f) \).

How fast is the convergence? In the case of independent draws from \( \pi \), the analysis is relatively simple. The variance of \( \hat{I}(f, \pi) \) with respect to \( \pi \) equals \( \text{Var}_{\pi}(f/\pi)/T \). From this we see that for fixed \( f \) and \( \pi \) the estimator converges with a linear speed (in the variance). For a fixed \( T \), it can be shown by the Cauchy-Schwarz inequality that the variance is minimized by the importance distribution that satisfies \( \pi \propto |f| \). Since we assume that \( f \) is nonnegative, the optimal importance sampling function is given by \( \pi = f/I(f) \), yielding trivially \( \hat{I}(f, \pi) = I(f) \) for all sample sizes and samples. Unfortunately, this optimal distribution cannot be used in practice. This is because the evaluation of \( \pi \) is needed for computing the integral estimate, which already involves the computation of \( I(f) \).

This simple analysis suggests the following well-known, loose characterization of the efficiency of the estimator \( \hat{I}(f, \pi) \).

**Remark 4.4** A sufficient and necessary condition for the importance sampling estimator \( \hat{I}(f, \pi) \) to perform well is that the importance sampling distribution \( \pi \) is approximately proportional to the integrand \( f \). \( \Box \)

This observation applies also to other ergodic sampling processes than independent sampling, most importantly, for sampling along a suitably specified Markov chain (see Section 4.4.4). A point that is not transparent in the above

\(^1\)We use the term "ergodic" in a loose manner to refer to a sample (or a process) to whose mean the ergodic laws of large numbers apply; for an introduction to ergodicity see, e.g., [Wal00].
remark is that \( \pi \) and \( f \) should be alike over the whole support of \( f \). Especially, \( \pi \) should have fairly heavy tails. Namely, the possibility of a large ratio \( f(x)/\pi(x) \), even when such \( x \) are rare with respect to \( \pi \), would lead to a large variance for the estimator.

We continue examples 4.2 and 4.3 to illustrate the role of the importance sampling distribution in typical Bayesian integration tasks.

**Example 4.5** Consider the posterior expectation introduced in Example 4.2. Letting the importance sampling distribution \( \pi(x_{C \cup D}) \) be equal to the posterior \( p(x_{C \cup D} \mid x_S) \) the Monte Carlo estimator for the posterior expectation becomes

\[
E_{post}(g) \approx \frac{1}{T} \sum_{t=1}^{T} g(x_{C \cup D}^{(t)}).
\]

Usually this estimator behaves well, since the functions \( g \) of interest are typically relatively flat.

Consider then the marginal likelihood introduced in Example 4.3. A natural choice for the importance sampling distribution is the prior, \( \pi(x_{C \cup D}) = p(x_{C \cup D}) \), yielding the Monte Carlo estimator

\[
p(x_S) \approx \frac{1}{T} \sum_{t=1}^{T} p(x_S \mid x_{C \cup D}^{(t)}).
\]

However, this estimator is often ill-behaving, since typically the likelihood may be very peaky while the prior is relatively flat. \( \square \)

The above description of the Monte Carlo estimator does not fix any interpretation of the mathematical probability. Yet, it naturally suggests the frequentist interpretation, and accordingly the following reasoning. Among a large number of repeated samples of size \( T \) from \( \pi \), the frequency of samples for which the estimate \( \hat{I}(f, \pi) \) will be close to \( I(f) \) is high, provided that \( T \) is sufficiently large. Therefore, after realizing a single sample of size \( T \) from \( \pi \) the confidence on that \( \hat{I}(f, \pi) \) is close to \( I(f) \) is high. This line of thought is purely frequentist as pointed out by O'Hagan [O'H87]. From the Bayesian point of view there are two important objections to this procedure [O'H87]. First, the Monte Carlo estimator depends on the arbitrary choice of the sampling distribution \( \pi \). Thus, the same sample \( \{x^{(t)}\} \) yields different estimates if drawn from different distributions. From the Bayesian point of view this dependence is unreasonable, since it is the sample that provides information about the integral, not the arbitrarily chosen sampling distribution. This dependence on irrelevant information is a reflection of the
frequentist aim at confidence on the estimate. The second objection is that the Monte Carlo estimator ignores the values $x^{(t)}$ and only uses the derived values $f(x^{(t)})/\pi(x^{(t)})$. Thus it throws away relevant “spatial” information.

The Bayesian version of Monte Carlo [O’H87, O’H91, RG04] avoids the inconsistencies of frequentist Monte Carlo. The idea is, of course, to assign the probability distribution of $I(f)$ given a sample $\{x^{(t)}\}$. This can be done by treating the value of the integrand at $x$ as an unknown quantity until it is evaluated. The prior of $f$ can be then updated to a posterior of $f$ based on observations $\{f(x^{(t)})\}$. The posterior of the integrand in turn determines the posterior distribution of the integral. Unfortunately, the current methods are computationally feasible only for some restricted classes of priors of $f$, and for relatively small sample sizes. For this reason we henceforth only consider the computationally simpler frequentist procedures.

4.3 Tempered integration

In general, it may be difficult to find a good importance sampling distribution. Here we consider a technique that decomposes the original, difficult integration task into a number of easier ones. This idea has been earlier presented in slightly different forms by Neal [Nea93] and Gelman and Meng [GM98].

A decomposition into easier integrals is obtained as follows. Let $h_1, h_2, \ldots, h_{K+1}$ be nonnegative functions such that $h_1 = f$ and $h_{K+1} = 1$. We call these functions intermediate functions for a reason to be clear soon. Further, we define $\pi_i = h_i/I(h_i)$. It is easy to verify the identity

$$I(h_1)/I(h_{K+1}) = \prod_{i=1}^K I(\pi_{i+1}h_i/h_{i+1}).$$

Here and henceforth we assume that all integrals and well-defined and finite. This decomposition directly suggests an estimator,

$$\hat{I}_h(f) = \mu(A) \prod_{i=1}^K \hat{I}(\pi_{i+1}h_i/h_{i+1}, \pi_{i+1}).$$

Here $\mu(A) = \int_A \mu(dx) = I(h_{K+1})$, the measure of $A$, is assumed to be known (or easy to compute).

The estimator $\hat{I}_h(f)$ is efficient when the functions $h_i$ are defined so that every $h_{i+1}$ is close to $h_i$. That is, $h_1, \ldots, h_{K+1}$ "smoothly" connects the extreme functions $h_1$ and $h_{K+1}$. In principle, there are many ways to
construct a sequence of such functions. One is the geometric scheme:

\[
h_i = f^{\tau_i}, \quad 1 = \tau_1 > \tau_2 > \ldots > \tau_{K+1} = 0,
\]

where the tempering scheme \( \tau = (\tau_1, \ldots, \tau_{K+1}) \) serves as a parameter to be specified. From here we borrow the term “tempering” to be used also in the general case. Thus, we may call the sequence of intermediate functions \( h = (h_1, \ldots, h_{K+1}) \) a tempering scheme, and the value \( K \) the number of tempering steps.

There are several reasonable ways to define the optimal scheme. For example, an optimal scheme for a given total sample size \( T \) and precision \( \varepsilon \) can be defined as

\[
\hat{h}(T, \varepsilon) = \arg \max_h P\{|\hat{I}_h(f) - I(f)| < \varepsilon\},
\]

where \( \hat{I}_h(f) \) is composed based on \( K \) samples of size \( T/K \). Here the number \( K \) is understood to be determined by the scheme \( h \). Alternatively, when the interest is rather in a logarithmic scale and when no single precision is of special interest, a reasonable definition might be, for example,

\[
\hat{h}(T) = \arg \min_h \mathbb{E}(|\ln \hat{I}_h(f) - \ln I(f)|^2),
\]

where the expectation is with respect to \( P \). Unfortunately, solving this type of optimization problems seems to be very difficult.

To get some idea how the tempering scheme affects the performance of the estimator we present an asymptotic result concerning the latter criterion of optimality. Since the following theorem does not appear in related studies [Nea93, GM98], we also include a proof. After the proof we briefly discuss how this expression might be used to find good tempering schemes.

**Theorem 4.6** Let \( h = (h_1, \ldots, h_{K+1}) \) be a tempering scheme. Then

\[
\lim_{T \to \infty} T \mathbb{E}(|\ln \hat{I}_h(f) - \ln I(f)|^2) = \sum_{i=1}^{K} \left( \frac{I(h_i^2/h_{i+1})I(h_{i+1})}{I(h_i)^2} - 1 \right),
\]

where the expectation is with respect to independent sampling from the distributions specified by the tempering scheme \( h \).

**Proof** Denote \( Z = \ln \hat{I}_h(f) - \ln I(f) \). Also for each \( i = 1, \ldots, K \) denote \( Y_i = \hat{I}(\pi_{i+1}h_i/h_{i+1}, \pi_{i+1}) \) and \( \gamma_i = I(\pi_{i+1}h_i/h_{i+1}) \). We have

\[
Z = \sum_{i=1}^{K} (\ln Y_i - \ln \gamma_i).
\]
For each $Y_i$ the central limit theorem (for the Monte Carlo average) gives that

$$\sqrt{T}(Y_i - \gamma_i) \longrightarrow \mathcal{N}(0, \sigma_i^2),$$

where $\sigma_i^2 = \text{Var}_{\pi_{i+1}}(h_i/h_{i+1})$ and the convergence is in distribution as the sample size $T$ grows. Here $\mathcal{N}$ denotes the normal distribution.

We then apply the following result: Let $a$ and $v$ be real numbers and \( \{X_n\} \) a sequence of random variables such that $\sqrt{n}(X_n - a) \longrightarrow \mathcal{N}(0, v)$. If a function $g$ is continuously differentiable at $a$ with $|g'(a)| < \infty$, then $\sqrt{n}(g(X_n) - g(a)) \longrightarrow \mathcal{N}(0, v|g'(a)|^2)$. This result is often called the delta method; for a more precise statement with a proof see e.g., [vdV98, Ch. 3].

Applying this result with $g(\gamma) = \ln \gamma$ yields

$$\sqrt{T}(\ln Y_i - \ln \gamma_i) \longrightarrow \mathcal{N}(0, \sigma_i^2/\gamma_i^2).$$

Since draws from different $\pi_i$ are assumed to be independent, we get

$$\sqrt{T}Z \longrightarrow \mathcal{N}(0, \sum_{i=1}^{K} \sigma_i^2/\gamma_i^2).$$

This implies that

$$T \text{E}(|Z|^2) = \text{E}(|\sqrt{T}Z|^2) \longrightarrow \text{Var}(\sqrt{T}Z) \longrightarrow \sum_{i=1}^{K} \sigma_i^2/\gamma_i^2$$

(4.1)

as $T$ grows.

Finally, we calculate $\gamma_i = I(\pi_{i+1}h_i/h_{i+1}) = I(h_i)/I(h_{i+1})$ and $\sigma_i^2 = I(\pi_{i+1}h_i^2/h_{i+1}^2) - I(\pi_{i+1}h_i/h_{i+1})^2 = I(h_i^2/h_{i+1})/I(h_{i+1}) - \gamma_i^2$. Hence,

$$\frac{\sigma_i^2}{\gamma_i^2} = \frac{I(h_i^2/h_{i+1})/I(h_{i+1})}{I(h_i)^2/I(h_{i+1})^2} - 1 = \frac{I(h_i^2/h_{i+1})I(h_{i+1})}{I(h_i)^2} - 1.$$

Substituting this into (4.1) gives the claim. \hfill \Box

The above result suggests a heuristic for finding a good tempering scheme. One may try to minimize the sum of the $K$ terms, i.e., the asymptotic variance. Unfortunately this functional involves integral terms that cannot be handled analytically, e.g., the ultimate target integral $I(h_1) = I(f)$. A further heuristic is to consider an integrand $g$ that is in some sense similar to $f$ yet easy to handle analytically. A drawback of this approach is that it requires analysis of the integrand $f$, which may be difficult to automatize.
For a simplistic example, suppose that we want to integrate \( g(x) = ax^b \) over the range \([0, 1]\) for some constants \(a\) and \(b\). Of course, the integral is easy to find in closed form, but here we consider the sampling method. Then, for tempered integration with \(K = 2\) tempering steps, one can find the optimal geometric tempering scheme by elementary calculus. The optimal intermediate function is \( h_2 = g^{\hat{\tau}} \), where the optimal exponent is given by \( \hat{\tau} = (1 + \sqrt{b+1})^{-1} \). (Derivation of this result is somewhat tedious and therefore omitted here.) This tempering scheme is close to optimal also for a more complex integrand \( f \), provided that \( f \) is similar to \( g \).

But there are further complications that make analytic finding of a good tempering scheme hard. One is that the usual assumption of independence between and within samples from distributions \( \pi_i \) may need to be dropped. Namely, drawing an independent sample from a given \( \pi_i \) is difficult in general. Moreover, coupling the sampling distributions is advantageous, as discussed in the next section. Due to these dependences of the samples, analysis of tempering schemes becomes difficult—we have to resort to intuition, heuristics, and experiments.

4.4 Metropolis-coupled Markov chain Monte Carlo

The Monte Carlo integration method assumes that an ergodic sample can be drawn from the importance sampling distribution. Naturally this is also a requirement for the tempered importance sampling method. Conceptually the simplest ergodic sample is, perhaps, obtained by independent draws from the importance sampling distribution. Also the analysis of the resulting estimator is most easily carried out under independence. In complex cases, however, independent sampling is often computationally infeasible. Fortunately, independent sampling is not the only way to obtain an ergodic sample.

The Metropolis-Hastings algorithm [Has70] draws an ergodic sample from a given target distribution \( \pi \) along a Markov chain. The algorithm proceeds as follows. First a value \( x^{(0)} \) is drawn from a prespecified initial distribution. Then iteratively for \( t = 0, 1, \ldots \), a proposal \( x' \) for a new value \( x^{(t+1)} \) is drawn from a predefined proposal distribution \( q \) conditionally on the current state \( x^{(t)} \). The proposal is then accepted with probability \( \min\{1, \alpha(x^{(t)}, x')\} \), where

\[
\alpha(x^{(t)}, x') = \frac{\pi(x')q(x^{(t)} \mid x')} {\pi(x^{(t)})q(x' \mid x^{(t)})}
\]
is the Metropolis-Hastings acceptance ratio. If $x'$ is accepted, then $x^{(t+1)}$ is set to $x'$, otherwise it is set to the previous state $x^{(t)}$. The procedure outputs a realization $x^{(0)}, x^{(1)}, \ldots$ of a Markov chain with stationary distribution $\pi$. It can be shown (see, e.g., [Tie94]) that in the limit, as $t$ grows, the algorithm gives an ergodic sample from the stationary distribution $\pi$.

In practice, only a finite number of iterations can be simulated, which makes the procedure approximative. Usually the procedure involves two phases. First, a number of preliminary iterations, so called burn-in, is run to ensure that the chain has approximately converged to the stationary distribution. Often this means that, after burn-in, the chain lies in a region where the probability mass of the target distribution is concentrated on. In the second phase, additional iterations are performed to collect a sample from the stationary (target) distribution. The number of additional iterations needed depends on how effectively the chain mixes, i.e., explores the important parts of the state space. A good proposal distribution is needed to obtain a rapidly mixing chain.

A central design issue in the Metropolis–Hastings algorithm is the proposal distribution. Ideally the proposal distribution coincides with the target distribution in which case every proposal will be accepted yielding independent sampling. However, we assumed that independent sampling from the posterior is hard. For the Metropolis–Hastings algorithm we need to generate proposals efficiently. Various generic, and rather simple, forms for proposal distributions have been suggested (see e.g., [GRS96]). Most of these are based on local changes of the current state of the chain. How well such local proposals work seems to depend crucially on specific properties of the target distribution. For multimodal high-dimensional distributions local proposals are not efficient.

To construct efficient proposal distributions for complex cases, tempering techniques can be applied [Gey91, Nea93, Nea01, GM98]. The idea is very much the same as in the tempered integration method described in the previous section: Several intermediate distributions are introduced as a path between the target distribution and an easy base distribution, say the uniform distribution. Below we describe the Metropolis-coupled Markov chain Monte Carlo (MC$^3$) due to Geyer [Gey91]. Neal [Nea01] introduces a related method on which some comments are given in the next section.

MC$^3$ [Gey91] is a special case of the Metropolis–Hastings algorithm in a product state space corresponding to several target distributions. In agreement with the notation of the previous section we let $\pi_1, \ldots, \pi_{K+1}$ be a sequence of distributions. The MC$^3$ algorithm constructs a chain whose stationary distribution is the product distribution $\pi_1 \pi_2 \cdots \pi_{K+1}$ by
simulating $K+1$ chains in parallel using the Metropolis–Hastings algorithm for each chain. In addition, it occasionally (with some fixed probability) proposes a swap of the states $x_i^{(t)}$ and $x_{i+1}^{(t)}$ of two adjacent chains with stationary distributions $\pi_i$ and $\pi_{i+1}$. The proposal is then accepted with probability $\min\{1, \beta(x_i^{(t)}, x_{i+1}^{(t)})\}$, where

$$\beta(x_i^{(t)}, x_{i+1}^{(t)}) = \frac{\pi_i(x_{i+1}^{(t)}) \pi_{i+1}(x_i^{(t)})}{\pi_i(x_i^{(t)}) \pi_{i+1}(x_{i+1}^{(t)})}.$$ 

If the swap is accepted, then $(x_i^{(t+1)}, x_{i+1}^{(t+1)})$ is set to $(x_i^{(t)}, x_{i+1}^{(t)})$, and otherwise to $(x_i^{(t)}, x_{i+1}^{(t)})$. The states of the other chains are not changed. Coupling the chains is effective, as the proposals for chain $i$ are drawn from a distribution that is close to $\pi_i$. A swap of two states may correspond to a nonlocal move on the two chains. This way the relatively rapid mixing of tempered chains can feed into the cooler chains.

When run for $T$ iterations, MC$^3$ outputs a sample $\{x_i^{(0)}, \ldots, x_i^{(T)}\}$ for each $i = 1, \ldots, K+1$. Geyer [Gey91] discards the samples for $i = 2, \ldots, K+1$, since the aim was at drawing a sample from the coolest distribution $\pi_1$. But, as we found in the previous section, the samples from the tempered distributions may also be useful.

The MC$^3$ algorithm is particularly suitable in Bayesian statistics. Namely, the algorithm solves three essential tasks in a single run. First, the algorithm can effectively draw a sample from a complex posterior distribution. Such a sample can be used to estimate posterior expectations (see Example 4.5). Second, as a side product it draws samples from tempered distributions. These samples can be used to estimate the normalizing constant of the posterior distribution, i.e., the marginal likelihood (see Example 4.5). And third, the MC$^3$ algorithm can analyze several data sets of different sizes simultaneously. To do this, one can define the intermediate distributions, used by the algorithm, as the posterior distributions for nested subsets of data. We elaborate this idea in the example below.

**Example 4.7** We follow the notation of Example 4.5. Suppose that there are $n$ observed variables, $x_S = (x_1, \ldots, x_n)$. Define $K+1$ intermediate distributions as follows. For each $i = 1, \ldots, K+1$ put $\pi_i(x_{C\cup D}) = p(x_{C\cup D} \mid x_S)$, where $S_i = \{1, \ldots, n_i\}$ and $n = n_1 > n_2 > \cdots > n_{K+1} = 0$. Here we assume that $K \leq n$. In a typical Bayesian model the distributions defined in this way form a fairly smooth path between the posterior $p(x_{C\cup D} \mid x_S)$ and the prior $p(x_{C\cup D})$.

The output of the MC$^3$ algorithm can be used to estimate posterior expectations, based on a sample from $\pi_1$, as well as the marginal likelihood.
\( p(x_S) \) based on samples from distributions \( \pi_2, \ldots, \pi_{K+1} \). Moreover, the samples can be used to analyze these quantities similarly for each smaller data set \( S_i \), since the samples from distributions \( \pi_i, \ldots, \pi_{K+1} \) are available.

\[ \square \]

### 4.5 Related methods

Interestingly, the idea of tempered integration has been proposed in various forms, leading to slightly different estimators for the target integral. What we have studied in the previous sections is just one branch in a tree. In this section we briefly explore some closely related methods: Geyer’s reverse logistic regression [Gey93], Neal’s annealed importance sampling [Nea01], bridge sampling due to Meng and Wong [MW96], and path sampling due to Gelman and Meng [GM98].

**Metropolis coupling and reverse logistic regression.** Geyer [Gey91] proposes the Metropolis coupling scheme in order to improve the mixing of the Markov chain in a setting where the objective is to draw a sample from a single stationary distribution of the coolest chain. In that context the samples from the intermediate distributions are discarded. Later [Gey93], he suggests a serialized version of the original method. A drawback of that method was that it required estimates for the normalizing constants \( I(h_i) \) in our context. As a solution, Geyer [Gey93] introduces an estimator that uses the output of a preliminary run. This estimator is based on reverse logistic regression. Though originally designed for the serialized algorithm, the estimator also applies to the output of the original parallel version.

For \( i = 1, \ldots, K + 1 \) let \( \pi_i \) be proportional to \( h_i = f^{\tau_i} g^{1-\tau_i} \), where \( \tau_1 > \tau_2 > \cdots > \tau_{K+1} = 0 \). Here it is supposed that sampling from a distribution that is proportional to function \( g \) is relatively easy. The objective of the reverse logistic regression method is to estimate the normalizing constants \( c_i = I(h_i) \) for all \( i = 1, \ldots, K + 1 \). Geyer [Gey93] suggests maximization of a quasi log-likelihood of \( c = (c_1, \ldots, c_{K+1}) \), defined by

\[
l(c) = \sum_{i=1}^{K+1} \sum_{t=1}^{T} \ln p_i(x_i^{(t)}, c),
\]

where \( \{x_i^{(t)} : t = 1, \ldots, T\} \) is a sample from \( \pi_i = h_i/c_i \) and

\[
p_i(x, c) = \frac{h_i(x)/c_i}{\sum_{j=1}^{K+1} h_j(x)/c_j}.
\]
Note that this rather heuristic likelihood function is constructed as if the samples $x_i^{(t)}$ were drawn from a mixture of the intermediate distributions, thus forgetting the group membership $i$. It can be shown that $l(c)$ is concave, which makes simple iterative optimization methods (e.g., gradient descent) applicable.

We see that the reverse logistic regression method and the tempered integration method of Section 4.3 are quite different, although they both use the output of a Metropolis-coupled Markov chain.

**Annealed importance sampling.** An objective of the annealed importance sampling method by Neal [Nea01] is to compute an estimate for the ratio $I(f)/I(g)$ of two nonnegative functions $f$ and $g$. In the case when $g$ is a probability function, this would give directly an estimate for $I(f)$.

The annealed importance sampling produces a sample of points $x^{(1)}, \ldots, x^{(T)}$, and associated importance weights, $w^{(1)}, \ldots, w^{(T)}$. An estimate for the integral is then found by

$$I(f)/I(g) \approx \frac{1}{T} \sum_{t=1}^{T} w^{(t)}.$$

Here each point $x^{(t)}$ is drawn independently by an annealed procedure that works in an augmented state space as follows. The idea is to generate each $x^{(t)}$ via a sequence of intermediate distributions $\pi_i$ defined to be proportional to $h_i = f^{\tau_i} g^{1-\tau_i}$, where $1 = \tau_1 > \tau_2 > \cdots > \tau_{K+1} = 0$, for a $K$-step annealing scheme. A sequence $x_1, \ldots, x_K$ is generated, in the reversed order, from a Markov chain so that the joint density is given by

$$q_{K+1}(x_K) T_{K+1}(x_K, x_{K-1}) \cdots T_2(x_2, x_1),$$

where each transition function $T_i$ is defined such that it leaves $\pi_i$ invariant. Finally $x^{(t)}$ is set to $x_1$ and the weight is calculated as

$$w^{(t)} = \prod_{i=1}^{K} \frac{h_i(x_i)}{h_{i+1}(x_i)}.$$

With these weights the Monte Carlo estimator is consistent.

We see that the above described annealed importance sampling method and the tempered integration method of Section 4.3 are similar, yet different. Essentially, the annealed estimator is a sum of products whereas the tempered estimator is a product of sums.
**Bridge sampling.** Meng and Wong [MW96] study the following interesting bridge identity

\[
I(f)/I(g) = I(\pi_2 \beta/g)/I(\pi_1 \beta/f),
\]

where \(f\) and \(g\) are two positive functions, \(\pi_1 = f/I(f)\) and \(\pi_2 = g/I(g)\), and \(\beta\) is an (almost) arbitrary bridge function between \(f\) and \(g\). This identity immediately suggests a consistent estimator for the ratio \(I(f)/I(g)\) as a ratio of two Monte Carlo averages based on two draws on \(\pi_1\) and \(\pi_2\) respectively.

Meng and Wong push the idea further. For \(i = 1, \ldots, K + 1\) let \(\pi_i\) be proportional to an intermediate function \(h_i\), such that \(h_1 = f\) and \(h_{K+1} = g\). Further, for \(i = 1, \ldots, K\) let \(\beta_i\) be a bridge function between \(h_i\) and \(h_{i+1}\). By using the bridge identity \(K\) times in a telescoping fashion they find,

\[
\frac{I(f)}{I(g)} = \prod_{i=1}^{K} \frac{I(h_i)}{I(h_{i+1})} = \prod_{i=1}^{K} \frac{I(\pi_{i+1} \beta_i/h_{i+1})}{I(\pi_i \beta_i/h_i)}.
\]

This suggests constructing a consistent estimator as a product of ratios of Monte Carlo averages based on draws from the \(K + 1\) distributions \(\pi_i\). Note that the functions \(\beta_i\) are computed but not sampled from.

Do the bridge sampling and the tempered integration method of Section 4.3 coincide for some choice of the bridge functions \(\beta_i\)? The answer is no: The bridge sampling method always draws from the coolest distribution \(\pi\) but the tempered integration method never.

Is the bridge sampling then more efficient than the tempered integration method? The answer depends on the number of tempering steps \(K\). In the case \(K = 1\) the optimal bridge sampling can be shown (e.g., [GM98]) to have asymptotic (squared) error \(1/I(\beta) - 1)^2/T\), where \(\beta\) is the optimal intermediate distribution, the harmonic average, \(\beta = 2\pi_1 \pi_2/(\pi_1 + \pi_2)\). This assumes \(T\) independent draws from both distributions \(\pi_1\) and \(\pi_2\). To estimate \(I(f)\) for \(f(x) = ax^b\) on \(A = [0, 1]\) we put \(\pi_1(x) = (b + 1)x^b\) and \(\pi_2(x) = 1\). Hence, we have \(I(\beta) = 2 - 2 \int_0^1 [(b + 1)x^b + 1]^{-1} dx\). By numeric integration we get approximations of the errors (times \(T\)) for \(b = 1, 2, 10, 50\) as 0.22, 0.53, 2.77, 11.3, respectively. As a comparison, consider the tempered integration method with the optimal geometric tempering scheme given in Section 4.3. The expression given in Theorem 4.6 yields the optimal errors 0.19, 0.44, 1.9, 5.8. This suggests that the tempered method is in fact more efficient than the bridge sampling. In other words, an intermediate distribution seems to be more beneficial as a sampling distribution than as a pure computational bridge. This observation does not appear in
4.5 Related methods

[MW96, GM98]). However, this view changes as we take more tempering steps (calculations omitted). Note also that typically a sample from \( \pi_1 \) is also obtained, but ignored by the tempered integration method.

**Path sampling.** Gelman and Meng [GM98] introduce a general framework of path sampling for simulation of normalizing constants. For two functions \( f \) and \( g \) with a continuous path of intermediate functions \( h_\tau \) with \( \tau \in [0, 1] \), one finds by an interchange of differentiation and integration the beautiful path expression:

\[
\ln \frac{I(f)}{I(g)} = \ln \frac{I(h_1)}{I(h_0)} = \int_0^1 \int_A \pi_\tau(x) \frac{d}{d\tau} \ln[h_\tau(x)] \mu(dx) d\tau,
\]

where \( \pi_\tau = h_\tau / I(h_\tau) \) are the normalized functions. It is assumed here that \( h_1 = f \) and \( h_0 = g \). A suitable path may be, for instance, the geometric path \( h_\tau = f^\tau g^{1-\tau} \). As noted by Gelman and Meng, specialized forms of the path expression have been earlier used in statistical physics under the name of thermodynamic integration [Nea93, Section 6.2], and later in statistics under the name of Ogata’s method (see [GM98] and references therein).

Assuming the geometric path and that \( g \) is the uniform distribution on \( A \), the above expression directly suggests a relatively simple importance sampling Monte Carlo estimator,

\[
\ln I(f) \approx \frac{1}{T} \sum_{t=1}^T \frac{\ln f(x^{(t)})}{p(\tau^{(t)})},
\]

where the sample \( \{ (\tau^{(t)}, x^{(t)}): t = 1, \ldots, T \} \) is drawn from a joint distribution \( p(\tau, x) = p(\tau) \pi_\tau(x) \). Here the distribution \( p(\tau) \) is arbitrary and should be chosen so that the resulting estimator has a small variance. Actually, Gelman and Meng go a step further and show that optimizing the path in the distributional space, not restricting to geometric paths, is possible.

We see that the method is similar to the tempered integration method of Section 4.3. Essentially, the path estimator is a product of products whereas the tempered estimator is a product of sums. It is not clear that the path estimator is more efficient than the tempered estimator. Also, a drawback of the path integration method is that it may be difficult to implement. Straightforward MCMC does not apply to generating samples from \( p(\tau, x) \) since the normalizing constants \( I(h_\tau) \) are not known for distributions \( \pi_\tau \). Gelman and Meng [GM98] propose two methods that are essentially the methods of Geyer [Gey91, Gey93]: parallel Metropolis coupling and serial simulated tempering coupling. As already mentioned, the latter method
requires estimates for the normalizing constants and thus leads to an iterative procedure. In conjunction with the former method, MC$^3$, the path integration method is almost identical to tempered integration. Note, however, that MC$^3$ requires a fixed finite tempering scheme. In that sense the path integration method would not integrate over a continuous path, which was the original idea of the method.

4.6 Concluding remarks

The generality of the presented tempering method for complex integration does not come for free. Many parameters of the method are left unspecified, including the tempering scheme, chain-specific proposal distributions, the length of the burn-in period, and the total number of iterations in the MC$^3$ algorithm. These case-dependent parameters are probably best found by experimentally and manually, as their automatic optimization without human interaction is notoriously difficult (for a relevant discussion see e.g., [KCGN98]). A different issue is that the methods do not exploit the possible structure of the integrand explicitly.

A natural way to extend the integration problem is to represent the integrand as a product of local functions and the integral as a marginal function. This elaboration, of course, makes the problem more complicated. One ought to exploit the problem structure, perhaps in a variable elimination fashion. Also, one has to use some effective approximate representations for the intermediate and ultimate marginal functions. Specialized algorithms are developed under the frameworks of "sequential Monte Carlo" (see e.g., [DdFG01]) and "Hybrid Bayesian networks" (see e.g., [Ler02]). Koller et al. [KLA99] provide a unified approach in the context of Bayesian network inference. These methodologies are developed with emphasis on low dimensional (intermediate) integrals. Consequently, they employ simple analytic approximations and basic importance sampling. When moderate or large dimensional integrals are needed, tempering techniques might offer significant improvements.
Chapter 5

Population risks and genetic models

In this chapter we introduce the problem of analyzing epidemiological data under a genetic model. We start by considering population risks of binary-valued traits in Section 5.1. Our focus is on genetic traits, and therefore the concepts of pedigree and relationship types are introduced. In Section 5.2 we discuss the usual way of obtaining data: sampling from the population. A general probabilistic model for handling pedigree data is reviewed in Section 5.3. As a special case we obtain a general model for population risk data. A more detailed model is introduced in Section 5.4. Finally, in Section 5.5 we introduce two inter-related problems: The direct problem is to compute recurrence risks given a fully specified model. The inverse problem is to infer model parameters given risk data. These problems are studied further in chapters 6 and 7, respectively.

5.1 Population risks of binary traits

Many diseases are important public health concerns. Thus they play a major role in scientific research that pursues better understanding of the underlying biological mechanisms, drug design, as well as medical risk analysis and consulting. Asthma, diabetes, schizophrenia, cardiovascular diseases, and cancers of different types are examples of such disorders.

In this thesis, it is appropriate to view diseases as special cases of traits. By a trait we mean a property that, at least in principle, can be measured or diagnosed in an individual. The notion of trait covers also properties that need not be disorders, e.g., blood group or handedness.

The measured value of a trait for an individual is called the phenotype or trait status of the individual. In practice it is, however, often the case that the trait as such cannot be measured, but only some symptoms or features
of it. We suppose that the trait we consider is an observable phenotype.

Usually studies of traits concern populations, i.e., sets of individuals. Population is a very central but somewhat vague concept. Therefore we clarify the meaning we use henceforth. First, a population may refer to a set of individuals living in some restricted geographical region. A population may consist of individuals from past generations, the current generation, and potential future generations. Second, we may sometimes consider a collection of imaginary populations. Such a collection can be very large (uncountable) as it serves as a basis of a theoretical probability model. Such a meta-population should not be confused with the actual, realized populations. Last, we consider mainly human populations in this study, though the methodology directly applies to many other species as well.

We focus on binary traits. A trait is called binary if, in the population under consideration, it occurs in exactly two distinct forms. It is clear that any binary trait can be reformulated so that an individual either has the trait or does not have the trait; alternatively we may say that an individual manifests the trait. That said, we treat the trait of interest as an indicator function that takes values in the set \( \{0, 1\} \).

Given a population and a binary trait, several characterizing statistics can be defined. The population prevalence, denoted by \( K_P \), is the relative frequency of the trait, i.e., the proportion of individuals who have the trait. Thus, the prevalence can be viewed as an overall or general risk for the trait. When occurrence of the trait is correlated with one or more other attributes, e.g., age, sex, or some environmental conditions, we speak about conditional risk statistics. Of such conditional risk statistics, we are, in this thesis, particularly interested in the so called recurrence risks. Loosely stated, recurrence risk is a risk for the trait given that a specific relative has the trait. For instance, sibling risk is the risk for a sibling of a person who has the trait. Recurrence risks are interesting because they provide information about possible genetic explanations for the trait. To formulate recurrence risks more precisely we next introduce the concept of pedigree.

A pedigree is a graph that relates a set of individuals. We omit a precise definition and give only a semi-formal description. We say that a tuple \((V, fa, mo, id)\) is a pedigree if \(V\) is a set (of individuals), and \(fa\), \(mo\), and \(id\) are binary relations on \(V\) with the following interpretations. First, \((u, v) \in fa (mo)\) if and only if \(v\) is the father (mother) of \(u\). Second, \((u, v) \in id\) if and only if \(v\) is an identical sibling of \(u\). These relations are required to satisfy the natural topological constraints. Thus, for example, \(fa\) and \(mo\) are asymmetric relations while \(id\) is a symmetric relation. Naturally, we say that \(u\) is an ancestor of \(v\) if there exists a path \(u_1, \ldots, u_k\) for some
5.1 Population risks of binary traits

$k \leq 1$ such that $u_1 = u$, $u_k = v$, and for each $i = 1, \ldots, k - 1$ either $u_{i+1} = fa(u_i)$ or $u_{i+1} = mo(u_i)$. Specially, every individual is her (or his) own ancestor. For a subset $U \subseteq V$ of individuals the pedigree induces a subpedigree as the corresponding subgraph. For an illustration of a pedigree see Figure 5.1.

A pairwise relationship type is a relation for pairs of individuals. In a pedigree, the relationship of two individuals is determined by the subgraph topology induced by the union of the ancestors of the two individuals. But this gives a too detailed characterization of relationship types. For example, under this definition, the relationship of a sibling pair is usually different than the relationship of another sibling pair. This is impractical. Therefore, we relax the definition by including only some most recent ancestors. We occasionally refer to such a pedigree as the canonical pedigrees of the relationship in question. This way we can find natural definitions for common relationship types. We focus mainly on the following types: parent-offspring, siblings, cousins, identical twins, denoted by O, S, C, I, respectively. We make a distinction between the two orders of a pair. If the order is not specified, we follow order by age. To emphasize this we may use singular terms, such as offspring, younger sibling. Some relationship types are illustrated in Example 5.1 below.

Given a population and a trait, recurrence risks for different relationship types are well defined. For a fixed relationship type $R$, the recurrence risk for $R$, denoted by $K_R$, is the ratio of the number of pairs (of type $R$) where
both have the trait, and the number of pairs (of type \( R \)) where at least the second member of the pair has the trait. Example 5.1 clarifies this definition.

**Example 5.1** Consider the population represented in Figure 5.1. We observe that the population prevalence is given by \( K_P = 6/11 \). To calculate offspring risk we first notice that there are 14 parent-offspring pairs in the pedigree. In 5 cases both have the trait, while in 7 cases the parent has the trait. Thus the offspring risk is \( K_O = 5/7 \). The number of sibling pairs is 5. In 2 cases both have the trait, while in 4 cases the elder sibling has the trait. Thus the sibling risk is \( K_S = 2/4 = 1/2 \). Similarly we obtain the cousin risk \( K_C = 2/4 = 1/2 \). The risk for identical twin is not defined in this population. \( \square \)

### 5.2 Sample-based risk statistics

We are usually interested in populations that cannot be examined exhaustively. The reason may be the large size of the population, or that the population involves individuals from past (e.g., issues of historical reconstruction) or future (e.g., prediction issues) generations. Consequently, the values of population risks cannot be observed.

Sampling is a usual method to obtain information concerning a population. A sufficiently large sample of individuals, drawn from the population, serves as a representative subpopulation so that the population statistics of interest can be estimated by the corresponding subpopulation statistics. Such an inductive generalization is valid in a sense specified by the chosen scientific paradigm. Theoretical foundations of sampling and generalization are well-studied in statistics and probability theory.

In our case, we would like to estimate the population prevalence and certain recurrence risks for the trait of interest. The assumption is that one can draw a random sample from the population. In the case of population prevalence, this means that all subsets of individuals of a given size have an equal probability to be realized as the actual sample. In the case of recurrence risks, say sibling risk, one is supposed to draw a random sample of sib-pairs whose elder sibling has the trait. This rather frequentist phrasing has, of course, a natural degree-of-belief interpretation: one does not assume that the sample is drawn by a “truly random method”; instead, one judges (without better knowledge) all subsets to be equally probable a priori.

We assume a simple representation of risk data. Suppose we have data on a collection \( \mathcal{R} \) of relationship types. For each \( R \in \mathcal{R} \) suppose that \( N_R \) pairs of type \( R \) are drawn from the population under the condition
that the first member in the pair has the trait. Let $M_R$ be the number of pairs where both have the trait. We denote the data by $(N_R, M_R)$, where $N_R = \{ N_R : R \in R \}$ and $M_R = \{ M_R : R \in R \}$. Observations concerning the population prevalence can also be included to the data as numbers $N_P$ and $M_P$, in which case P is treated as a special relationship type.

5.3 General genetic model

We are interested in genetic explanations for the population risk observations. Therefore, we consider genetic hypotheses that describe possible underlying genetic mechanisms at an appropriate level. A well-formulated genetic hypothesis should state precise values of the quantities of our principal interests. However, it is advisable to also introduce some ancillary parameters when specifying how hypotheses and data are related.

The general model for segregation analysis by Elston and Stewart [ES71] is a good starting point when considering genetic hypotheses. This model aims at describing a high-level structure of models that can be used for the analysis of familial aggregation of phenotypes in arbitrary pedigrees. It is comprised of four components. First, it describes the essential parts of genetic material, that is, the genotypes. The second component specifies the population frequencies of different forms of the genotypes. The third component tells how the genotypes are transmitted from parents to children. Finally, the fourth component describes how the genotypes and phenotypes are related. Once all these component are fully specified, we have a genetic hypothesis that can be compared against the observed pedigree data.

We now consider the general model in more detail. Let $(V, E)$ be a pedigree, where $V$ is a set of individuals and $E$ describes the relations between the individuals. Denote by $x^v$ the genotype and by $y^v$ the phenotype of an individual $v \in V$. Suppose that the domain of the genotypes is defined by the first component of the model. The other three components are used as building blocks of the joint distribution $p(y^V, x^V)$ of the phenotypes and genotypes of all individuals in the pedigree. Because this composition of the joint distribution involves some central assumptions, a derivation is given below.

We start by using the chain rule:

$$p(y^V, x^V) = p(y^V \mid x^V) p(x^V). \quad (5.1)$$

Then we make the assumption that given the genotype, the phenotype of an
individual is independent of the other individuals’ genotypes. This yields
\[ p(y^V \mid x^V) = \prod_{v \in V} p(y^v \mid x^v). \]

To decompose the second part in (5.1) we first use the chain rule and write
\[ p(x^V) = p(x^F) p(x^{V-F} \mid x^F), \tag{5.2} \]
where \( F \subseteq V \) is the set of founder individuals in the pedigree. Assuming that the genotypes of the founders are independent we get
\[ p(x^F) = \prod_{v \in F} p(x^v). \]

Finally we decompose the second term in (5.2) under the assumption that given the genotypes of the parents, the genotype of an individual is independent of the others’ genotypes. This yields
\[ p(x^{V-F} \mid x^F) = \prod_{v \in V-F} p(x^v \mid x^{fa(v)}, x^{mo(v)}), \]
where for each \( v \) a father \( fa(v) \) and a mother \( mo(v) \) are determined by the pedigree. Here, identical twins make an exception. Since genotypes of identical twins are identical, the genotypes are not independent. Therefore, only one twin per pair should be included in the expression. Ignoring this exception we summarize the above development:
\[ p(y^V, x^V) = \left[ \prod_{v \in F} p(x^v) \right] \left[ \prod_{v \in V-F} p(x^v \mid x^{fa(v)}, x^{mo(v)}) \right] \left[ \prod_{v \in V} p(y^v \mid x^v) \right]. \tag{5.3} \]

Thus, the general model consists of three components: frequency model \( p(x^v) \), transmission model \( p(x^v \mid x^{fa(v)}, x^{mo(v)}) \) and penetrance model \( p(y^v \mid x^v) \). Any specific model specifies these components, usually based on some appropriate parametrization so that a parameter configuration corresponds to a genetic hypothesis.

Elston and Stewart [ES71] choose the frequentist interpretation. Accordingly, the phenotypes and genotypes are treated as random variables whose distribution is defined by the unknown population parameters (population proportions) and the sampling method (independent random sampling). The population parameters, in the most general case, correspond directly to the above mentioned three distributional components, determined by the corresponding population proportions.
We choose a Bayesian framework, which differs only in the interpretation of “random sampling”. For instance, \( p(x^v) \) is not interpreted as the frequency of genotype \( x^v \) but as the degree of belief that \( v \) carries \( x^v \). In notation, we need to be careful. Namely, since the parameters that determine the distributions are usually unknown, they need to be included in the notation. Therefore, letting \( \theta \) denote the parameters, we write \( p(x^v \mid \theta) \) rather than \( p(x^v) \), and similarly for the other components.\(^1\) This way, the general model of Elston and Stewart can be understood as a likelihood model, which is part of a full Bayesian model.

We finally define the probability of risk data based on the above general decomposition for pedigrees. For a relationship type \( R \), we interpret the observations \( N_R \) and \( M_R \) as a collection of \( N_R \) canonical pedigrees of \( R \), where only the phenotype of the two relatives of type \( R \) are observed. Furthermore, in \( M_R \) cases both relatives have the trait, and in \( N_R - M_R \) cases only the elder relative has the trait. This kind of representation of a collection of individuals as a number of separate pedigrees assumes that the individuals in different pedigrees are distant relatives so that the relationship need not be represented. For such pedigree data the general model yields a binomial model,

\[
p(M_R \mid N_R, \theta) = \prod_{R \in \mathcal{R}} \binom{N_R}{M_R} K_R(\theta)^{M_R} (1 - K_R(\theta))^{N_R - M_R}. \quad (5.4)
\]

Here the (theoretical) recurrence risk \( K_R(\theta) \) is defined by

\[
K_R(\theta) = p(y^v = 1 \mid y^u = 1, \theta)
\]

in the canonical pedigree of \( R \), where \((u, v)\) is of the relationship type \( R \).

Sometimes the above model is not appropriate. Namely, in practice many relatives per proband\(^2\) are diagnosed systematically. Also, the data is often based on a single large pedigree where all individuals are relatives. Corrections to the formula (5.4) are needed to take into account such ascertainment schemes (see e.g., [Guo98]). In this work we only consider the above idealized model (5.4).

\(^1\)In fact, both of these notations are valid but they refer to different quantities. The latter one denotes the unconditional probability that \( v \) carries \( x^v \)—the unknown parameters are integrated away with respect to their prior distribution.

\(^2\)In epidemiology “proband” means an individual by whom also a set of relatives are included into the study. Usually the proband herself is known to manifest the trait of interest.
5.4 Epistatic Mendelian model

In this section we describe a genetic model that is considered in the subsequent parts of this thesis. We continue the approach taken in the previous section by introducing a model that describes the structure of the genotypes and parametrizes their population frequencies, transmission model, and penetrance model. Here we also fix some notation.

5.4.1 Ingredients of genotypes

A central parameter is the number of genes underlying the trait. For a gene we use a less vague term *locus*. We let $\ell$ denote the number of loci involved. We mainly consider the cases where $\ell = 1, 2, 3$. This parameter is particularly important because it essentially specifies the complexity of the model. This point becomes clear later in this and subsequent sections. We do not care about the physical locations of the loci.

With each locus $i = 1, \ldots, \ell$ we associate two alleles, i.e., variants of the gene, denoted by $A_i$ and $a_i$. Sometimes these alleles are understood as “predisposing” and “neutral” variants. Since a gene may actually have more than two variants, these two variants should be understood as super alleles (classes of alleles). We let $\mathcal{A}_i$ denote the set \{A_i, a_i\} of alleles at locus $i$. For convenience we sometimes denote the alleles $A_1, A_2, A_3$, and $a_1, a_2, a_3$, respectively by $A, B, C$, and $a, b, c$. But we may also consider a more general case with $m_i$ (super)alleles at locus $i$.

Genotypes are combinations of alleles. A *genotype at locus* $i$ is an unordered pair of alleles. Formally we define genotype at locus $i$ as a subset of $\mathcal{A}_i$ with one or two elements. Genotypes of size one are called *homozygous* whereas genotypes of size two are called *heterozygous*. We denote the set of these subsets by $\mathcal{Z}_i$. An *ordered genotype at locus* $i$ is an ordered pair of alleles, formally, an element of the Cartesian product $\mathcal{A}_i \times \mathcal{A}_i$, denoted by $\mathcal{X}_i$. The ordered version can express the parental origin of the alleles. Our convention is that the first member of a pair refers to the paternal allele, and the second to the maternal allele. A genotype over the $\ell$ loci is called simply a *genotype*. Formally, a genotype is an element of $\mathcal{Z}_1 \times \cdots \times \mathcal{Z}_\ell$. Similarly, an *ordered genotype* over the $\ell$ loci is an element of $\mathcal{X}_1 \times \cdots \times \mathcal{X}_\ell$.

The distinction between unordered and ordered genotypes is useful. On one hand, it is often natural to assume that the ordered version of an unordered genotype provides no additional information concerning the phenotype. On the other hand, the ordered version serves as a more natural building block in the transmission model. We come back to these issues in the subsequent sections.
Finally, we fix some useful notation. In order to notate complex partial genotypes conveniently, we use indexing by subsets. Henceforth we let $L$ denote the set of all $2\ell$ allelic positions. Formally, we define $L = \{1, 2, \ldots, 2\ell\}$. The interpretation is that the first $\ell$ positions refer to the paternal allele locations and the latter $\ell$ to the maternal allele locations. Thus, if $S = \{i_1, \ldots, i_k\}$ is a subset of $L$ with $i_1 < \cdots < i_k$, then $A_S$ denotes the set $A_{i_1'} \times \cdots \times A_{i_k'}$, where $i_j' = i_j \pmod{\ell} + 1$. Similarly, if $x$ is an element of $A_L$, then its projection on locations $S$ is an element $x_S \in A_S$, called a partial genotype. In case $S = L$ we may drop out the index. For a singleton $S = \{i\}$ we may drop out the brackets from the notation.

Although the sets $A_1' \times \cdots \times A_\ell'$ and $A$, as defined above, are different, they both give a representation of ordered genotypes. Without further conventions this might be confusing. For example, if $x$ is an ordered genotype, it is not clear whether $x_i$ should be understood as an ordered genotype at locus $i$ or as an allele at allelic position $i$. To avoid confusion, we follow the latter interpretation, if not explicitly stated otherwise.

When referring to the genotype of an individual we use the following conventions. If $u$ denotes an individual, we usually denote by $z^u$ her unordered genotype and by $x^u$ her ordered genotype. These are understood as random variables that take their values from the sets $Z$ and $A$ respectively. Yet, we may write, for instance, “$\sum_{z^u \in Z}$” as a shorthand for the summation over all events “$z^u = z$”, where $z \in Z$. Analogous conventions hold for the phenotypes ($y^u$).

### 5.4.2 Genotype frequencies

We next consider the frequencies of genotypes which are essential when assigning probabilities for genotypes. The full parametrization of the frequency distribution would require $|A| - 1$ parameters, one for each genotype. However, it is common practice to postulate independence assumptions that significantly reduce the required number of parameters.

We compose the genotype frequencies from marginal frequencies. Let $q : A \to [0, 1]$ be a frequency distribution. For each subset $S$ of $L$ we define the $S$-marginal frequency distribution $q_S$ by

$$q_S(x_S) = \sum_{x_{L-S} \in A_{L-S}} q(x).$$

We proceed with two assumptions. First, we assume that for each $i = 1, \ldots, \ell$, we have $q_i = q_{i+i}$, that is, the frequencies are the same for paternal and maternal allelic positions. Second, we assume that for all ordered
genotypes $x \in \mathcal{A}$, we have
\[
q(x) = \prod_{i \in \mathcal{L}} q_i(x_i).
\]
This is a strong version of Hardy-Weinberg equilibrium (HWE) and linkage equilibrium (LE).

Under these two assumptions it is sufficient to specify distributions $q_i$ for each locus $i$, taking $\sum_{i=1}^{\ell} (m_i - 1)$ parameters. Namely, the frequency of any partial genotype is determined by the allele frequencies:
\[
q_S(x_S) = \prod_{i \in S} q_{i'}(x_{i'}), \quad \text{where } i' = i \mod \ell + 1.
\]
In the biallelic case we may abbreviate $q_i(A_i)$ by $q_i$.

We assign probabilities based on frequencies. The probability that an individual $v$ has an ordered genotype $x^v$ given the allele frequencies $q$, is written as $p(x^v \mid q)$. If $v$ is a founder, then we assign $p(x^v \mid q) = q(x^v)$.

### 5.4.3 Transmission model

The general model of Elston and Stewart allows for arbitrary conditional transmission probabilities. Thus, in the most general setting, almost $|\mathcal{A}|^3$ parameters need to be introduced. It is common practice, however, to use much simpler models.

We assume a simple Mendelian transmission model. This model is fully specified in the sense that no "free" parameters need to be introduced. The transmission probabilities are assigned by
\[
p(x^v \mid x^{fa(v)}, x^{mo(v)}) = \prod_{i=1}^{\ell} p(x^v_i \mid x^{fa(v)}_{i,i+i+\ell}) \prod_{i=\ell+1}^{2\ell} p(x^v_i \mid x^{mo(v)}_{\{i-\ell, i\}}).
\]
This decomposition expresses the assumption of unlinked transmission, meaning that events at different loci are independent. It also states that paternal and maternal transmissions are independent. We further assume uniformity, meaning that paternal alleles are transmitted with equal probabilities:
\[
p(x_i^v \mid x^{fa(v)}_{i,i+i+\ell}) = \begin{cases} 
1/2 & \text{if } x_i^v = x_i^{fa(v)} \text{ or } x_i^v = x_i^{fa(v)}_{i,i+i+\ell}, \\
0 & \text{otherwise};
\end{cases}
\]
and similarly for maternal alleles.

The first assumption above can be viewed as a formulation of the second law of Mendel, while the latter two assumptions state Mendel's first law.
5.4 Epistatic Mendelian model

Under these assumptions the transmission frequencies, and hence, the transmission probabilities, are fully specified. Thus, no free parameters need to be introduced.

It is worth noting that the transmission model can be seen as a marginal of a more detailed model:

**Remark 5.2** To each allelic position $i$ of each individual $u$ (in the pedigree of interest) we associate an indicator variable $\xi^u_i$ which is set to 1 if the allele $x^u_i$ is inherited from a grandfather, and to 0 if it is inherited from a grandmother. Augmenting the joint probability model of genotypes and phenotypes by these *path indicators* is straightforward. □

### 5.4.4 Relation of genotypes and phenotypes

The model of Elston and Stewart leaves us the freedom to specify a probabilistic relation between genotypes and phenotypes. For a binary trait, the most general model takes one parameter for each genotype. Also here we could postulate several independence assumptions, similarly as we did in the case of genotype frequencies and the transmission model. However, for complex traits usually no background knowledge justifies such simplifications. Therefore, we have motivation to allow for complex relationships between genotypes and phenotypes.

In our model we let the penetrance be an arbitrary function of ordered genotypes. That is, we use $|\mathcal{A}|$ parameters to specify a penetrance function $f : \mathcal{A} \rightarrow [0, 1]$. We interpret $f(x)$ as the relative frequency of the trait among the individuals (in a fixed reference population) who carry $x$. Accordingly, the probability that individual $v$ has phenotype $y^v$, given that she carries ordered genotype $x^v$, is assigned simply by

$$p(y^v \mid x^v, f) = \begin{cases} f(x^v) & \text{if } y^v = 1, \\ 1 - f(x^v) & \text{otherwise.} \end{cases}$$

It is often natural to assume that the unordered genotype gives the same information about the phenotype as its ordered version. We call a penetrance function $f$ *symmetric* if $f(x) = f(x')$ whenever $x$ and $x'$ are two ordered versions of the same unordered genotype $z$. For a symmetric penetrance function $f$ we may use unordered genotypes as arguments.

The general parametrization of penetrance allows for arbitrary patterns of *epistasis*, i.e., gene-gene interactions. It covers many important special cases, including additive and multiplicative models, where penetrances are obtained from “marginal penetrances” as a sum or as a product, respectively [Ris90]. In practical studies and method development, the general model
is seldom used for more than one locus; for exceptions see [CE71, FG03].
The reason for this is perhaps the rapid growth of the number of penetrance
parameters with the number of loci. However, more theoretical studies often
concern the general epistatic model (e.g., [Kem57, Bul80]).

5.4.5 Summary

We introduced an epistatic Mendelian model. The starting point was the
course decomposition of the joint distribution of genotypes and phenotypes
due to Elston and Stewart [ES71]. We proceeded toward a finer decom-
position by introducing a parametrization based on a number of simplifying
independence assumptions. This parametrization consists of three com-
ponents: the number of loci, allele frequencies, and penetrances, denoted by \( \ell \),
\( q \), and \( f \), respectively. We considered only the conditional distributions of
genotypes and phenotypes, given a genetic hypothesis. This is the likelihood
part of a Bayesian model.

It is convenient to partition the likelihood model into several models, one
per a number of loci. Thus, we may speak about the one-locus model, the
two-locus model, and so forth. These models are obviously distributionally
nested in the sense that a \( k \)-locus model is a subset of the \((k+1)\)-locus model.
However, they are not parametrically nested in the sense that the vector of
\( k \)-locus model parameters cannot be simply extended to \((k+1)\)-locus model
parameters.

Next we complete the model by introducing a prior distribution on the
parameters. When modeling some specific trait the modeler ought to care-
fully elicitate her prior beliefs concerning the model parameters. Here we
consider some general uniform-like priors only.

We compose the prior distribution in a component-wise manner. Let
\( \theta \) denote an \( \ell \)-locus epistatic Mendelian hypothesis \((q, f)\). Structurally we
assign \( p(\theta) = p(\ell) p(q \mid \ell) p(f \mid \ell) \). This says that the allele frequencies
and penetrances are independent given the number of loci. For the number
of loci, \( \ell \), we assign the uniform prior on \( \{1, 2, 3\} \). To consider priors for
frequencies and penetrances we next introduce two notions of monotonicity.

Consider a case where each loci \( i \) has two variants: a predisposing one,
\( A_i \), and a neutral one, \( a_i \). We call an allele frequency distribution \( q \) mono-
tonic if \( q_i(A_i) \leq q_i(a_i) \) for each locus \( i \). This assumption makes sense if
one believes that predisposing alleles correspond to mutations that are rare
in the population. A penetrance function \( f \) is called monotonic if the fol-
lowing holds for every genotype \( x \): If genotype \( x' \) is obtained from \( x \) by
changing a normal allele to the predisposing one at the same locus, then
\( f(x') \geq f(x) \). For example, under the one-locus model, this assumption
states that $f(AA) \geq f(Aa) \geq f(aa)$. The assumption of monotonic penetrance is not very restrictive as it is natural to think that an increase in the number of predisposing alleles cannot reduce the penetrance. It is straightforward to generalize these definitions of monotonicity to models with more than two alleles per locus.

Later in this work we consider both monotonic and nonmonotonic penetrances, but only nonmonotonic allele frequencies. We generally assign a uniform prior on the parameter space, possibly restricted by the monotonicity constraint. We come back to this issue in Chapter 7.

## 5.5 Direct and inverse inference problem

In the previous sections we introduced binary traits, recurrence risks, and genetic models. Related to these notions, we next separate and outline two computational problems.

The direct problem is to infer the (theoretical) recurrence risk given a relationship type and an epistatic Mendelian hypothesis. This problem is well defined in the sense that the recurrence risk is a function of model parameters. Thus, the question is algorithmic: how fast can we compute recurrence risks. We study this problem in more detail in Chapter 6.

The inverse problem is to infer the model parameters given a risk data set. This problem is ill defined in the sense that there are usually more than one hypothesis that are consistent with the data. Indeed, we follow the Bayesian paradigm for statistical inference and end up with a posterior distribution on the model parameters.

Unfortunately, the inverse problem is complicated in many ways. It is known that the parameters of an $\ell$-locus epistatic Mendelian model, for all $\ell \geq 1$, are unidentifiable [Jam71]. No matter how many relationship types are included in the data, there will be several hypotheses that give the same probability for the data. This means that we cannot expect to find a reasonable point estimate for the parameters unless the prior is significantly nonuniform. For this reason, the frequentist methods [CE71], based on maximum likelihood and asymptotic normality, can hardly be justified.

We use Bayesian methods which do not rely on point estimates. To perform statistical inference and, ultimately, to facilitate decision making, we compute the marginal posterior distribution of the relevant quantities. This way we avoid some difficulties associated with unidentifiability.

In this work, our special emphasis is on inferring the number of loci. Given a data set the question is, what is the correct number of underlying loci. A related meta-level question is, how the amount of the information
provided by the data depends on different features of the data. We approach this question by estimating the Bayesian power of inferring the number of loci. Experiments are reported in Chapter 7.
Chapter 6

Computation of population risks

This chapter is devoted to the problem of computing population risks under a given genetic model. We formulate this computational task in Section 6.1. That section also discusses some existing approaches and algorithms, and motives the pursuit of advanced algorithms. Section 6.2 presents a novel algorithm for computing recurrence risks. The main idea in the method is to represent the essential computational problem as a Yates transform. In Section 6.3 we develop an alternative algorithm for computing recurrence risks. The method is based on a different sum-product representation which can be viewed as a Möbius transform. After algorithmic considerations we, in Section 6.4, jump into a separate, rather mathematical question concerning inequalities on recurrence risks. Based on formulas derived in sections 6.1–6.3, we prove an inequality on recurrence risks. We get an interesting corollary: under the Mendelian epistatic model the offspring risk is always at least as high as the sibling risk; this result has been published in [KM01]. Finally, in Section 6.5, we conclude with a number of remarks regarding the presented algorithms, their extensions, and some open problems.

6.1 The problem

In this section we formulate the problem of computing recurrence risks. The initial problem formulation of Section 5.5 was based on a somewhat ambiguous and inconvenient definition of relationship type. That problem formulation as such is not sufficient for our purposes here. Our aim is to find a reformulation that clearly states the input-output relation of an algorithmic problem. To achieve this we identify relationship types with appropriately defined coefficients of relatedness. We show that such an identification can be justified under the simple Mendelian transmission model.
6.1.1 Relationship types: coefficients of relatedness

The relationship type of two individuals can be defined by the topology of their ancestral pedigree as follows. (Note that here and henceforth we use term “topology” in an informal sense to refer to a pedigree structure.)

**Definition 6.1** Let $u$ and $v$ be two individuals in a pedigree $P$ and let $u'$ and $v'$ be two individuals in a pedigree $P'$. Let the corresponding restrictions to the ancestral pedigrees be $Q$ and $Q'$. We say that for the pairs $(u,v)$ and $(u',v')$ the relationships are equivalent if the pedigrees $Q$ and $Q'$ are isomorphic.\(^1\) The corresponding relationship type is the equivalence class of this equivalence relation.

For our purposes, however, it is convenient to use a simpler description of relationship types. An alternative characterization for relationship types can be given by various coefficients of relatedness. These coefficients measure the amount of genetic material shared by the relatives.

The definitions of various coefficients of relatedness are based on the concept of identity-by-descent which concerns the hereditary origin of alleles. Strictly speaking, this notion relates two or more allelic positions on one or more individuals. To define this relationship more precisely, we consider a fixed pedigree. Moreover, we consider an augmented version of the pedigree where the transmission paths of alleles from parents to children are explicit. Given such a descent graph we say that alleles at two allelic positions are identical by descent (i.b.d.) if their transmission paths start from the same allelic position of a founder. (More formal definition is omitted.) We note that this pairwise relation is clearly an equivalence relation.

Consider then the four alleles of two individuals at some locus. Label the individuals by $u$ and $v$, their maternal alleles by $a^u$ and $a^v$, and their paternal alleles by $b^u$ and $b^v$, respectively. Let $\equiv$ denote the identity-by-descent relation on the four alleles in a generic manner. Enumeration of all possible states of identity-by-descend on the set of four alleles gives 15 detailed identity states. Each of these states corresponds to a partition of the set $\{a^u, b^u, a^v, b^v\}$ into equivalence classes. For instance, suppose that the pair $(a^u, b^v)$ is the only identical by descent pair (i.e., $a^u \equiv b^v$) out of the six possible pairs. Then this detailed identity state corresponds to the partition $\{\{a^u, b^v\}, \{b^u\}, \{b^v\}\}$.

If the states that are mirror images of each other, due to maternal and paternal symmetry, are collapsed together, then nine states are left. These

\(^1\)Here an isomorphism does not need be faithful to gender.
6.1 The problem

*condensed identity states* can be characterized canonically by the following expressions [Lan97, Sha98]:

\[
S_1 : a^u \equiv b^u \equiv a^v \equiv b^v, \\
S_2 : a^u \equiv b^u \not\equiv a^v \equiv b^v, \\
S_3 : a^u \equiv b^u \equiv a^v \not\equiv b^v, \\
S_4 : a^u \equiv b^u \not\equiv a^v \not\equiv b^v, \\
S_5 : a^u \not\equiv b^u \equiv a^v \equiv b^v, \\
S_6 : a^u \not\equiv b^u \not\equiv a^v \equiv b^v, \\
S_7 : a^u \equiv a^v \not\equiv b^u \equiv b^v, \\
S_8 : a^u \equiv a^v \not\equiv b^u \not\equiv b^v, \\
S_9 : a^u \not\equiv a^v \not\equiv b^u \not\equiv b^v.
\]

While a detailed identity state corresponds to a partition of the set of four allelic loci, a condensed identity state corresponds to a set of partitions. For example, state \( S_8 \) comprises four partitions: \( \{\{a^u, a^v\}, \{b^u\}, \{b^v\}\}, \{\{a^u, b^v\}, \{b^u\}, \{a^v\}\}, \{\{b^u, a^v\}, \{a^u\}, \{b^v\}\} \) and \( \{\{b^u, b^v\}, \{a^u\}, \{a^v\}\} \). Collapsing symmetrical states (partitions) is natural when dealing with unordered genotypes.

The identity state patterns range from the case where all four alleles are i.b.d., to the case where none of the alleles is i.b.d. with any of the other alleles. We say that two persons are *genetic relatives* or *genetically related* if they are in one of the states \( S_1, S_3, S_5, S_7, \) or \( S_8 \), i.e., they share at least one allele i.b.d.. Concerning a single individual \( u \), we say that she is *genetically inbred* if \( a^u \equiv b^u \); otherwise she is *genetically outbred*. Let it be emphasized that all these definitions were given for some fixed locus. For example, identical twins are always in the state \( S_7 \) at every locus, provided that their parents are not genetically related. A mother and her child are always in the state \( S_8 \), provided that the mother is not genetically related to the father of the child. However, under the above definition, it could be that two siblings are not genetically related. For instance, if the parents of the siblings are not genetically related, then the siblings can be in state \( S_9 \). This motivates another definition for two individuals to be relatives.

We say that two individuals are *potential genetic relatives* or briefly *relatives* or *related* if they could be genetic relatives given only the pedigree information but not the allelic transmission paths. This definition of relatedness comes closer to the concept used in natural language. However, although in natural language the pedigree is understood in an informal way, here we need a more formal treatment.
We follow the idea that the relatedness and the relationship type of two individuals are defined by the topology of their ancestral pedigree. The possibility that the two individuals are genetic relatives depends on two things. First, it depends on the topology of their ancestral pedigree, so that, if the two individuals have a common ancestor (in their ancestral pedigree), then it is possible (though not necessary) that they share an allele i.b.d. Second, it depends on the genetic relatedness of the founders of the ancestral pedigree, so that, even in the case that the two individuals do not have a common ancestor, they could share an allele i.b.d. due to genetic relatedness of their (different) ancestors.

We simplify this setting by assuming that the founders of the ancestral pedigree are never genetically related. Of course, this is definitely an incorrect assumption, and so it should be understood as an idealizing approximation. Under this assumption the possibility of two individuals to be genetically related depends only on the topology of their ancestral pedigree. Moreover, under the simple Mendelian transmission model, the topology also specifies how probable it is that the two individuals are at some fixed locus in state $S_s$, for each $s = 1, \ldots, 9$. We use these probabilities to characterize the relationship types.

**Definition 6.2** Let $u$ and $v$ be two individual in pedigree $P$. The i.b.d. coefficient vector of $u$ and $v$ (in $P$) is a vector $\psi = (\psi_1, \ldots, \psi_9)$ such that the probability\(^2\) that the individuals $u$ and $v$ are in state $S_s$ equals $\psi_s$ for all $s = 1, \ldots, 9$. The component $\psi_s$ is called the $s$'th i.b.d. coefficients.

Based on this definition we give some more definitions. For a relationship type $R$ we define the corresponding i.b.d. coefficient vector $\psi^R$ in a natural way. We say that two individuals of relationship type $R$ are related or relatives if $\psi_1^R + \psi_3^R + \psi_5^R + \psi_7^R + \psi_9^R > 0$, that is, the probability that they share an allele i.b.d. is not zero. Concerning a single individual, we say that she is inbred if the probability that her two alleles are i.b.d. is greater than zero. This is equivalent to saying that her parents are relatives. Otherwise we call her outbred. We extend the above concepts to multiple loci as follows. Two individuals are called relatives if they are relatives at some locus. Also, a person is inbred if she is inbred at some locus; otherwise she is outbred. Finally, we define the kinship coefficient $\phi^R$ of relationship type $R$ as the probability that the two relatives have an i.b.d. allele at two randomly

\(^2\)Note that this probability is obtained as a marginal of an epistatic Mendelian model when augmented with the path indicators introduced in Remark 5.2 in Section 5.4.3. Here the locus and the pedigree (and all individuals in it) are fixed; randomness arises due to ignorance.
6.1 The problem

Table 6.1: Coefficients of Relatedness for Most Common Relationship Types

<table>
<thead>
<tr>
<th>Relationship Type</th>
<th>i.b.d. coefficient</th>
<th>Kinship coeff.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\psi_7$</td>
<td>$\psi_8$</td>
</tr>
<tr>
<td>identical twins (mz)</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>siblings</td>
<td>$1/4$</td>
<td>$1/2$</td>
</tr>
<tr>
<td>parent–offspring</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>half siblings</td>
<td>0</td>
<td>$1/2$</td>
</tr>
<tr>
<td>grandp.–offspring</td>
<td>0</td>
<td>$1/2$</td>
</tr>
<tr>
<td>uncle–nephew</td>
<td>0</td>
<td>$1/2$</td>
</tr>
<tr>
<td>first cousins</td>
<td>0</td>
<td>$1/4$</td>
</tr>
<tr>
<td>double first cousins</td>
<td>$1/16$</td>
<td>$6/16$</td>
</tr>
<tr>
<td>second cousins</td>
<td>0</td>
<td>$1/16$</td>
</tr>
<tr>
<td>nth-dg anc.–desc.</td>
<td>0</td>
<td>$(1/2)^{n-1}$</td>
</tr>
<tr>
<td>nonrelatives</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

picked allelic positions at a locus. It follows that $\phi^R = 1/2\psi_7^R + 1/4\psi_8^R$ for all outbred relationship types $R$. Table 6.1 displays the i.b.d. and kinship coefficients for some common relationship types.

In summary, we have considered mainly one type of coefficients of relatedness, the i.b.d. coefficients. In addition, we introduced the kinship coefficient as a derivative of the i.b.d. coefficients. These coefficients are defined with respect to a fixed relationship type in terms of the probabilities of the (condensed) identity states. The probability model is the simple Mendelian transmission model. An i.b.d. vector gives a succinct characterization for a relationship type. However, the mapping is not one-to-one, for an i.b.d. vector may correspond to different relationship types (e.g., half siblings and uncle–nephew). On the other hand, the topology-based definition of the relationship types only describes potential genetic relatedness. Namely, two pairs manifesting the same relationship type (e.g., two sibling pairs) may actually be in different identity states due to different topology of the allelic transmissions of the pairs. In that sense, the i.b.d. coefficient vector already captures the essential uncertainty pattern of the amount of shared genetic material when the actual transmission paths of the alleles are not known (or realized). This justifies the identification of the relationship types by the i.b.d. vectors when we consider computation of recurrence risks for various relationship types.
6.1.2 The problem of computing a recurrence risk

In Chapter 5 we stated the so-called direct problem, where the task is to compute the theoretical recurrence risk for a given relationship type in a fully specified genetic model. Here we restate the problem in a slightly simplified form. We consider only common relationship types so that we can assume that the corresponding i.b.d. coefficients are known (or precomputed). Note that the i.b.d. coefficients do not depend on the genetic hypothesis, provided that the simple Mendelian transmission model is assumed. Thus, the computation of those coefficients can be regarded as an independent problem.

**Definition 6.3** Given a hypothesis $\theta$ from the set of epistatic Mendelian hypotheses and a relationship type $R$ from the set of common relationship types, the recurrence risk computation problem (RRC) is to compute the recurrence risk $K_R(\theta)$.

What kind of solutions to the RRC problem are we interested in? Naturally, we aim at an algorithm that is correct and efficient. We will consider exact algorithms only, that is, deterministic algorithms that always output the exact correct value. We measure the efficiency of an algorithm by its time complexity in a model of computation, where arithmetic operations are assumed to take a constant time.

We notice that the input size of the problem is already exponential in the number of loci. Thus, the problem instances become rapidly computationally intractable as the number of loci grows. For this reason, our focus is on small, practical number of loci (up to three). Yet, it is reasonable to ask about the problem complexity. Could one find an algorithm that is cubic, quadratic, or even better? In this work we give only upper bounds on the complexity.

It turns out that the RRC problem is essentially a sum-product problem (as we already mentioned in Chapter 5). However, no related sum-product expression is directly visible in the problem definition. To formulate the essential sum-product problem, we have to express, in an appropriate way, the structure of the probability model specified by the given epistatic Mendelian hypothesis. Interestingly, we find two different approaches that lead to different algorithms for RRC.

We note that RRC is not a new problem. Campbell and Elston [CE71], for instance, must have dealt with related computational issues, as they run experiments using the maximum likelihood method. However, their paper [CE71] considers mainly some statistical modeling and representation aspects without any remark on the involved algorithmic problem. More
generally, RCC can be viewed as an old problem, since it is a special case of the general likelihood evaluation for segregation and linkage analysis in pedigrees. These computational tasks have been extensively studied during the last three decades [ES71, LE75, CTS78, LB83, LG87, O'C01, FG03]. In the next subsection we discuss some existing algorithms.

6.1.3 Why are advanced methods needed?

Recurrence risks are special cases of general likelihoods in pedigrees. Hence, a natural question is, can the general likelihood evaluation methods, such as the Elston–Stewart algorithm and the Lander–Green algorithm [ES71, LG87], be applied successfully to recurrence risk computations. In this section we discuss weaknesses of those methods. Thereby, we motivate our search for advanced methods.

Let us first recall the general likelihood evaluation problem. Let \((V, E)\) be a given pedigree where phenotypes \(y^U\) of a set \(U \subseteq V\) of individuals are observed. For a given genetic hypothesis \(\theta\) the task is to evaluate the probability of the observations. This likelihood can be expressed as

\[
p(y^U \mid \theta) = \sum_{y^{V-U}} \sum_{x^V} p(y^V, x^V \mid \theta),
\]

where \(p(y^V, x^V | \theta)\) is the joint probability of the phenotypes \(y^V\) and ordered genotypes \(x^V\) of the individuals in the pedigree. Recall that the joint probability enjoys the factorization given in Chapter 5 (in equation 5.3). This directly yields a sum-product problem, and so, the likelihood evaluation problem can be solved by the general variable elimination algorithm. The Elston–Stewart algorithm [ES71] and the Lander–Green algorithm [LG87] are special cases of the general variable elimination algorithm. The difference in the algorithms is mainly due to different elimination orderings. However, the Lander–Green algorithm also needs some extra independence assumptions, which we discuss soon.\(^3\)

The Elston–Stewart algorithm considers each individual in turn. The associated variables are eliminated one by one. The ordering depends on the topology of the pedigree (but, in principle, also on the sizes of the sets of possible genotypes per individual). In structurally simple pedigrees the arithmetic complexity of the algorithm depends on the costs of single elimination steps, and the effect of the pedigree size is only linear. However, the

\(^3\)We note that the Lander–Green algorithm is designed for linkage analysis where in addition to phenotype information also observation of alleles at a number of marker loci are used to infer the location of a trait locus. The Elston–Stewart algorithm is also used in linkage analysis, but it was originally designed for the analysis of pure phenotype data.
cost of a single elimination step can be fairly large. Namely, when elimin-
ating a child from a family, the variables associated with the parents need
to be tabulated due to the connection via the transmission probabilities.
The case is similar if eliminating a parent instead. Consequently, even in
simple pedigrees the arithmetic complexity is, roughly, cubic in the number
of genotypes. 4

The Lander–Green algorithm considers each locus in turn. The associ-
ated variables are eliminated one by one. This is efficient only when the
joint distribution $p(y^V, x^V \mid \theta)$ enjoys a factorization into components involv-
ing only one locus or two consecutive loci. Then the arithmetic complexity
grows linearly in the number of loci but exponentially in the number of
(non-founder) individuals in the pedigree. This factorization requirement
is not met in the general Elston–Stewart model nor in our special case,
in the epistatic Mendelian model. This is because we allow for arbitrary
epistasis in the penetrances. If the penetrance model is multiplicative, then
the Lander–Green algorithm would be applicable. On the other hand, the
Lander–Green algorithm is designed to handle linked loci. Consequently,
it seems to be too general for our case, where the loci are assumed to be
unlinked.

Recently there have been attempts to join the classical Elston–Stewart
and Lander–Green algorithms [O’C01, FG03]. Perhaps the most sound
of the attempts is the one proposed by Fishelson and Geiger [FG03]. In
their method the variable elimination is not forced to an individual-based
or locus-based order. Instead, they automatically choose a good variable
elimination order from the whole spectrum of hybrid orders. This extension
to the classical algorithms can be easily understood, and is the obvious
choice, when viewing the likelihood evaluation problem from the perspective
of the general sum-product framework. Though it is not easy to analyze
the complexity of the hybrid method, it seems evident that, again, allowing
for linkage does not come without a cost.

6.1.4 Preliminary observations

As an introductory to computational techniques to be presented in Sections
6.2 and 6.3, we make some elementary observations.

4More precisely, the arithmetic complexity is at least the number of compatible genotype configurations of a mother-father-child triplet. This number can be shown to equal $(m^1 - \binom{m}{2})^2$ when there are $m$ different alleles at $\ell$ loci. In the biallelic case this gives $15^\ell$ instead of the immediate bound of $27^\ell$. This reduces to $10^\ell$ when ignoring the order information (about paternal and maternal alleles).
6.1 The problem

First, we notice a consequence of the assumptions made in the epistatic Mendelian model.

**Proposition 6.4** Let $\theta$ be an epistatic Mendelian hypothesis. Let $u$ be an outbred individual. Then the distribution of $u$’s genotype is in equilibrium, i.e.,

$$p(x_u \mid \theta) = \prod_{i=1}^{\ell} q_i(x_i^u).$$

Moreover, we have $p(y_u = 1 \mid \theta) = K_P(\theta)$.

**Proof** Since $u$ is outbred, both of her alleles must have their origins at different allelic positions of one or more founders. The alleles of the founders are assumed to be independent. Furthermore, it is assumed that the transmission of the alleles is independent and uniform (the assumptions of the transmission model). Thus, we get the first equality of the claim. The second equality is an immediate consequence.

Second, we consider computation of the population prevalence.

**Theorem 6.5** Under the epistatic Mendelian model the population prevalence can be computed in linear time.

**Proof** Let $\theta$ be an $\ell$-locus epistatic Mendelian hypothesis. We consider a “pedigree” that consists of a founder $u$ only. The population prevalence under $\theta$ can be expressed as

$$p(y_u = 1 \mid \theta) = \sum_{z_u \in Z} p(z_u \mid \theta) p(y_u = 1 \mid z_u, \theta)$$

$$= \sum_{z_i^u \in Z_1} \cdots \sum_{z_\ell^u \in Z_\ell} \left[ \prod_{i=1}^{n} q_i(z_i^u) \right] f(z_1^u, \ldots, z_\ell^u).$$

Here $q_i(z_i^u)$ denote the frequencies of the locus-wise genotypes, for which we have

$$q_i(c_i, c'_i) = |\{c_i, c'_i\}| q_i(c) q_i(c') \quad \text{for } c, c' \in A_i.$$

We see that this can be evaluated by the variable elimination algorithm in time $O(\sum_{i=1}^{\ell} |Z_{\{1, \ldots, i\}}|)$, which is $O(|Z|)$ and thus linear with respect to the input size. 

\qed
6.2 Matrix decomposition and the Yates algorithm

In this section we consider a general method for the RRC problem. The development is general and applies to arbitrary relationship types. However, we focus on the case of symmetric penetrance and, consequently, operate mainly with unordered genotypes. We start by finding a matrix expression for recurrence risks; here we borrow much from earlier works presented in [LS54, CE71]. The idea is to separate the different types of parameters of the epistatic Mendelian hypothesis: the penetrances and allele frequencies. Then, we continue by factorizing the part of the expression that involves the allele frequencies. Thereby, we end up with a formula where each locus contributes multiplicatively to the frequency-dependent part. While computation of these locus-wise contributions are technically somewhat tricky to automatize, the overall complexity essentially lies on the coarser structure of the expression. We recognize a Yates transform problem, which naturally leads us to apply the Yates algorithm.

6.2.1 Quadratic form

A recurrence risk can be expressed as a ratio of the joint probability that the two relatives have the trait and the marginal probability that the proband has the trait.\(^5\) For both these probabilities, we have a simple formula that effectively separates the probabilistic relationship of relatives’ genotypes from the penetrances of the genotypes. We find a matrix notation convenient here.

**Definition 6.6** Let \( u \) and \( v \) be individuals of a relationship type \( R \). The *genotype matrix* of \( R \), denoted \( G^R \), is defined by

\[
G^R(z, z') = p(z^u = z, z^v = z' \mid \theta)
\]

for all unordered genotypes \( z, z' \in \mathcal{Z} \).

To express recurrence risks in matrix notation, we treat the penetrance function \( f \) as a column vector of length \( |\mathcal{Z}| \) with an appropriate indexing order. The transpose of \( f \) is denoted by \( f^T \). We let \( 1 \) denote a vector of length \( |\mathcal{Z}| \) with all components equal to 1.

**Theorem 6.7** For any relationship type \( R \) we have

\[
K_R(\theta) = \frac{f^T G^R f}{f^T G^R 1}.
\]

\(^5\)Recall that the proband individual is known to manifest the trait. The recurrence risk gives the probability that the proband’s relative is also affected.
6.2 Matrix decomposition and the Yates algorithm

**Proof** Let \( R \) be the relationship type of individuals \( u \) and \( v \). By the definition of recurrence risks and conditional probability,

\[
K_R(\theta) = p(y^v = 1 \mid y^u = 1, \theta) = \frac{p(y^v = y^u = 1 \mid \theta)}{p(y^u = 1 \mid \theta)}.
\]

Applying basic rules of probability gives

\[
p(y^v = y^u = 1 \mid \theta) = \sum_{z^u \in Z} \sum_{z^v \in Z} p(y^v = y^u = 1, z^u, z^v \mid \theta) = \sum_{z^u \in Z} \sum_{z^v \in Z} p(z^u, z^v \mid \theta)p(y^v = y^u = 1 \mid z^u, z^v, \theta).
\]

Since we work under the epistatic Mendelian model, the phenotypes are independent given the underlying genotypes. Hence,

\[
p(y^v = y^u = 1 \mid \theta) = \sum_{z^u \in Z} \sum_{z^v \in Z} p(z^u, z^v \mid \theta)p(y^u = 1 \mid z^u, \theta)p(y^v = 1 \mid z^v, \theta).
\]

We recognize this immediately as

\[
\sum_{z^u \in Z} \sum_{z^v \in Z} G^R(z^u, z^v) f(z^u) f(z^v) = f^T G^R f.
\]

Similarly, for the denominator we write

\[
p(y^u = 1 \mid \theta) = \sum_{z^u \in Z} p(z^u \mid \theta)p(y^u \mid z^u, \theta) = \sum_{z^u \in Z} \sum_{z^v \in Z} p(y^u \mid z^u, \theta)p(z^u, z^v \mid \theta) = f^T G^R 1,
\]

which completes the proof. \(\square\)

What is the use of the above formulation? Of course, a matrix formulation solely cannot help in computations. However, in the compact matrix notation the separation of genetic dependencies and penetrance effects becomes more transparent. The expression \( f^T G^R f \) can be recognized as a quadratic form. In probability theory and statistics, quadratic forms of this kind arise frequently, as they are closely related to mean-variance-covariance considerations of multiple variables. This viewpoint is also relevant here and is considered in detail later, in Section 6.3.

In this section we do not take that perspective. Instead, we follow a more direct approach to find a computationally efficient sum-product representation. We immediately notice that having the matrix \( G^R \) at hand, the
recurrence risk $K_R(\theta)$ could be computed straightforwardly in the quadratic $O(|Z|^2)$ time. This would readily be a nice result since the straightforward running of the Elston–Stewart algorithm would take at least a cubic time even for fairly simple relationship types.\(^6\) However, to achieve the quadratic running time, we have to be able to compute the matrix $G^R$ also in quadratic time. Fortunately, it turns out that this is not a problem, when we exploit some special properties of the epistatic Mendelian model that are not shared by the more general Elston–Stewart model. Actually, we get more. Namely, finding a decomposition of the matrix $G^R$ leads us to a sum-product formulation which can be evaluated even faster than in quadratic time.

### 6.2.2 Locus-wise decomposition

We exploit the following essential additional property of the epistatic Mendelian model over the Elston–Stewart model: The allelic events over multiple loci are independent over the loci. By an event we mean here statements that concern genotypes but not phenotypes. Intuitively this is clear, since in the epistatic Mendelian model we assume linkage equilibrium and independent Mendelian transmission of alleles from parent to offspring. Here we need a specialized version to decompose the genotype matrix.

**Proposition 6.8** Let $R$ be the relationship type of individuals $u$ and $v$. Then for all unordered genotypes $z$ and $z'$,

$$G^R(z, z') = p(z^u = z, z^v = z' \mid \theta) = \prod_{i=1}^\ell p(z^u_i = z_i, z^v_i = z'_i \mid \theta).$$

**Proof** Let $(V, E)$ be a pedigree that defines the relationship $R$ of individuals $u$ and $v$. A straightforward application of the general Elston–Stewart model with the independence assumptions of allele frequencies and transmission probabilities yields that

$$p(x^u, x^v \mid \theta) = \prod_{i=1}^\ell p(x^u_{\{i,i+\ell\}}, x^v_{\{i,i+\ell\}} \mid \theta)$$

for all ordered genotypes $x$ and $x'$. Since clearly $z^u_i$ only depends on $x^u_{\{i,i+\ell\}}$ (and similarly $z^v_i$) we get the factorization as claimed. \(\square\)

We adhere to matrix notation:

\(^6\)For relationship types with complex pedigrees the running time can be much worse.
Definition 6.9 Let $R$ be the relationship type of individuals $u$ and $v$. The \textit{i\textsuperscript{th} locus genotype matrix} of $R$ is defined by
\[
G_i^R(z_i, z_i') = p(z_i^u = z_i, z_i^v = z_i' | \theta).
\]

We restate the above result and recall the notion of tensor product (also called direct or Kronecker product; for definition and properties see, e.g., [MM92]).

Proposition 6.10 Let $R$ be a relationship type. Then the genotype matrix of $R$ is a tensor product of the locus-wise genotype matrices of $R$.

Proof For all unordered genotypes $z = (z_1, \ldots, z_\ell)$ and $z' = (z'_1, \ldots, z'_\ell)$ we have
\[
G^R(z, z') = G_1^R(z_1, z'_1) \cdots G_\ell^R(z_\ell, z'_\ell).
\]
Hence, by definition of tensor product (see, e.g., [MM92]),
\[
G^R = G_1^R \otimes \cdots \otimes G_\ell^R,
\]
where $\otimes$ denotes the tensor product operation for two matrices.

It is the above simple locus-wise decomposition that is the key to efficient evaluation of recurrence risks. By plugging the product expression into the quadratic form we get a sum-product expression. In Section 6.2.4 we will apply the Yates algorithm for that problem. Before that we, in Section 6.2.3, consider a remaining minor detail, namely, the construction of the locus-wise matrices for given relationship type and genetic hypothesis.

6.2.3 Computation of locus-wise matrices

To complete the locus-wise decomposition we show how the matrices $G_i^R$, can be constructed. We note that the matrices $G_i^R$ depend on both the relationship type $R$ and the allele frequencies of the genetic model. To separate these dependencies, we consider genotype probabilities conditionally on i.b.d. states.

Proposition 6.11 Given a condensed identity state of two individuals, the joint probability distribution of their unordered genotypes does not depend on their relationship type.
Proof Let $s$ be a condensed identity state. Let $r$ be a detailed version of $s$. We note that given the detailed state $r$ the distribution of ordered genotypes is a simple product of allele frequencies, one term per equivalence class. This does not depend on the relationship type. Yet, different detailed states correspond to different distributions of ordered genotypes. But, since the different detailed states are symmetric with respect to maternal and paternal positions, they must induce the same distribution of unordered genotypes.

This observation justifies the following definition.

**Definition 6.12** Let $u$ and $v$ be two individuals of a relationship type $R$, and $s$ a condensed identity state. The $i$'th locus genotype matrix of state $s$ is defined by

$$G_i^{(s)}(z_i^u, z_i^v) = p(z_i^u, z_i^v \mid S_{u,v} = s, \theta),$$

where $S_{u,v}$ denotes the condensed identity state of individuals $u$ and $v$. □

The following result shows that the matrices $G_i^R$ are linear combinations of the matrices $G_i^{(s)}$ with the i.b.d. coefficients. Note that the matrices $G_i^{(s)}$ only depend on the allele frequencies whereas the coefficients depend only on the relationship type $R$.

**Proposition 6.13** Let $R$ be a relationship type. Let $i$ be a locus. Then

$$G_i^R = \sum_{s=1}^{9} \psi_s^R G_i^{(s)}. \quad (6.1)$$

Proof Immediate from the definitions. □

Next we consider the computation of locus-matrices for given identity states. It is fairly obvious that when it comes to the overall complexity of computing recurrence risks, the computation of these matrices is not important. This is because the matrices depend only on the allele frequencies at a single locus, as is apparent in Definition 6.12. But from the practical point of view, algorithmic considerations are relevant, as they support implementation.

We introduce some notation. Recall that a detailed identity state of two individuals is defined as a partition of a set of four elements that corresponds to the individuals’ paternal and maternal alleles at a fixed locus $i$. For simplicity, we label these four elements by numbers 1, 2, 3, 4. Here 1 and 2 refer to the paternal and maternal allelic positions of the first relative,
respectively, whereas 3 and 4 refer to the paternal and maternal allelic positions of the second relative, respectively. Thus a detailed identity state is a partition of the set \{1, 2, 3, 4\}. A condensed identity state is a set of detailed identity states. If \{C_1, \ldots, C_k\} is a detailed identity state, then an allele assignment to the tuple \( (C_1, \ldots, C_k) \) is a tuple \( (c_1, \ldots, c_k) \), where \( c_j \in \mathcal{A}_i \) for all \( j = 1, \ldots, k \).

**Algorithm 6.14** (locus-wise matrix for an identity state)

**Input** allele frequencies \( q_i \) for a locus \( i \), a condensed identity state \( s \)

**Output** the matrix \( G_i^{(s)} \)

**Method**

1. initialize \( A \) as the \( |Z| \times |Z| \) matrix with all entries zero
2. let \( \{C_1, \ldots, C_k\} \) be any member in \( s \)
3. for each allele assignment \( (c_1, \ldots, c_k) \) to \( (C_1, \ldots, C_k) \) do
4.  for each \( j = 1, \ldots, k \) and \( t \in C_j \) do
5.  \( x(t) \leftarrow c_j \)
6.  \( z_i \leftarrow \{x(1), x(2)\} \)
7.  \( z_i' \leftarrow \{x(3), x(4)\} \)
8.  \( A(z_i, z_i') \leftarrow A(z_i, z_i') + \prod_{j=1}^{k} q_i(c_j) \)
9. return \( A \)

**Algorithm 6.15** (locus-wise matrix for a relationship type)

**Input** allele frequencies \( q_i \) for a locus \( i \), a common relationship type \( R \)

**Output** the matrix \( G_i^R \)

**Method**

1. run Algorithm 6.14 to obtain the matrices \( G_i^{(s)} \) for \( s = 1, \ldots, 9 \)
2. return \( \sum_{s=1}^{9} \psi_s G_i^{(s)} \)

**Theorem 6.16** Algorithm 6.15 works correctly.

**Proof** The algorithm trivially evaluates (6.1), provided that Algorithm 6.14 outputs a correct matrix for each \( s \). Therefore, it is sufficient to show that Algorithm 6.14 works correctly.

Consider the probability of an allele assignment \( (c_1, \ldots, c_k) \) to \( (C_1, \ldots, C_k) \). We note that alleles in different equivalence classes are independent, as their origins are at different founder positions. Therefore, the probability is obtained as the product \( \prod_j q_i(c_j) \). Each allele assignment induces a pair of ordered genotypes. In Algorithm 6.14 these genotypes are formed on lines 6 and 7.

But a genotype pair may be induced by many allele assignments. Thus, the probability of the genotype pair is the sum of the probabilities of these allele assignments. Obviously, these summations are computed on line 8 in Algorithm 6.14 when performed for different allele assignments. \( \square \)
Theorem 6.17  Algorithm 6.15 can be implemented to run in time $O(|A|^4)$.

Proof  Obvious, since the number $k$ and the sizes of the sets $C_j$ in Algorithm 6.14 are bounded by the constant four. □

6.2.4 Applying the Yates algorithm

We are now ready to present an algorithm for the RRC problem. From the preceding development we get that the recurrence risk of type $R$ under an epistatic Mendelian hypothesis $\theta$ can be expressed as

$$K_R(\theta) = \frac{f^T(G_1^R \otimes \cdots \otimes G_\ell^R)1}{f^T(G_1^R \otimes \cdots \otimes G_\ell^R)1}.$$

Here $f$ is the penetrance function specified by $\theta$ treated as a column vector. For each locus $i$, the matrix $G_i^R$ is specified by the relationship type and the allele frequencies of the locus. We have shown that these matrices can be constructed efficiently. Recognizing the Yates transform problem both in the denominator and numerator (see Remark 3.29 in Section 3.4) suggests the following method.

Algorithm 6.18  (recurrence risk computation using Yates transform)

Input an epistatic Mendelian hypothesis $\theta$, a common relationship type $R$

Output the recurrence risk $K_R(\theta)$

Method
1. run Algorithm 6.15 to obtain the matrices $\{G_i^R : i = 1, \ldots, \ell\}$
2. compute $w = f^T(G_1^R \otimes \cdots \otimes G_\ell^R)$ by running the Yates algorithm
3. return $(w \cdot f)/(w \cdot 1)$

Theorem 6.19  Algorithm 6.18 works correctly.

Proof  By Theorem 6.16 each $i$th locus genotype matrix for $R$ is correctly computed on line 1. Given these matrices as input, the Yates algorithm outputs a row vector $w$ of size $|Z|$ such that $w = f^T(G_1^R \otimes \cdots \otimes G_\ell^R)$. Hence, by Proposition 6.10 we obtain that the dot product $w \cdot f$ equals $f^T G^R f$ and similarly $w \cdot 1$ equals $f^T G^R 1$. Thus, by Theorem 6.7 the returned ratio equals $K_R(\theta)$, as claimed. □

Theorem 6.20  Algorithm 6.18 can be implemented to run in time $O(|Z| \sum_{i=1}^\ell |Z_i|)$. If $|Z_i| = n$ for all $i$, then this is $O(n^{\ell+1})$. For the case $\ell > 2$ the running time can be improved to $O(n^{\ell+0.34})$, and for the case $\ell = 2$, to $O(n^{2.38})$. 
6.3 Variance decomposition and fast Möbius transform

**Proof** Immediate consequence of the complexity of Yates transforms (Theorem 3.30 in Chapter 3).

6.3 Variance decomposition and fast Möbius transform

In this section we develop an alternative algorithm for RCC. Unlike the treatment in the previous section, we take here an indirect approach that is based on a standard variance decomposition representation [Kem57]. The idea can be summarized as follows. We treat the penetrance as a so called genotypic value, a random variable due to underlying randomness of genotypes. The main difference to the approach of the previous section is that here we find an orthogonal reparametrization of the penetrance function, i.e., of the genotypic value. Each of the orthogonal components is a function of the partial genotypes over a subset of allelic position. The variances of these components play a major role in our development.

Our first goal is to define the components $V_{A^s D^t}$, the total variance due to an $s$th order interaction of additive components and $t$th order interaction of dominance components. After that we can show Kemphorne’s [Kem57] result: the covariance of the genotypic values for two outbred relatives of type $R$ equals

$$
\sum_{0 \leq s+t \leq \ell} (2\phi^R)^s (\psi^R)^t V_{A^s D^t}.
$$

(6.2)

When the penetrance is taken as the genotypic value, the covariance equals the joint probability of the two relatives to have the trait [Jam71]. Recall that this is the quantity that essentially specifies the recurrence risk. Here the restriction to outbred relatives is essential. For inbred relatives a related covariance expression [Har64] becomes much more complex and is not useful from the perspective of computational complexity.

Our second goal is to design an algorithm for evaluating the above covariance expression, given the parameters of an epistatic Mendelian hypothesis. It turns out that the computational complexity of the method lies in the reparametrization step mentioned above, while the rest is then computationally rather straightforward. We recognize that the reparametrization into orthogonal components is actually defined via a Möbius transform on a subset lattice. This is the link to general sum-product problems leading us to apply the fast Möbius transform algorithm.

Since the above mentioned reparametrization and variance decomposition are rather general tools, we describe these techniques separately from
our particular genetic model. We consider a general setting of a function of multiple independent random variables. However, to obtain the covariance expression, the genetic model with its additional independence assumptions and genetic semantics is needed. For computation of the reparametrization and variance components, we derive general algorithms as well as specialized algorithms that are able to exploit certain properties of the epistatic Mendelian model.

6.3.1 Genotypic value as a function of multiple variables

Traditionally, the decomposition of genetic variance is associated with a quantitative trait [Fis18, Kem57]. The starting point is to model the phenotype of a random individual\(^7\) as a random variable \(Y\) that decomposes into a genetic effect \(G\) and an environmental effect \(E\) as \(Y = G + E\). It is already in the definition of the genetic effect that \(G\) is determined by the genotype of a random individual, that is, \(G\) is a deterministic function of the alleles over some gene loci. All variation in the phenotype among individuals who carry the same genotype is modeled by the environmental component \(E\). These effects, \(G\) and \(E\), are supposed to be uncorrelated, and hence, the phenotypic variance equals the sum of the genotypic and environmental variance, \(\text{Var}(Y) = \text{Var}(G) + \text{Var}(E)\). Regarding genetic risks, it is the genetic component and its variance that are interesting and a subject of further modeling.

In our case the phenotypic value of interest is a binary trait, treated as a 0/1-valued random variable \(Y\). It is the penetrance of a genotype that is taken as the genotypic value. Recall that in an \(\ell\)-locus epistatic Mendelian model the penetrance \(f\) is a function of \(\ell\) parental and \(\ell\) maternal alleles. These \(2\ell\) alleles together form an ordered genotype \(x\) whose probability distribution factorizes as

\[
p(x) = p(x_1)p(x_2)\cdots p(x_{2\ell-1})p(x_{2\ell}),
\]

given the allele frequencies (dropped out from the notation). For the decomposition of the variance of the genotypic value, i.e., the penetrance, this independence structure is exploited.

The orthogonalization technique is rather general and applicable in any setting with the above independence structure. For this reason we, in the next two subsections, first consider the orthogonalization technique in a general setting. After that, we return to the genetic model.

\(^7\)I.e., and individual that is picked from a population uniformly at random.
6.3.2 Orthogonalization and decomposition of variance

In this subsection we consider the following probability model. Let $n$ be a natural number. For indexing purposes let $L$ be a set of integers from 1 to $n$. Let $X_1, \ldots, X_n$ be independent random variables, that is, we consider a joint probability measure $P$ that factorizes as

$$P\{X_1 = x_1, \ldots, X_n = x_n\} = P\{X_1 = x_1\} \cdots P\{X_n = x_n\}$$

for all $x_1, \ldots, x_n$. For convenience we assume that each $X_i$ has a finite domain $\mathcal{A}_i$, hence, $P\{X_i \in \mathcal{A}_i\} = 1$. We use indexing by subsets in a generic way, and for the full scope, $L$, we may drop out the index and write briefly $X$ instead of $X_L$. We also hide the names of random variables in the notation and loosely denote a generic probability function by $p$. Under these conventions we may write the above factorization as $p(x) = p(x_1) \cdots p(x_n)$.

We consider functions from the full domain $\mathcal{A} = \mathcal{A}_L = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$ onto the set of real numbers. We also consider functions that are, in essence, defined on fewer variables, i.e., on a domain $\mathcal{A}_S$ with $S \subseteq L$. The following definition says when a function defined on all the variables is effectively defined already on fewer variables.

**Definition 6.21** Let $S$ be a subset of $L$. A function $f : \mathcal{A}_L \to \mathbb{R}$ is said to be a function over $S$ if $f(x) = f(y)$ for all such $x, y \in \mathcal{A}_L$ that $x_S = y_S$. In this case we write $f(x_S)$ instead of $f(x)$. Then $S$ is called the scope of $f$ and the size $|S|$ is called the arity of $f$. \hfill $\Box$

Note that, if $S$ is a subset of $T$, then any function over $S$ is also a function over $T$. Also note that any function over the empty set is a constant function. Yet, we treat all functions as if they were defined on the full domain $\mathcal{A}$. An advantage of this representation is the relative ease in adapting the definition of addition and multiplication operators, "+" and "\cdot", on functions via pointwise ordinary addition and multiplication on the real numbers.

When a function over $S \subseteq L$ is applied to a random variable $X_S$, the function value is also a random variable. Partial expectation operator, as defined next, can be viewed as a mapping that, by marginalization, reduces the effective arity of such a function.

**Definition 6.22** Let $S$ and $T$ be subsets of $L$. The partial $T$-expectation of a function $f$ over $S$ is the function $g$ over $S - T$, defined by

$$g(x_{S-T}) = \sum_{x_T} p(x_T)f(x_S).$$
The corresponding expectation operator is denoted by $E_T$. Hence, above we have $g = E_T(f)$. 

Note that the $T$-expectation operator, for any $T \subseteq L$, is a mapping from the set of real-valued functions defined on the full domain $\mathcal{A}$ onto itself. This mapping enjoys the following useful properties.

**Proposition 6.23 (properties of partial expectations)** Let $f$ be a function over $U$ and $g$ a function over $L$. Then for all $S, T \subseteq L$,

(a) $E_L(f) = E(f)$, where $E$ the ordinary expectation operator;

(b) $E_T(f + g) = E_T(f) + E_T(g)$;

(c) $E_T(fg) = f E_T(g)$, if $T$ and $U$ are disjoint;

(d) $E_T(E_S(f)) = E_{T \cup S}(f)$.

**Proof** Obvious. 

We aim at a decomposition of an arbitrary function over $L$ into mutually uncorrelated components. This can be done by an orthogonalization transformation to be described next.

**Definition 6.24** Let $S$ be a subset of $L$. For a function $f$ over $L$ the $S$-marginal is the function defined by

$$f_S := E_{L - S}(f),$$

and the $S$-component is the function defined recursively by

$$\tilde{f}_S := f_S - \sum_{T \subseteq S} \tilde{f}_T.$$

We may call an $S$-marginal simply marginal and an $S$-component simply component. 

**Proposition 6.25** If $f$ is a function over $L$, then $f = \sum_{S \subseteq L} \tilde{f}_S$.

**Proof** Obvious from Definition 6.24 since $f = f_L$. 

The above definition and result together yield the desired decomposition for an arbitrary function over $L$. Our remaining task is to show that the components are uncorrelated. In the next two lemmas we will consider an arbitrary but fixed function $f$ over $L$. 

6.3 Variance decomposition and fast Möbius transform

**Lemma 6.26** If $S$ and $T$ are subsets of $L$ with a common element, then $E_T(\tilde{f}_S) = 0$.

**Proof** Let $T, S \subseteq L$. Using first the definition of $\tilde{f}_S$ and the linearity of partial expectation, and then the property (d) and the definition of the marginals, gives

$$E_T(\tilde{f}_S) = E_T(f_S) - \sum_{S' \subseteq S} E_T(\tilde{f}_{S'}) = f_{S-T} - \sum_{S' \subseteq S} E_T(\tilde{f}_{S'}) - \sum_{S' \subseteq S} E_T(\tilde{f}_{S'}) = 0,$$

We then proceed with induction on $|S|$. Suppose therefore first that $S = \{r\}$, where $r \in T$. It follows that

$$E_T(\tilde{f}_S) = f_{\{r\}-T} - E_T(\tilde{f}_\emptyset) = f_\emptyset - f_\emptyset = 0,$$

since $\tilde{f}_\emptyset = f_\emptyset$ is a constant. Then suppose that $E_T(\tilde{f}_{S'}) = 0$ whenever $|S'| < |S|$ and $S' \cap T \neq \emptyset$. Substitution into the equation derived above yields

$$E_T(\tilde{f}_S) = f_{S-T} - \sum_{S' \subseteq S} 0 - \sum_{S' \subseteq S} \tilde{f}_{S'} = f_{S-T} - \sum_{S' \subseteq S-T} \tilde{f}_{S'} = 0.$$

Here we used the fact that $E_T(\tilde{f}_{S'}) = \tilde{f}_{S'}$ if $T$ and $S'$ are disjoint (the property (c)).

**Lemma 6.27** If $S$ and $T$ are two different subsets of $L$, then the random variables $\tilde{f}_S$ and $\tilde{f}_T$ are uncorrelated.

**Proof** Let $S \neq T$. By the definition of covariance and by Lemma 6.26,

$$\text{Cov}(\tilde{f}_S, \tilde{f}_T) = E(\tilde{f}_S \tilde{f}_T) - E(\tilde{f}_S) E(\tilde{f}_T) = E(\tilde{f}_S \tilde{f}_T).$$

Without loss of generality we may assume that $T - S \neq \emptyset$. By the property (d), the expectation over $L$ can be taken in two phases, first over $(T - S)$ and then over the remaining $L - (T - S)$. We get

$$E(\tilde{f}_S \tilde{f}_T) = E_{L-(T-S)}(E_{T-S}(\tilde{f}_S \tilde{f}_T)) = E_{L-(T-S)}(\tilde{f}_S E_{T-S}(\tilde{f}_T)) = 0.$$

Here, we used property (c) to conclude that $E_{T-S}(\tilde{f}_S \tilde{f}_T) = \tilde{f}_S E_{T-S}(\tilde{f}_T)$, and Lemma 6.26 to conclude that $E_{T-S}(\tilde{f}_T) = 0$. 

We have shown that for any $f$ the components are mutually uncorrelated. Thus, the collection of the components is an orthogonal representation of $f$. We get the following decomposition of the variance of $f$. 
Theorem 6.28 Let $f$ be a function over $L$. The variance of $f$ decomposes as

$$\text{Var}(f) = \sum_{S \subseteq L} \text{Var}(\tilde{f}_S).$$

Proof Take variances on both sides of the decomposition equation of Proposition 6.25. By Lemma 6.27 the random variables $\{\tilde{f}_S\}$ are uncorrelated.

The above development borrows much from Bulmer’s representation [Bul80, Ch. 4]. However, there are also some differences. Namely, Bulmer defines $S$-marginals and assumes the relationship of marginals and components stated in Proposition 6.25. From this he derives the recurrence relationship of $S$-components stated in Definition 6.24. Thus, compared to the above presentation, Bulmer’s definition of $S$-components is somewhat implicit.

6.3.3 Computation of variance components

In this subsection we consider algorithmic computation of orthogonalization and variance components under the general framework of the previous subsection. Henceforth we let $f$ denote the function of interest. We consider three associated computational problems and give three algorithms. Following the definitions of the previous subsection we first consider the computation of the set of $S$-marginals of $f$, then the computation of the set of $S$-components of $f$ given the set of $S$-marginals, and finally, the computation of the variance components given the components.

The obvious algorithm for computing the set of $S$-marginals of $f$ is as follows.

Algorithm 6.29 (marginals)

Input a function $f$ over $L$ and a probability model $p$

Output the set $\{f_S : S \subseteq L\}$

Method

1. $g_L \leftarrow f$
2. $\mathcal{F} \leftarrow \{g_L\}$
3. for each $S \subseteq L$ in decreasing order of size do
4.   choose $j \in L - S$
5.   $S' \leftarrow S \cup \{j\}$
6.   for each $x_S \in \mathcal{A}_S$ do
7.     $g_S(x_S) \leftarrow \sum_{x_j \in \mathcal{A}_j} p(x_j) g_{S'}(x_{S'})$
8.   $\mathcal{F} \leftarrow \mathcal{F} \cup \{g_S\}$
9. return $\mathcal{F}$
6.3 Variance decomposition and fast Möbius transform

**Proposition 6.30** Algorithm 6.29 works correctly.

**Proof** We note that for any $S$ on lines 3–8, there is at least one subset $S'$ so that $S' - S$ contains exactly one element that can be chosen as a $j$ on line 5. We also note that then the $f_{S'}$ has already been computed as a $g_{S'}$. By a simple induction,

$$g_S = E_j(g_{S'}) = E_j(f_{S'}) = E_j(E_{L-S'}(f)) = E_{(L-S') \cup \{j\}}(f) = f_S,$$

concluding the proof. □

**Proposition 6.31** Algorithm 6.29 can be implemented to run in $O(m2^n)$ arithmetic operations on the function space, where $m$ is the maximum of the domain sizes $|\mathcal{A}_1|, \ldots, |\mathcal{A}_n|$. The running time (in the conventional model of computation) is $O((m + 1)^n \min\{m, n\})$.

**Proof** The number of additions and multiplications performed on lines 6–7 equals $|\mathcal{A}_j| \leq m$. Hence, the total number of additions and multiplications is bounded by $(2^n - 1)m$, proving the first statement.

For the latter result we observe that the total number of additions and multiplications performed on line 6 is given by

$$\sum_{S \subseteq L} m^{|S|+1} = m \sum_{l=0}^{n-1} \binom{n}{l} m^l = m(m+1)^n - m^{n+1}. \quad (6.3)$$

Using the general identity, $a^n - b^n = (a - b)(a^{n-1} + a^{n-2}b + \ldots + ab^{n-2} + b^{n-1})$, the upper bound (6.3) becomes

$$m \sum_{l=0}^{n-1} (m + 1)^{n-1-l} m^l \leq nm(m + 1)^{n-1} \leq n(m + 1)^n. \quad (6.4)$$

Combining the bounds (6.3) and (6.4) yields the claimed asymptotic requirement. A complication is that the straightforward evaluation of the summation on line 6 requires $|\mathcal{A}_{S'}|$ table lookups. When the functions are represented as obvious tree structures, a single table lookup would take $O(|S'|)$ time. However, it is not difficult to see that the summation can be implemented to run in $O(|\mathcal{A}_{S'}|)$ time by simultaneous traversing of the trees of $g_{S'}$ and $g_S$ (see Proposition 3.13 in Chapter 3). Thus, the claimed time requirement holds even when taking the table lookups into account. □

We then consider the problem of computing the components given the marginals. First note that a straightforward recursive implementation of
Definition 6.24 would give an algorithm that takes $\Omega(\sum_{S \subseteq L} 2^{|S|}) = \Omega(3^n)$ additions and subtractions on the function space. Similarly, the time requirement would be much larger than that of the above algorithm for computing the marginals. It turns out that a more efficient algorithm exists.

To derive an efficient algorithm for computing the components, we first show that the components are actually defined as a Möbius inversion of the marginals.

**Proposition 6.32** Let $f$ be a function over $L$ and $S$ a subset of $S$. Then

$$\tilde{f}_S = \sum_{T \subseteq S} (-1)^{|S-T|} f_T. \quad (6.5)$$

**Proof** Definition 6.24 gives immediately that

$$f_S = \sum_{S \subseteq T} \tilde{f}_T \quad \text{for all } S \subseteq L.$$  

Hence, by Proposition 3.34 (in Chapter 3),

$$\tilde{f}_S = \sum_{T \subseteq S} (-1)^{|S-T|} f_T \quad \text{for all } S \subseteq L,$$

as claimed. \qed

The above reexpression of the components immediately suggests the use of the fast Möbius inversion algorithm (see Theorem 3.37 in Chapter 3) to compute the components given the marginals. The following algorithm directly applies the general Möbius inversion algorithm on a subset lattice.

**Algorithm 6.33** (components)

Input a set $\{f_S : S \subseteq L\}$ of marginals

Output the set $\{\tilde{f}_S : S \subseteq L\}$ of components

Method

1. return the Möbius inversion of $\{f_S : S \subseteq L\}$

**Proposition 6.34** Algorithm 6.33 works correctly.

**Proof** Follows immediately from Proposition 6.5. \qed

**Proposition 6.35** Algorithm 6.33 can be implemented to run in $\mathcal{O}(n2^n)$ arithmetic operations on the function space. If $m$ is the maximum of the domain sizes $|A_1|, \ldots, |A_n|$, then the running time is $\mathcal{O}(n(m+1)^n)$. 
6.3 Variance decomposition and fast Möbius transform

Proof Follows immediately from the complexity of general Möbius inversion (Theorem 3.37).

We are now ready to consider the problem of computing the variance components given the set of component functions. The following algorithm computes the variance components one by one in a straightforward manner.

Algorithm 6.36 (variance components)
Input a set \( \{\hat{f}_S : S \subseteq L\} \) of components and a probability model \( p \)
Output the set \( \{\text{Var}(\hat{f}_S) : S \subseteq L\} \) of variance components
Method
1. \( V \leftarrow \emptyset \)
2. for each \( S \subseteq L \) do
3. \[ v \leftarrow \sum_{x_S \in A_S} p(x_S)\hat{f}_S(x_S)^2 \]
4. \( V \leftarrow V \cup \{v\} \)
5. return \( V \)

Proposition 6.37 Algorithm 6.36 works correctly.

Proof Obviously, it is sufficient to show that the value of \( v \) computed on line 3 equals the variance \( \text{Var}(\hat{f}_S) \). To see this, note that \( \text{Var}(\hat{f}_S) = \text{E}(\hat{f}_S^2) - \text{E}(\hat{f}_S)^2 \), where the expectation is taken with respect to \( p(x) \). By Lemma 6.26 we have \( \text{E}_S(\hat{f}_S) = 0 \). Hence, \( \text{E}(\hat{f}_S) = 0 \) and \( \text{Var}(\hat{f}_S) = \text{E}(\hat{f}_S^2) = \text{E}_S(\hat{f}_S^2) \), and thus, exactly as computed on line 3.

Proposition 6.38 Algorithm 6.36 can be implemented to run in time \( O((m + 1)^n) \), where \( m \) is the maximum of the domain sizes \( |A_1|, \ldots, |A_n| \).

Proof It is clear that the time complexity is proportional to \( \sum_{S \subseteq L} |A_S| \). An upper bound for this is

\[
\sum_{S \subseteq L} m^{|S|} = \sum_{k=0}^{n} \binom{n}{k} m^k = (m + 1)^n \text{,}
\]

as claimed.

We summarize the above algorithmic development. Putting the three given algorithms together gives an algorithm that computes the variance components of a given function \( f \) over \( L \). The running time of the algorithm is obtained as the sum of the running times of the three algorithms. We see that the time complexity is dominated by that of Algorithm 6.33 (computation of components) yielding the total complexity of \( O(n(m+1)^n) \).
6.3.4 Covariance representation of recurrence risks

The general orthogonalization method considered in the previous subsections can be directly applied to decomposition of genetic variance. The set $L$ is now taken to be the set of allelic positions, $\{1, \ldots, 2\ell\}$, and $\mathcal{A}_i = \mathcal{A}_{i+\ell}$ the set of possible alleles at locus $i$. Also recall that the penetrance function $f$ serves as the genotypic value whose decomposition we are interested in. Our goal in this section is to show Kempthorne’s covariance representation (6.2) which is the key in designing an efficient algorithm for the RRC problem.

Kempthorne’s covariance representation is based on condensed variance components. The condensed representation of the variance components is obtained by merging components of the same type. Loosely speaking, two components are of the same type if the components cover equally many doubles and singletons of the set of allelic positions. More precisely, we have the following definition.

**Definition 6.39** Let $S$ be a subset of $L$ with $s$ doubled and $t$ singleton loci. We say that $S$ holds an $s$th order additive interaction and $t$th order dominance interaction. Denote the set of such subsets $S$ by $\mathcal{L}(t, s)$. The **total variance due to an $s'$th order interaction of additive components and $t'$th order interaction of dominance components** is defined as

$$V_{A^sD^t} = \sum_{S \in \mathcal{L}(t, s)} \text{Var}(\hat{f}_S).$$

We also call these quantities more briefly **interaction variance components**. □

To express the covariance of genotypic values of two related individuals we need to consider the joint probability model of their genotypes. Such a model is specified by any epistatic Mendelian model. We state the following result of Kempthorne [Kem57] without a proof.\(^8\)

**Theorem 6.40** ([Kem57]) For two random individuals $u$ and $v$ of an outbred relationship type $R$ the covariance of the genotypic value $f$ is given by

$$\text{Cov}(f^u, f^v) = \sum_{0 \leq s + t \leq \ell} (2\phi^R)_{s}^{(R)}(\psi^R_{t})^t V_{A^sD^t}.$$

\(^8\)We note that Kempthorne gave this result for “random-mating populations”. The assumptions of the random mating model are satisfied by the epistatic Mendelian models.
When the penetrance function is taken as the genotypic value the covariance of the genotypic value has an immediate connection to recurrence risks. For a penetrance function \( f \) and two random individuals \( u \) and \( v \) of an outbred relationship type \( R \) we let \( \text{Cov}_R \) denote the covariance \( \text{Cov}(f^u, f^v) \). The following observation is due to James [Jam71].

**Theorem 6.4.1** For an outbred relationship type \( R \),

\[
K_R = K_P + \text{Cov}_R / K_P.
\]

**Proof** Write

\[
\text{Cov}_R = \text{Cov}(f^u, f^v) = E(f^u f^v) - E(f^u)E(f^v)
\]

and note that \( E(f^u f^v) = p(y^u = y^v = 1) \), \( E(f^u) = p(y^u = 1) \), and \( E(f^v) = p(y^v = 1) \). Since \( u \) and \( v \) are outbred we have \( p(y^u = 1) = p(y^v = 1) = K_P \). Hence,

\[
K_R = p(y^u = y^v = 1) / p(y^u = 1) = \text{Cov}_R / K_P + K_P,
\]

completing the proof. \( \square \)

### 6.3.5 Computation of recurrence risks

We next apply the general algorithms of Section 6.3.3 to the RRC problem. From theorems 6.40 and 6.41 we notice that having the variance components \( V_{A^* D^*} \) in hand, the recurrence risk for an outbred relationship type can be computed easily. Thus, the computational complexity lies essentially in the computation of the variance components. Consequently, we start by considering computation of the variance components \( \text{Var}(\hat{f}_S) \) (for all \( S \subseteq L \)). We tune the general algorithms developed in Section 6.3.3 so that they can take advantage of the symmetry property of the penetrance function, namely that the penetrance is the same for all ordered versions of an unordered genotype. Exploiting this symmetry is expected to result in a significant reduction of computational complexity. In the end of this section, we put the pieces together and present an algorithm for the RRC problem in the case of symmetric penetrance.

We first consider computation of the marginals of \( f \). There are two related symmetry properties to be exploited. Before going into more technical considerations, recall that a subset \( S \) of \( L \) may consist of two types of elements. If both \( i \) and \( i + \ell \) belong to \( S \), then \( i \) and \( i + \ell \) are said to be *doubled* in \( S \). If just one or the other belongs to \( S \), then that one is said to
be a singleton in $S$. A locus $i = 1, \ldots, \ell$ is called doubled if $i$ and $i + \ell$ are
doubled in $S$, and single if $i$ or $i + \ell$ is single in $S$. This terminology helps
in describing the following observations about the symmetry properties of the
penetrance.

An immediate observation is that for a marginal $f_S$ it suffices to compute
the values $f_S(x_S)$ for unordered partial genotypes $x_S$ only. That is, for
doubled loci $i \in S$ it suffices to consider the representations $\{x_i, x_{i+\ell}\} \subseteq
A_i$ instead of the ordered pairs $(x_i, x_{i+\ell}) \in A_i \times A_{i+\ell}$. An equivalent,
constrained pair representation is, however, more convenient: for any $S \subseteq L$
and $x_S \in A_S$, we define $\beta_S(x_S)$ as an element $x'_S$ in $A_S$ such that $x'_i = x_i$
for all singletons $i \in S$, and $\{x'_i, x'_{i+\ell}\} \subseteq \{x_i, x_{i+\ell}\}$ with $x'_i \leq x'_{i+\ell}$
for all doubled loci $i \in S$. Hence, the image of the mapping $\beta_S$ defines the set of
partial unordered genotypes $B_S = \beta_S(A_S) \subseteq A_S$.

A related finding is that although the marginals $f_S$ are well defined
for all subsets $S$ of $L$, not all of them are distinct. For example, let the
number of loci be $\ell = 2$ and let $S = \{1, 2, 3\}$ and $T = \{1, 2, 4\}$. Then
$f_S(x_S) = f_T(x_T)$ whenever $x_3 = x_4$. (Note that $x_{\{1, 2\}}$ is shared by $x_S$ and
$x_T$.) This means that $f_S$ and $f_T$ are effectively the same; given that $f_S$ has
already been computed the computation of $f_T$ could be virtually avoided.
This observation can, of course, be extended to arbitrary $S$ and $T$ that differ
only at the singleton elements. We let $L$ denote the condensed collection
of the subsets of $L$, i.e., $L = \{S \subseteq L : \text{if } i + \ell \in S \text{ then also } i \in S\}$.

The following algorithm for computing the marginal penetrances takes
advantage of the above observations. Note that we require the functions $f_S$
to be computed for the arguments in $B_S$ only.

**Algorithm 6.42** (marginals of symmetric penetrance)

**Input** a penetrance function $f$ and a probability model $p$

**Output** the set $\{f_S : S \in L\}$ of marginals

**Method**

1. $g_L \leftarrow f$
2. $F \leftarrow \{g_L\}$
3. for each $S \in L - \{L\}$ in decreasing order of size do
   4. choose $S' \in L$ such that $S' = S \cup \{j\}$ for $j \not\in S$
   5. for each $x_S \in B_S$ do
      6. $g_S(x_S) \leftarrow \sum_{x_j \in A_j} p(x_j) g_{S'}(\beta_{S'}(x_{S'}))$
   7. $F \leftarrow F \cup \{g_S\}$
8. return $F$

**Proposition 6.43** Algorithm 6.42 works correctly.

**Proof** By induction on the size of the subsets $S \in L$ we show that $g_S$
equals $f_S$. The case $S = L$ is trivial. Then assume that for $S \in L$ we have
6.3 Variance decomposition and fast Möbius transform

\[ g_{S'} = f_{S'} \] for all \( S' \in \mathcal{L} \) such that \( S \subset S' \). Then this also holds for the particular \( S' = S \cup \{j\} \) chosen on line 4. Thus, for all \( x_S \in \mathcal{B}_S \),

\[ g_S(x_S) = \sum_{x_j \in A_j} p(x_j) f_{S'}(\beta_{S'}(x_{S'})) = \sum_{x_j \in A_j} p(x_j) f_{S'}(x_{S'}) = f_S(x_S), \]

since \( f_{S'}(\beta_{S'}(x_{S'})) = f_{S'}(x_{S'}) \) by the symmetry of the penetrance.

**Proposition 6.44** Algorithm 6.42 can be implemented to run in time \( \mathcal{O}\left(\left(\frac{m+2}{2}\right)^\ell \cdot \min\{\ell, m\}\right) \), where \( m \) is the maximum of the domain sizes \( |A_1|, \ldots, |A_\ell| \).

**Proof** The total number of arithmetic operations performed on line 6 is obviously proportional to

\[ \sum_{S \in \mathcal{L} - \{L\}} |\mathcal{B}_S||A_j|. \]

Supposing that \( |A_i| \leq m \) for all \( i \), we get a tight upper bound

\[ u(m, \ell) = \sum_{k=0}^{\ell} \binom{\ell}{k} \sum_{s=0}^{k} \binom{k}{s} \left( \frac{m+1}{2} \right)^s m^{k-s} - \left( \frac{m+1}{2} \right)^\ell m. \]

Here \( k \) refers to the size of \( S \) and \( s \) to the number of doubled loci in \( S \). Simplification gives

\[
\begin{align*}
    u(m, \ell) &= m \sum_{k=0}^{\ell} \binom{\ell}{k} \sum_{s=0}^{k} \binom{k}{s} \left( \frac{m+1}{2} \right)^s m^{k-s} - \left( \frac{m+1}{2} \right)^\ell m \\
    &= m \left[ \sum_{k=0}^{\ell} \binom{\ell}{k} m^k \left( \frac{m+1}{2} + 1 \right)^k - \left( \frac{m+1}{2} \right)^\ell \right] \\
    &= m \left[ \left( \frac{m+1}{2} + m + 1 \right)^\ell - \left( \frac{m+1}{2} \right)^\ell \right].
\end{align*}
\]

On one hand, ignoring the last term gives

\[ u(m, \ell) \leq m \left( \frac{m+2}{2} \right)^\ell. \]

On the other hand, by applying \( a^n - b^n \leq (a - b)na^{n-1} \) we get

\[ u(m, \ell) \leq m(m+1)\ell \left( \frac{m+2}{2} \right)^{\ell-1}. \]
Combining these two bounds gives the claim.

Finally, we note that traversing thru the subsets $S$ and evaluation of $g_{S'}(\beta_{S'}(x_{S'}))$ can be done with a negligible additional cost. \hfill \Box

We then consider computation of the components, which are obtained by a Möbius inversion from the marginals. To exploit the symmetry property of the penetrance, we tune the general fast Möbius inversion algorithm. Here, it is convenient to use an indicator representation for the subsets of $L$. The indicator representation of a set $S \subseteq L$ is a vector $(S_1, \ldots, S_{2\ell})$ such that $S_j = 1$ if $j \in S$ and $S_j = 0$ otherwise. It is also helpful to introduce a mapping $\gamma$ that maps a subset $S$ of $L$ to an element $S' = \gamma(S)$ of $L$ such that $S$ and $S'$ consist of the same singleton and doubled loci. Note that $\gamma$ is unique.

**Algorithm 6.45** (components of symmetric penetrance)

**Input** a set $\{f_S : S \in \mathcal{L}\}$ of marginals

**Output** the set $\{\hat{f}_S : S \in \mathcal{L}\}$ of components

**Method**

1. for each $S \in \mathcal{L}$ do
2. \hspace{1em} $g_S^{(0)} \leftarrow f_S$
3. for each $j = 1, \ldots, 2\ell$ (in this order) do
4. \hspace{1em} for each $S \in \mathcal{L}$ do
5. \hspace{2em} $g_S^{(j)} \leftarrow \sum_{T_j \leq S_j} (-1)^{S_j-T_j} g_{\gamma(T)}^{(j-1)}$, where $T_i = S_i$ for all $i \neq j$.
6. \hspace{1em} return $\{g_S^{(2\ell)} : S \in \mathcal{L}\}$

**Proposition 6.46** Algorithm 6.45 works correctly.

**Proof** We show by induction on $j$ that for all $S \in \mathcal{L}$

$$g_S^{(j)} = \sum_{T_1 \leq S_1} \cdots \sum_{T_j \leq S_j} (-1)^{\sum_{i=1}^j T_i} f_{\gamma(T)}, \quad (6.6)$$

where $T_i = S_i$ for all $i > j$. The case $j = 0$ is trivial, since $T = S = \gamma(T)$ and $g_S^{(0)} = f_S$ holds due to the substitution on line 2.

Suppose then that (6.6) holds for all $j < k \leq 2\ell$. Let $S \in \mathcal{L}$. The substitution on line 5 gives

$$g_S^{(k)} = \sum_{T_k \leq S_k} (-1)^{S_k-T_k} g_{\gamma(S')}^{(k-1)},$$

where $S' = (S_1, \ldots, S_{k-1}, T_k, S_{k+1}, \ldots, S_{2\ell})$. By the induction assumption we may write

$$g_S^{(k)} = \sum_{T_k \leq S_k} (-1)^{S_k-T_k} \sum_{T_1 \leq S'_1} \cdots \sum_{T_{k-1} \leq S'_{k-1}} (-1)^{\sum_{i=1}^{k-1} S'_i-T_i} f_{\gamma(T_1, \ldots, T_{k-1}, S'_k, \ldots, S'_{2\ell})},$$

and this completes the proof.
where \( S'' = \gamma(S') \). It is now sufficient to show that

\[
\gamma(T_1, \ldots, T_{k-1}, S''_{k+1}, \ldots, S''_{2\ell}) = \gamma(T_1, \ldots, T_{k-1}, T_k, S_{k+1}, \ldots, S_{2\ell}).
\]

(6.7)

Let \( k' \) be a position such that \( k = k + \ell \) or \( k = k - \ell \). Obviously, \( S''_i = S'_i = S_i \) for \( i \neq k, k' \). But we also have \( \{S''_{k}, S''_{k'}\} = \{S'_k, S'_k\} = \{T_k, S_{k'}\} \). Hence, (6.7) follows.

We have shown by induction that (6.6) holds for all \( j = 0, 1, \ldots, 2\ell \). Now, in particular,

\[
g^{(2\ell)}_S = \sum_{T_1 \leq S_1} \cdots \sum_{T_{2\ell} \leq S_{2\ell}} (-1)^{\sum_{i=1}^{2\ell} s_i - T_1} f_{\gamma(T_1, \ldots, T_{2\ell})} = \sum_{T \leq S} (-1)^{|S - T|} f_T = \hat{f}_S,
\]

since \( f_{\gamma(T)} = f_T \) due to the symmetry of \( f \). This completes the proof. \( \square \)

**Proposition 6.47** Algorithm 6.45 can be implemented to run in time \( \mathcal{O}(\ell^{(m+2)/2}) \), where \( m \) is the maximum of the domain sizes \( |A_1|, \ldots, |A_{\ell}| \).

**Proof** Consider first the additions on line 5. At most two additions of two functions are performed, both with domain \( B_S \) or one with domain \( B_S \) and the other with domain \( B_{S''} \), where \( S'' = \gamma(S_1, \ldots, S_{j-1}, 0, S_{j+1}, \ldots, S_{2\ell}) \).

Note that \( S'' \subseteq S \). Suppose that each function \( g^{(j)}_S \), where \( S \in \mathcal{L} \), is represented as a tree. Then the additions on line 5 can obviously be computed in time \( \mathcal{O}(|B_S|) \). The total time complexity is seen to be proportional to

\[
\ell \sum_{S \in \mathcal{L}} |B_S|.
\]

Supposing that \( |A_i| \leq m \) for all \( i \) we get a tight upper bound

\[
u(m, \ell) = \ell \sum_{k=0}^{\ell} \binom{\ell}{k} \sum_{s=0}^{k} \binom{k}{s} \left( \frac{m+1}{2} \right)^s m^{k-s}.
\]

Here \( k \) refers to the size of \( S \) and \( s \) to the number of doubled loci in \( S \). Simplification gives

\[
u(m, \ell) &= \ell \sum_{k=0}^{\ell} \binom{\ell}{k} \sum_{s=0}^{k} \binom{k}{s} \left( \frac{m+1}{2} \right)^s m^k
\]

\[= \ell \sum_{k=0}^{\ell} \binom{\ell}{k} m^k \left( \frac{m+1}{2} + 1 \right)^k
\]

\[= \ell \left( \binom{m+1}{2} + m + 1 \right)^\ell
\]

\[= \ell \left( \binom{m+2}{2} \right)^\ell,
\]

\[= \ell \left( \binom{m+2}{2} \right)^\ell,
\]
concluding the proof.

\[ \square \]

**Remark 6.48** We note that the calculations in the proof of Proposition 6.47 could be avoided by using a simple combinatorial argument. Namely, add a new element, say 0, to each domain \( A_i \). Then there are \( \binom{m+2}{2} \) possible unordered pairs (from \( A_i \cup \{0\} \)) at every locus, and hence, \( \binom{m+2}{2} \ell \) unordered genotypes. This is precisely the sum of the sizes \( |B_S| \), since a single 0 (heterozygous genotype) at locus \( i \) contributes (only) to the cases where \( i \) is a member of \( S \) while \( (i + \ell) \) is not, and a double 0 (homozygous genotype) contributes (only) to the cases where neither \( i \) nor \( (i + \ell) \) belongs to \( S \). \[ \square \]

Given the components of penetrance it is rather straightforward to compute the variance components. Yet, care must taken because we want to perform computation using the unordered genotypes. Recall that the generic probability model \( p \) directly specifies the probabilities for ordered genotypes only. For unordered genotypes we get the probability by multiplying the probability of its ordered version by a factor that corresponds to the number of heterozygous loci. We denote the number of heterozygous loci of \( x_S \in A_S \) by \( \text{het}(x_S) \) and the probability of an unordered genotype with a representation \( x_S \in B_S \) by \( p'(x_S) \). Thus, we have \( p'(x_S) = 2^{\text{het}(x_S)} p(x_S) \).

**Algorithm 6.49** (variance components of symmetric penetrance)

**Input** components \( (\tilde{f}_S : S \in \mathcal{L}) \) and a probability model \( p \)

**Output** variance components \( \{\text{Var}(\tilde{f}_S) : S \in \mathcal{L}\} \)

**Method**

1. for each \( S \in \mathcal{L} \) do
2. \( v_S \leftarrow \sum_{x_S \in B_S} p'(x_S) \tilde{f}_S(x_S)^2 \)
3. return \( \{v_S : S \in \mathcal{L}\} \)

**Proposition 6.50** Algorithm 6.49 works correctly.

**Proof** By the proof of Proposition 6.37 it is sufficient to show that each \( v_S \) computed on line 2 equals \( \text{ES}(\tilde{f}_S^2) \), where the expectation is with respect to the distribution \( p(x_S) \). To see this, write first

\[
\sum_{x_S \in B_S} p'(x_S) \tilde{f}_S(x_S)^2 = \sum_{x_S \in B_S} 2^{\text{het}(x_S)} p(x_S) \tilde{f}_S(x_S)^2.
\]

Then note that each \( x_S \in B_S \) corresponds to \( 2^{\text{het}(x_S)} \) ordered versions, i.e., \( |\beta_S^{-1}(x_S)| = 2^{\text{het}(x_S)} \). By the symmetry of \( p \) and \( \tilde{f}_S \), the above equals

\[
\sum_{x_S \in A_S} p(\beta_S(x_S)) \tilde{f}_S(\beta_S(x_S))^2 = \sum_{x_S \in A_S} p(x_S) \tilde{f}_S(x_S)^2 = \text{ES}(\tilde{f}_S^2),
\]
6.3 Variance decomposition and fast Möbius transform

completing the proof.

**Proposition 6.51** Algorithm 6.49 can be implemented to run in time $O\left(\left(\frac{m+2}{2}\right)^{\ell}\right)$, where $m$ is the maximum of the domain sizes $|A_1|, \ldots, |A_n|$.

**Proof** It is clear that the time complexity is proportional to $\sum_{S \in \mathcal{L}} |B_S|$. Similarly as calculated in the proof of Proposition 6.47, a tight upper bound is found to be $\left(\frac{m+2}{2}\right)^{\ell}$, as claimed.

We next proceed to consider computation of the interaction variance components. In principle, the computation is straightforward based on Definition 6.39. However, we need to be careful as we have computed the variance components for elements in the restricted subset collection $\mathcal{L}$ only. The following algorithm takes account of this.

**Algorithm 6.52** (interaction variance components of symmetric penetrance)

**Input** variance components $(\text{Var}(\hat{f}_S) : S \in \mathcal{L})$  

**Output** interaction variance components $\{V_{A^sD^t} : 0 \leq s + t \leq \ell\}$

**Method**
1. for positive integers $s$ and $t$ such that $0 \leq s + t \leq \ell$ do
2. \hspace{1em} $v_{st} \leftarrow 0$
3. \hspace{1em} for each $S \in \mathcal{L}$ do
4. \hspace{2em} let $s$ be the number of doubled loci in $S$
5. \hspace{2em} let $t$ be the number of singleton loci in $S$
6. \hspace{2em} $v_{st} \leftarrow v_{st} + 2^t \text{Var}(\hat{f}_S)$
7. \hspace{1em} return $\{v_{st} : 0 \leq s + t \leq \ell\}$

**Proposition 6.53** Algorithm 6.52 works correctly.

**Proof** By Definition 6.39 and the symmetry of the penetrance,

$$V_{A^sD^t} = \sum_{S \in \mathcal{L}(t,s)} \text{Var}(\hat{f}_{\gamma(S)}).$$

We observe that the set $\{\gamma^{-1}(S) : S \in \mathcal{L}(t,s)\} = \{\gamma^{-1}(S) : S \in \mathcal{L}(t,s) \cap \mathcal{L}\}$ is a partition of $\mathcal{L}(t,s)$. Moreover, for all $S \in \mathcal{L}(t,s) \cap \mathcal{L}$ we have $|\gamma^{-1}(S)| = 2^t$ (that many different choices for the positions of the singleton loci). Hence, we get

$$V_{A^sD^t} = \sum_{S \in \mathcal{L}(t,s) \cap \mathcal{L}} 2^t \text{Var}(\hat{f}_S).$$

This is seen to be equal to the value of $v_{st}$ after the second loop (lines 3–6). This completes the proof.
Proposition 6.54 For an $\ell$-locus epistatic Mendelian model with symmetric penetrance the interaction variance components can be computed in time $O\left(\ell\binom{m+2}{2}^\ell\right)$, where $m$ is the maximum of $|A_1|, \ldots, |A_\ell|$.

Proof The variance components can be computed by running serially Algorithms 6.45, 6.49, and 6.52. By Propositions 6.47, 6.51, and 6.54 the time complexity is $O\left(\ell\binom{m+2}{2}^\ell\right)$. Thus, it remains to show that this bound also holds for Algorithm 6.52. But this is obvious, since the sets $S \in \mathcal{L}$ can clearly be traversed in time $O(|\mathcal{L}|) = O(3^\ell)$ so that $s$ and $t$ as well as the factor $2^t$ can also be computed in a constant (amortized) time. \qed

Finally, we are ready to conclude the presented variance decomposition method by a summarizing algorithm for the RRC problem.

Algorithm 6.55 (recurrence risk computation by variance components)

Input an epistatic Mendelian hypothesis $\theta$ and a common outbred relationship type $R$

Output the recurrence risk $K_R(\theta)$

Method
1. compute the interaction variance components $V_{A^*,D^*}$
2. compute the covariance $\text{Cov}_R$
3. compute the population prevalence $K_P$
4. return $K_P + \text{Cov}_R/K_P$

Correctness of Algorithm 6.55 is clear by Theorem 6.41 that relates the phenotypic covariance and recurrence risks. Based on Algorithm 6.55, the following theorem bounds the time complexity of the RRC problem.

Theorem 6.56 For an $\ell$-locus epistatic Mendelian model with nonsymmetric penetrance the recurrence risk of any common outbred relationship type can be computed in time $O\left(\ell\binom{m+2}{2}^\ell\right)$, where $m$ is the maximum of $|A_1|, \ldots, |A_\ell|$.

Proof We analyze the time completely of Algorithm 6.55. By Theorem 6.56 the interaction variance components can be computed in time $O\left(\ell\binom{m+2}{2}^\ell\right)$. The covariance $\text{Cov}_R$ can be computed in a straightforward manner according to the expression given in Theorem 6.40. Clearly, this takes $O(\ell^2)$ time. Finally, by Theorem 6.5 the population prevalence $K_P$ can be computed in linear time, $O\left(\binom{m+1}{2}^\ell\right)$. Hence, the complexity of Algorithm 6.55 is bounded as claimed. \qed
6.4 An inequality for recurrence risks

It is intuitively clear that the closer two individuals are related, the higher the recurrence risk is, for any trait. Here we give a formal result that relates the i.b.d. coefficients and recurrence risks.

**Theorem 6.57** Let $R$ and $R'$ be two outbred relationship types such that
\[ \phi^R \geq \phi^{R'} \quad \text{and} \quad \psi^R_7 \geq \psi^{R'}_7. \]
(6.8)
Then $K_R(\theta) \geq K_R'(\theta)$ for all epistatic Mendelian hypotheses $\theta$ with symmetric penetrance. \[ \square \]

An interesting corollary to this result orders sibling risk and offspring risk.

**Corollary 6.58** For any epistatic Mendelian hypothesis the sibling risk is at least as high as the offspring risk. \[ \square \]

Note that on average, a sibling shares half of the genes with another sibling, and a child shares half of the genes with a parent. Thus the validity of the inequality is by no means obvious.

There are at least two ways to prove Theorem 6.57. We first give a proof based on the decomposition of variance. This proof is simple, but it is built on the quite tedious development given in Section 6.3.2 and Kempthorne’s representation (Theorem 6.40). The second proof is based on the matrix decomposition given in Section 6.2. Both proofs have been published in [KM01].

**Proof of Theorem 6.57 via variance decomposition.** Following the statement of the theorem suppose that $R$ and $R'$ are two outbred relationship types such that (6.8) holds. Then for all nonnegative integers $s$ and $t$, we have
\[ (2\phi^R)^s(\psi^R_7)^t \geq (2\phi^{R'})^s(\psi^{R'}_7)^t. \]
Hence, by the covariance representation (Theorem 6.40),
\[ \text{Cov}_R \geq \text{Cov}_{R'} \]
for all epistatic Mendelian hypotheses, since the interaction variances are obviously nonnegative. Finally, by Theorem 6.41,
\[ K_R \geq K_{R'} \]
for all epistatic Mendelian hypotheses, since the population prevalence $K_P$ is obviously nonnegative. (In the case that $K_P = 0$, the recurrence risks are not defined.) \[ \square \]
Proof of Theorem 6.57 via matrix decomposition. Let $R$ and $R'$ be two outbred relationship types such that (6.8) holds. Consider an arbitrary epistatic Mendelian hypothesis $\theta$.

Recall that $G^R_i$ denotes a $|Z_i| \times |Z_i|$ matrix whose elements are the probabilities $p(z_i^u, z_i^v \mid \theta)$ for two relatives $u$ and $v$ of the type $R$. If $S = \{s_1, \ldots, s_k\}$ is a subset of loci with $1 \leq s_1 < \cdots < s_k \leq \ell$, then we denote

$$G^R_S = G^R_{s_1} \otimes \cdots \otimes G^R_{s_k},$$

and similarly $G^{R'}_S$. We will first show by induction on the number of loci $|S|$ that the matrices

$$D^-_S = G^R_S - G^{R'}_S \text{ and } D^+_S = G^R_S + G^{R'}_S$$

are positive semidefinite (psd). Recall that a matrix $A$ is psd if and only if $w^T A w \geq 0$ for all vectors $w$ (of the appropriate length). It will turn out that showing the single locus case, $|S| = 1$, is quite involved but the induction step can be carried out with a surprising ease.

Single locus case: Let $i \in S$. By Proposition 6.13 we have

$$G^R_i = \sum_{s=1}^9 \psi^R_s G^{(s)}_i. \quad (6.9)$$

Since $R$ is outbred we have $\psi^R_1 + \psi^R_8 + \psi^R_9 = 1$ and $\psi^R_j = 0$ for all $j < 7$. By the definition of the kinship coefficient, $\phi^R = 1/2\psi^R_7 + 1/4\psi^R_8$. Solving $\psi^R_8 = 4\phi^R - 2\psi^R_7$ and $\psi^R_9 = 1 + \psi^R_7 - 4\phi^R$ and substituting into (6.9) gives

$$G^R_i = \psi^R_7 (G^{(7)}_i - 2G^{(8)}_i + G^{(9)}_i) + 4\phi^R (G^{(8)}_i - G^{(9)}_i) + G^{(9)}_i. \quad (6.10)$$

Similar result is, of course, obtained for $R'$. We get

$$D^+_i = (\psi^R_7 - \psi^R_8)(E_i - F_i) + 4(\phi^R - \phi^{R'})(F_i, \quad (6.11)$$

$$D^-_i = (\psi^R_7 + \psi^R_8)(E_i - F_i) + 4(\phi^R + \phi^{R'})(F_i) + 2G^{(9)}_i,$$

where $E_i = G^{(7)}_i - G^{(8)}_i$ and $F_i = G^{(8)}_i - G^{(9)}_i$. To show that $D^-_i$ and $D^+_i$ are psd it is now sufficient to show that the matrices $E_i - F_i$, $F_i$, and $G^{(9)}_i$ are psd.

Let $m$ be the number of alleles at the locus $i$. Denote the population frequency of $j$th allele by $q_j$. Let $w$ be an arbitrary column vector of length $\binom{m+1}{2} = m(m+1)/2$. In the indexing $w_{jh}$, where $1 \leq j, h \leq m$, the index pair is understood as an unordered genotype. Thus, $w_{jh} = w_{hj}$. 


The following expressions are easy to find.

\[
 w^T G_i^{(7)} w = \sum_j q_j \sum_k q_k w_{jk}^2, \tag{6.11}
\]

\[
 w^T G_i^{(8)} w = \sum_j q_j \left( \sum_k q_k w_{jk} \right)^2, \tag{6.12}
\]

\[
 w^T G_i^{(9)} w = \left( \sum_j q_j \sum_k q_k w_{jk} \right)^2. \tag{6.13}
\]

We see immediately that \( G_i^{(9)} \) is psd. To show that \( F_i \) and \( E_i - F_i \) are also psd, we apply the following identity (of variance)

\[
 \sum_{j=1}^m q_j \lambda_j^2 - \left( \sum_{j=1}^m q_j \lambda_j \right)^2 = \frac{1}{2} \sum_{j=1}^m q_j \sum_{k=1}^m q_k (\lambda_j - \lambda_k)^2 \tag{6.14}
\]

that holds for all real numbers \( \lambda_1, \ldots, \lambda_m \). Substituting \( \lambda_j = \sum_h q_h w_{jh} \) gives

\[
 w^T F_i w = \frac{1}{2} \sum_{j=1}^m q_j \sum_{k=1}^m q_k \left( \sum_h q_h (w_{jh} - w_{kh}) \right)^2. \tag{6.15}
\]

This expression shows that \( F_i \) is psd, but soon it is also used for showing that \( E_i - F_i \) is psd. Applying (6.14) to the inner sums of (6.11) and (6.12) with \( \lambda_k = w_{kj} \) yields

\[
 w^T E_i w = \frac{1}{2} \sum_{j=1}^m q_j \sum_{k=1}^m q_k \sum_{h=1}^m q_h (w_{kj} - w_{hj})^2
 = \frac{1}{2} \sum_{j=1}^m q_j \sum_{k=1}^m q_k \sum_{h=1}^m q_h (w_{jh} - w_{kh})^2, \tag{6.16}
\]

after the permutation \((j,k,h) \mapsto (h,j,k)\). Note that we swapped the labels \( k \) and \( j \) in (6.11) and (6.12). Finally, applying (6.14) once again to the innermost sums of (6.15) and (6.16) with \( \lambda_h = w_{jh} - w_{kh} \) gives

\[
 w^T (E_i - F_i) w = \frac{1}{2} \sum_{j=1}^m q_j \sum_{k=1}^m q_k \frac{1}{2} \sum_{h=1}^m q_h \sum_{l=1}^m q_l (w_{jh} - w_{kh} - w_{jl} + w_{kl})^2
\]

Thus, \( E_i - F_i \) is psd. This completes the single locus case.

\textit{Induction step:} Let \( S \) be a subset of loci. Let \( i = \max S \) and denote \( S' = S - \{i\} \). We have

\[
 D_{S}^- = G_{S'}^R \otimes G_i^R - G_{S'}^{R'} \otimes G_i^{R'} \quad \text{and} \quad D_{S}^+ = G_{S'}^R \otimes G_i^R + G_{S'}^{R'} \otimes G_i^{R'}.
\]
(For the properties of tensor products see, e.g., [MM92].) We then apply the following identities

\[ A \otimes B - C \otimes D = \frac{1}{2} (A - C) \otimes (B + D) + \frac{1}{2} (A + C) \otimes (B - D) \]

and

\[ A \otimes B + C \otimes D = \frac{1}{2} (A + C) \otimes (B + D) + \frac{1}{2} (A - C) \otimes (B - D). \]

This gives us

\[ 2D_S^- = D_{S'}^+ \otimes D_i^- + D_{S'}^- \otimes D_i^+ \quad \text{and} \quad 2D_S^+ = D_{S'}^+ \otimes D_i^+ + D_{S'}^- \otimes D_i^- . \]

By the induction assumption, the four matrices \( D_{S'}, D_{S'}^+, D_i^-, \) and \( D_i^+ \) are psd. Using the fact that the tensor product preserves positive semidefiniteness (see, e.g., [MM92, Theorem 14.12.7]) gives that the matrices \( D_S^- \) and \( D_S^+ \) are psd.

In particular, we have now shown that

\[ f^T (G^R - G^{R'}) f \geq 0, \]

where \( f \) is the vector representation of the penetrance function specified the (arbitrary) hypothesis \( \theta \). Hence, since \( R \) and \( R' \) are outbred types, we have

\[ K_R K_P - K_{R'} K_P \geq 0. \]

Obviously, \( K_P \) is nonnegative. The inequality, \( K_R \geq K_{R'} \), follows, which completes the proof. \( \square \)

### 6.5 Concluding remarks

We have studied the problem of computing the recurrence risk for a given relationship type under a given epistatic Mendelian hypothesis. We derived two algorithms that were essentially based on a Yates transform and a Möbius transform, respectively. Compared to existing generic methods, such as the Elston–Stewart algorithm [ES71], the presented algorithms are very fast: time requirements are linear with respect to the input size, up to a logarithmic factor.

Which one of these two algorithms is faster? To investigate this, recall that the complexity of the method based on matrix decomposition is \( O(\ell^{m+1}\binom{m+1}{2} + 1) \) whereas the complexity of the method based on variance component is \( O(\ell^{m+2}\binom{m+2}{2}) \). Both these bounds are rather tight with respect to
both problem parameters $m$ and $\ell$. Here we ignore the possible asymptotic acceleration by fast matrix multiplication. To see when the matrix-based method is faster, we solve the inequality

$$
c \ell \left( \frac{m+1}{2} \right)^{\ell+1} \geq c' \ell \left( \frac{m+2}{2} \right)^{\ell},
$$

where $c$ and $c'$ stand for the hidden constant factors of the time bounds. After some easy manipulation we get a lower bound for $\ell$ as a function $m$, as

$$
\ell \leq \frac{\ln(m+1) + \ln m - \ln(c'/c)}{\ln(m+2) - \ln m}.
$$

For small $m$ a rough approximation is $2.5m - 2$, assuming $c = c'$. As the typical values might be $\ell = 3$ and $m = 2$, this consideration—although theoretically interesting—does not help much in choosing the algorithm.

The presented analysis leads to some natural questions. Does there exist a finer problem representation such that the two algorithms are just variable elimination algorithms on the same sum-product problem, but with different elimination orderings? Do there exist faster algorithms for the RRC problem? The nature of these questions is, however, rather theoretic, as both algorithms run in almost optimal, linear time.

It is worth noting that the variance decomposition method is more restricted than the matrix-based method. Namely, it applies to outbred relatives only. On the other hand, both methods assume symmetric penetrance. The matrix-based method, however, is straightforward to extend to handle nonsymmetric penetrance as well, provided that the i.b.d. coefficients for detailed identity states are available.

These two restrictions appear also in the inequality of recurrence risks (Theorem 6.57). An open question is, whether the inequality also holds for nonsymmetric penetrance. If not, can one prove some similar characterization in terms of detailed i.b.d. coefficients? An open problem is also to incorporate outbred relationships into the inequalities. For the former two questions, both matrix decomposition and variance decomposition approaches might be fruitful. For the latter problem, the matrix-based approach seems more promising, as extensions of the variance component expressions can be horrible (see, e.g., [Har64]).
Chapter 7

Exploration of genetic models by sampling

This chapter combines the presented methodology to analyze recurrence risk data; the task, called the inverse inference problem, was outlined in Chapter 5. The applicability of the methods is approached from three perspectives. First, the tempered integration method and coupled Markov chain Monte Carlo simulation are demonstrated in the application context. Second, we illustrate the Bayesian inference machinery in the presence of nonidentifiability, and investigate the relations of sample size, prior information, and Bayesian power. Third, we discuss how much useful knowledge Bayesian recurrence risk analysis may provide regarding genetic studies of complex traits. The fourth main theme of the thesis, exact sum-product algorithms, is not directly discussed, though selected algorithms have been implemented as a necessary part of the software.

This chapter is organized as follows. We start in Section 7.1 by briefly recalling the notions of recurrence risk data and epistatic Mendelian model. Section 7.2 provides some details concerning the implementation of the MCMC method. In Section 7.3, experiments on synthetic and two real data sets are reported. Conclusions are made in Section 7.4.

7.1 Data and model revisited

Throughout this chapter we suppose that there is data collected on some relationship types, for some trait. We use the notation introduced in Chapter 5. Thus, $K_P$ denotes the unknown population prevalence and $N_P$ and $M_P$ the corresponding sample sizes. For cousin, offspring, sibling, and identical twin, we replace $P$ by $C, O, S,$ and $I,$ respectively. We refer to the set of included relationship types by $R = \{P, C, O, S, I\}$. The data set $(N_R, M_R)$
is denoted by $D$. We are also interested in the effect of dropping the cousin risk from the data, which can be handled by putting $N_C = M_C = 0$.

The data is analyzed under the epistatic Mendelian model. We restrict our attention to one-, two-, and three-locus models. Each locus is modeled by two alleles. The parameters are taken as $\theta = (q, f)$, where the number of loci $\ell$ is implicit in the notation and $q$ and $f$ are allele frequencies and penetrances, respectively, for the $\ell$-locus model. We consider a Bayesian model, i.e., a joint distribution over the data and the parameters. The model decomposes into a prior and likelihood as $p(\theta, D) = p(\theta)p(D \mid \theta)$. We follow the assumptions and idealizations discussed in Chapter 5, and hence, the likelihood is given by

$$p(D \mid \theta) = p(M_R \mid N_R, \theta) = \prod_{R \in \mathcal{R}} \binom{N_R}{M_R} K_R(\theta)^{M_R} (1 - K_R(\theta))^{N_R - M_R}.$$

Recall that here the sample sizes $N_R$ are assumed to be known.

We use simple priors. We let the number of loci, $\ell$, be uniform on $\{1, 2, 3\}$. Given $\ell$, we let $q$ and $f$ be independent. For $q$ we assign the uniform prior. For $f$ we assign either the uniform or the monotonic uniform prior (see Chapter 5).

Given a data set, various posterior quantities are in our interests. Generally, the question is what can be said about the model parameters. We know that the parameters are nonidentifiable. Therefore, point estimates make no sense. But there is Bayesian learning and that is what we try to summarize. We do this by considering the marginal posterior distributions of the parameters. In particular, we focus on the posterior of the number of loci and the joint posterior distribution of the allele frequencies.

We need to evaluate complex integrals. The posterior probability of $\ell$ loci is given by

$$p(\ell \mid D) \propto p(D \mid \ell) = \int p(\theta \mid \ell) p(D \mid \theta) d\theta, \quad (7.1)$$

since the prior on the number of loci is uniform. Here the integral is over the parameter space of the $\ell$-locus model. In other words, we need to evaluate the marginal likelihood of the $\ell$-locus model for each $\ell = 1, 2, 3$. For the joint allele frequency distribution of $\ell$-locus hypotheses, we have

$$p(q \mid D) = \int p(q, f \mid D) df. \quad (7.2)$$

This is a marginal of the full posterior distribution.
7.2 Implementing tempered Markov chain Monte Carlo

We are also interested in the Bayesian power of inferring the number of loci based on data of given size (i.e., for fixed $N_R$). Recall that by definition (see Section 2.2) the Bayesian power is the prior probability that the posterior guess will be correct. Thus, it is an expectation over the joint distribution of (genetic) hypotheses and (recurrence risk) data sets of a given size. Hence, an expression for the Bayesian power is given by

$$\sum_D \int p(D, \theta) 1(\ell = \hat{\ell}(D))d\theta,$$

where $\hat{\ell}(D) = \arg \max_{\ell'} p(\ell' | D)$. Note that the Bayesian power is a function of data size and model $p$. It tells us how large a data set is needed to correctly infer the number of loci with high (subjective) probability. Similarly, we consider the expected information gain concerning the number of loci. An appropriate measure is the expected entropy,

$$- \sum_D \int p(D, \theta) \sum_{\ell=1}^{3} p(\ell | D) \log_2 p(\ell | D) d\theta.$$

These quantities can be estimated by a Monte Carlo procedure based on a sample from the joint distribution $p(D, \theta)$.

7.2 Implementing tempered Markov chain Monte Carlo

The tempered MCMC method described in Chapter 4 is almost ideal for the Bayesian analysis of recurrence risk data under the epistatic Mendelian model. Below we recall the three-fold benefit of the method, discussed in Chapter 3, from the perspective of recurrence risk data analysis. After that, we continue by considering certain implementation issues.

One aspect is the inference of the number of loci, i.e., the dimension of the model. It seems that there are no effective closed-form solutions to the required integrals (7.1). Therefore, we resort to simulation methods. We note that the popular reversible jump MCMC [Gre95] for Bayesian inference of model dimension is not easy to implement in this case. Namely, the parameter representations of models of different dimensions are not nested, even though the models themselves are, in the sense discussed in Chapter 5. Consequently, we approach the posterior inference of the number of loci by approximating the marginal likelihoods separately for each number of loci, $\ell = 1, 2, 3$. Then we compute approximations of the posterior probabilities
via Bayes’ rule. The marginal likelihoods are tricky to approximate. Sophisticated techniques, like those described in Chapter 4, are required. We note that due to nonidentifiability of the model parameters and relatively large data sizes, the likelihood “surface” can be expected to include multiple modes, steep ridges, and sharp peaks.

The second aspect is that the exploration of parameters other than the number of loci is likewise difficult. For example, there seems to exist no closed-form solution to the marginal posterior of the allele frequencies (7.2). By MCMC we get a sample approximation, provided that we are able to sample from the posterior effectively. Again, since the likelihood surface is expected to be complex, sophisticated methods are motivated. The Metropolis coupling of tempered chains answers to this need, as generally discussed in Chapter 4.

The third justification of the tempered MCMC method lies in the meta analysis point of view. The effect of data size can be studied based on a single run of the MC^3 algorithm, as explained in Chapter 4. More precisely, given a data set D we consider integrals of the form

$$\int p(\theta \mid \ell) p(D \mid \theta)^{\tau_i} d\theta,$$

where $\tau_i \in [0, 1]$ is the temperature of chain $i$. Accordingly, chain $i$ simulates from the posterior distribution that corresponds to a data set where the original numbers $N_R$ and $M_R$ are multiplied by $\tau_i$, for each relationship type $R$.^1 To estimate the Bayesian power for learning the correct number of loci, as a function of the data size, we just have to generate a number of large data sets; see Section 7.3.2 for details.

The generic MC^3 algorithm was described in Chapter 4; here we consider certain implementation details that need special care.

Several factors affect the construction of the tempering scheme. As discussed in Chapter 4, it seems to be difficult to find an optimal scheme. We consider a heuristic family of tempering schemes, where the temperature of chain $i$ is given by

$$\tau_i = (1 + \gamma^{k/2 - i})^{-1},$$

where $0 < \gamma < 1$ and $K + 1$ is the total number of chains. For small $i$, temperatures $\tau_i$ tend to 1, and for large $i$ they tend to 0, and $\tau_{k/2} = 1/2$. An advantage is that only two parameters, $k$ and $\gamma$, need to be chosen.

^1Note that the integral (7.3) is proportional to the marginal likelihood $p(D_i \mid \ell)$, where $D_i$ is obtained by taking a $\tau_i$-fraction of $D$. The missing constant involves binomial coefficients and it depends on $D$ and $\tau_i$ but not on $\ell$ nor $\theta$. 
In the experiments, these parameters were tuned based on a number of preliminary runs.

Another design issue is that of the proposal distribution. We follow the standard MC³ algorithm and compose the proposal from two parts: one proposes a simple swap of the parameters between two adjacent chains, and the other proposes new parameters within a chain. We next specify the latter component in more detail. Treat the parameters as a vector \( \theta = (\theta_1, \ldots, \theta_m) \), and denote by \( M \) the index set \( \{1, \ldots, m\} \), so that \( |M| = m \). We construct the proposal distribution as a mixture,

\[
q(\theta' \mid \theta) = \sum_{J \subseteq M} a_J p_J(\theta' \mid \theta),
\]

where the proportions \( a_J \) are nonnegative and sum up to one, and each \( p_J \) is a proposal distribution that suggests updating the parameter block \( \{\theta_j : j \in J\} \). We use simple, constrained uniform distributions:

\[
a_J = \begin{cases} 
\frac{1}{r} \left( \frac{|M|}{|J|} \right)^{-1} & \text{if } 1 \leq |J| \leq r, \\
0 & \text{otherwise};
\end{cases}
\]

\[
p_J(\theta' \mid \theta) = \begin{cases} 
p(\theta'_J \mid \theta_{M-J}) & \text{if } \theta'_j = \theta_j \text{ for all } j \notin J, \\
0 & \text{otherwise.}
\end{cases}
\]

Here \( r \) is the maximal number of parameters to be updated as a block; we use the value \( r = 4 \). The density \( p(\theta'_J \mid \theta_{M-J}) \) is understood as the conditional prior density \( p(\theta'_J \mid \theta'_{M-J}) \). Recall that aside from the uniform prior we also consider the monotonic prior.

When generating proposals from the monotonic prior, the implied constraints on the penetrances must be taken into account. Let us suppose that \( J \) contains only penetrance parameters. We note that a vector \( \theta'_J \) can be drawn from \( p(\theta'_J \mid \theta'_{M-J}) \) by simply sampling the coordinates \( \theta'_j \) independently from the uniform prior until the monotonicity constraints are met. In practice, this may be much improved by generating each \( \theta'_j \) uniformly from a bounded interval \([l_j, u_j]\), where the lowerbound \( l_j \) and the upperbound \( u_j \) are obtained from the fixed values \( \theta'_{M-J} \). Still, after this modification, not all trial proposals will be successful. However, since the number of parameters to be updated at a time is at most 4, the expected number of trials is low (at most \( 4! = 24 \)). We note that more effective random sampling procedures under partial ordering might exist, but they are not studied here.

Analyzing convergence and mixing properties of the constructed algorithm is difficult. In the experiments, heuristic monitoring of the simulation
output was used. One heuristic is to check that the obtained sample from each chain is symmetric with respect to the allele frequencies, as the model so implies. This applies to two- and three-locus models. Since the symmetry is not exploited in the proposal distributions, observed symmetry would serve as strong evidence for convergence and rapid mixing. A different way to monitor mixing is to check the acceptance rate of swaps between adjacent chains. A high rate, say greater than one percent, for every pair speaks for sufficiently rapid mixing. Namely, when the acceptance rate is high, then it is likely that cooler chains are helped by well-mixing warmer chains.

Finally, a word on computational complexity. The time and space complexity of the implemented MC$^3$ algorithm are obtained in a straightforward fashion. The running time is $O(T k |\mathcal{R}| \alpha(\ell))$, where $\alpha(\ell) = O(\ell 3^\ell)$ bounds the time required for computation of the recurrence risk for a given relationship type. The space complexity is $O(k \beta(\ell))$, where $\beta(\ell) = 3^\ell + \ell - 1$ is the number of parameters in $\ell$-locus model.

### 7.3 Experiments on synthetic and real data

By experiments on a number of recurrence risk data sets we try to answer selected questions about computational techniques and statistical issues. To mention a few: Is the implementation of the method correct? What can we say about the mixing and convergence properties of the implemented MCMC algorithm? How much data is sufficient for learning the correct number of loci, and how crucial is the choice of prior?

Three types of data generation procedures are carried out. In Section 7.3.1, we analyze all possible data sets of a given small size in order to validate the correctness of the implementation and to characterize and illustrate the problem of inferring the number of loci. Then, in Section 7.3.2, we generate a sample of large data sets “from the prior” and estimate the Bayesian power for learning the correct number of loci as well as the Bayesian information gain for certain quantities of interest. In Section 7.3.3, we continue with a study on data sets generated from a selected collection of parameter values. In addition, we analyze real data on diabetes and schizophrenia.

#### 7.3.1 An analysis on small data sets

We consider the collection of all data sets for which $N_R = 4$ for all relationship types $R \in \mathcal{R}$, except for cousins, $C$, for which we put $N_C = 0$. This way we get a collection $\mathcal{D}$ that consists of $5^4 = 625$ data sets. The purpose of these data sets is two-fold: to check the correctness of the software and
Figure 7.1: A summary of an analysis of 625 small data sets under the uniform nonmonotonic prior. (a) A histogram of posterior probabilities for one, two, and three loci. The bins are of width 0.05. (b) The curve of the marginal likelihood for one, two, and three loci. To get decreasing curves, the data sets are sorted separately for each number of loci.

to find out how many of the data sets support one-, two-, and three-locus models (as the most probable explanation for the data). Recall that for each number of loci, \( \ell \), we have

\[
\sum_{D \in D} p(D \mid \ell) = 1, \tag{7.5}
\]

and that the MCMC method should output an estimate of \( p(D \mid \ell) \).

For each number of loci \( \ell = 1, 2, 3 \), two runs were carried out per data set: one under the uniform prior and the other under the monotonic uniform prior. Since the data sizes are very small, we may expect that proper mixing and convergence are rather easy to achieve. So, identical program parameters values were used for every run: 8 coupled chains with the default tempering scheme (7.4) for \( \gamma = 1/2 \), and one million iterations with 10 % burn-in.

Figure 7.1 summarizes the results obtained under the uniform prior. We see that the distribution of the posterior probabilities are somewhat different for different number of loci, as shown in Figure 7.1(a). For example, for one locus the posterior probabilities tend to be more extreme than for two loci. Figure 7.1(b) displays a different characterization. The curves are almost identical for different numbers of loci. Recall that by (7.5),
Figure 7.2: A summary of an analysis of 625 small data sets under the uniform monotonic prior. (a) A histogram of posterior probabilities for one, two, and three loci. The bins are of width 0.05. (b) The curve of the marginal likelihood for one, two, and three loci. To get decreasing curves, the data sets are sorted separately for each number of loci.

for each number of loci the area bounded by the curve should sum up to one. Calculation gives the values 0.9998, 0.9999, and 0.9999, for one, two, three loci, respectively. These values were computed from intermediate results stored with a limited accuracy, which explains the slight deviation from 1. This observation speaks for correctness of the implementation and for efficiency of the estimators. A more surprising result is that for any permutation $\ell_1, \ell_2, \ell_2$ of the numbers 1, 2, 3 there are data sets for which the posterior probabilities satisfy $p(\ell_1 \mid D) \geq p(\ell_2 \mid D) \geq p(\ell_3 \mid D)$. For orderings $(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)$ the numbers of supporting data sets are 311, 33, 3, 1, 19, 258, respectively.

Figure 7.2 summarizes the results obtained under the monotonic uniform prior. The overall results are the same, but the contrasts between the three models are larger. For the ordering the numbers of supporting data sets are 373, 20, 1, 2, 2, 227.

### 7.3.2 A simulation study

Another collection of data sets was generated according to the Bayesian probability model. The sample sizes were fixed as

\[ N_P = 100,000, \quad N_C = 10,000, \quad N_S = 10,000, \quad N_O = 10,000, \quad N_I = 1,000, \]
A collection $\mathcal{D}' = \{D_1, \ldots, D_n\}$ was generated by drawing each $D_i$ independently via the following steps:

1. Draw the number of loci $\ell$ from the prior.
2. Draw the parameters $\theta = (q, f)$ from the prior $p(q, f \mid \ell)$.
3. For each $R \in \mathcal{R}$, calculate the recurrence risk $K_R(\theta)$.
4. For each $R \in \mathcal{R}$, draw $M_R$ from the distribution $p(M_R \mid N_R, \theta)$.

For each data set $D_i$ the generating parameters $\theta_i$ were recorded in order to compare them to the posterior distribution of the parameters. The number of data sets was set to $n = 333$; this many large data sets can be analyzed in a reasonable time.

For each number of loci $\ell = 1, 2, 3$, a single run of MC^3 was carried under the nonmonotonic uniform prior. The monotonic prior was not studied, as drawing from the prior (for three loci) is computationally infeasible (by simple techniques). Since the data sets are now very large, we can expect difficulties in mixing and convergence. Consequently, a relatively large number of parallel chains and a carefully chosen proposal distribution, including the tempering scheme, should be considered. These program parameters should ideally depend on the data (automatically without the human interaction). This would be, however, tedious to implement and would require sophisticated approaches that are beyond this work. Therefore, identical program parameters values were used for every run: 32 coupled chains with the default tempering scheme (7.4) for $\gamma = 1/2$, and 10 million iterations with 10% burn-in.

Figure 7.3 displays the results of the power analysis. We see that the power grows with the data size (Figure 7.3(a)). As expected, when there is little data, the posterior inference of the number of loci is not much better than the prior, uniform guess. Unfortunately, it seems that the power is bounded by a relatively low value, around 0.6. An explanation is suggested by the conditional power for two loci, which is around 0.4 even on large data. That is, when data is generated from the two-locus model, it is hard to infer the correct number of loci from the data. We also see (Figure 7.3(b)) that the posterior probability of the correct number of loci is often around 0.5 and rarely exceeds 0.80.

Estimates for expected information are shown in Figure 7.4. For the number of loci the expected entropy of the posterior distribution decreases as the data size grows (Figure 7.4(a)). On very small data the entropy is close to $-\ln 1/3 \approx 1.1$. We observe similar behavior for the allele frequencies of one-, two-, and three-locus models (Figure 7.4(b)).
Figure 7.3: A summary of a power analysis based on 333 large data sets. (a) Estimates of the Bayesian power for inferring the number of loci as a function of data size. Estimates are also given conditionally on the correct number of loci. (b) A quantile summary of the posterior probability of the correct number of loci.

7.3.3 Selected synthetic and real data sets

In order to get a more detailed view on the analysis of individual data sets, experiments on a small selection of synthetic and real data were carried out. Here the risk types were restricted to the population prevalence, offspring risk, sibling risk, and the identical twin risk.

Eight artificial data sets were generated from six one- and two-locus generating models with specified penetrances and allele frequencies; see Table 7.1. Three one-locus generating models were used: a fully recessive (A), a fully dominant (B), and an incomplete penetrance model (C). The fully penetrant models were also extended to the two-locus case to get a double-recessive model (D) and a double-dominant model (E). In addition, a data set was generated from a two-locus model with incomplete penetrance (F). For each generating model allele frequencies were chosen such that the induced population prevalence is around 2%. For each model a data set was obtained as follows. The sample sizes were set to $N_P = 200,000$, $N_O = N_S = 2,000$, and $N_I = 200$. Then each $M_R$ was assigned its expected value $K_R N_R$ (rounded to the nearest integer). Henceforth, we refer to these data sets by the labels A, B, $\ldots$, F, of the corresponding models. To examine the effect of data size, two additional data sets were generated from the model D. For the data set D1 (low accuracy) the numbers $N_R$ and $M_R$ were reduced to
7.3 Experiments on synthetic and real data

Figure 7.4: A summary of an analysis of expected information based on 333 large data sets. (a) Estimates of the information about the number of loci as a function of data size. Since the number of loci is discrete, we use the Shannon entropy. (b) Estimates of the information about the allele frequencies for different number of loci. Since the allele frequencies are treated as continuous variables, we use an estimate of the differential entropy computed from a posterior sample. These estimates are not conditional on the correct number of loci.

one half, and for the data set D_h (high accuracy) the numbers were doubled.

We use a simplified notation in Table 7.1 and henceforth. The allele frequencies of two loci are denoted by $q_A$ and $q_B$. Similarly $f_{AA}BB$ refers to the penetrance of a two-locus genotype with alleles $A$ and $a$ at the first locus and two $B$'s at the second locus. A penetrance vector for one locus is organized as $\{ f_{AA}, f_{Aa}, f_{aa} \}$. For two loci the columns of a penetrance matrix correspond to penetrance vectors for genotypes $BB,Bb,$ and $bb$ in this order.

Type I diabetes and schizophrenia served as examples of real traits. For these disorders, the empirical recurrence risks for most common relationship types can be found in literature; numbers for diabetes (P. Oukam, personal communication) and schizophrenia [Ris90] are shown in Table 7.1. Unfortunately, the underlying sample sizes were not available. To circumvent this problem, the numbers $N_R$ were set as for the synthetic data sets, and the numbers $M_R$ were generated according to the empirical risk estimates.

All these data sets were analyzed using the same program parameters. For each number of loci $\ell = 1, 2, 3$, a single run of $MC^3$ was carried under
Table 7.1: Synthetic and Observed Recurrence Risks

<table>
<thead>
<tr>
<th>Name</th>
<th>Penetrances</th>
<th>Frequencies</th>
<th>Risks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(q_A)</td>
<td>(q_B)</td>
</tr>
<tr>
<td>Synthetic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>{1 0 0}</td>
<td>.14</td>
<td>—</td>
</tr>
<tr>
<td>B</td>
<td>{1 1 0}</td>
<td>.01</td>
<td>—</td>
</tr>
<tr>
<td>C</td>
<td>{.75 .25 .01}</td>
<td>.02</td>
<td>—</td>
</tr>
<tr>
<td>D, Dl, Dh</td>
<td>{0 0 0} {0 0 0} {1 1 0}</td>
<td>.38</td>
<td>.38</td>
</tr>
<tr>
<td>E</td>
<td>{1 1 0} {0 0 0} {1 .5 0}</td>
<td>.07</td>
<td>.07</td>
</tr>
<tr>
<td>F</td>
<td>{.5 0 0} {0 0 0}</td>
<td>.22</td>
<td>.22</td>
</tr>
<tr>
<td>Real</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DIA</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>SCH</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

the monotonic uniform prior. The number of coupled chains was set to 8 with the default tempering scheme (7.4) for \(\gamma = 1/2\). Posterior samples were collected along 100 million iterations with 10% burn-in. Summaries of the analyses are shown in tables 7.2, 7.3, 7.4 and Figure 7.5.

**Synthetic data.** Posterior probability estimates for the number of loci are shown in Table 7.2. For the one-locus recessive and dominant data sets the correct number of loci has a probability close to 1. For the data set C the posterior is more vague. The results for D, Dl, and Dh show the effect of data size. Given the smallest data set, one-locus is clearly preferred. But when the data size is moderate or large, the correct number of loci is assigned the highest posterior probability. This rather rapid change can be understood as follows. Only a tiny subspace of two-locus parameters fits the data set D better than the best one-locus parameters. This difference becomes visible in posterior probabilities after additional data has enlarged the difference in the goodness of fit. For the data sets E and F, the chosen data size is already sufficient to support the correct two-locus model.

Table 7.3 summarizes the posterior estimates for the parameters of the one-locus model on the one-locus data sets. The mean estimates are accurate for the data sets A and B; they are in good agreement with the corre-
Table 7.2: Estimates of the Posterior Distribution of the Number of Loci

<table>
<thead>
<tr>
<th>DATA SET</th>
<th>ONE LOCUS</th>
<th>TWO LOC</th>
<th>THREE LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.000</td>
<td>.0000</td>
<td>.0000</td>
</tr>
<tr>
<td>B</td>
<td>1.000</td>
<td>.0000</td>
<td>.0000</td>
</tr>
<tr>
<td>C</td>
<td>.67</td>
<td>.24</td>
<td>.09</td>
</tr>
<tr>
<td>D</td>
<td>.0075</td>
<td>.9925</td>
<td>.0000</td>
</tr>
<tr>
<td>DI</td>
<td>.9991</td>
<td>.0009</td>
<td>.0000</td>
</tr>
<tr>
<td>Dh</td>
<td>.0000</td>
<td>.9999</td>
<td>.0001</td>
</tr>
<tr>
<td>E</td>
<td>.0000</td>
<td>1.000</td>
<td>.0000</td>
</tr>
<tr>
<td>F</td>
<td>.002</td>
<td>.993</td>
<td>.005</td>
</tr>
<tr>
<td>DIA</td>
<td>.839</td>
<td>.157</td>
<td>.004</td>
</tr>
<tr>
<td>SCH</td>
<td>.000</td>
<td>.888</td>
<td>.112</td>
</tr>
</tbody>
</table>

Table 7.3: Posterior Summary of the One-Locus Model on One-Locus Data

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>DATA A MEAN ± S.D.</th>
<th>DATA B MEAN ± S.D.</th>
<th>DATA C MEAN ± S.D.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_A$</td>
<td>.140 1.21e-3</td>
<td>.0100 1.76e-4</td>
<td>.0333  .0230</td>
</tr>
<tr>
<td>$f_{AA}$</td>
<td>.995 4.43e-3</td>
<td>.996 3.58e-3</td>
<td>.582   .223</td>
</tr>
<tr>
<td>$f_{Aa}$</td>
<td>4.78e-4 3.83e-4</td>
<td>.992 4.99e-3</td>
<td>.249   .137</td>
</tr>
<tr>
<td>$f_{aa}$</td>
<td>9.22e-5 9.19e-5</td>
<td>7.85e-5 7.60e-5</td>
<td>.00818 .00429</td>
</tr>
</tbody>
</table>

responding generating models (see Table 7.1). For the data set C, in contrast, large standard deviations suggest that the mean estimates cannot capture relevant features of the posterior distribution. Recall that, in general, four parameters are too many for complete identifiability [Jam71, FRSJ78].

Figure 7.5 shows estimated posterior distributions for the allele frequencies of the two-locus model, on different data sets. Since the two-locus model is greatly over-parametrized, one might not expect to see clear patterns in the posterior. As a little surprise, for the fully recessive data set A, the MCMC sample only supports hypotheses where the allele frequency is around .14 for one locus and close to 0 for the other. A similar phenomenon appears with the data set B. In this case the marginal posterior distribution of the allele frequencies can be characterized as the relationship $q_A + q_B \approx .01$. We note that hypotheses in which the genetic effect is shared by the two loci are also supported by the data set, though the points on the corresponding line segment are not clearly shown in Figure 7.5. On the data set C penetrances are not forced to values close to 0 or 1 due to
Table 7.4: A Selection of Two-locus Models, Indicated in Figure 7.5

<table>
<thead>
<tr>
<th>Model</th>
<th>Penetrance</th>
<th>Frequencies</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$q_A$</td>
</tr>
<tr>
<td>d1</td>
<td>.993</td>
<td>.005</td>
</tr>
<tr>
<td></td>
<td>.001</td>
<td>.001</td>
</tr>
<tr>
<td>d2</td>
<td>.999</td>
<td>.020</td>
</tr>
<tr>
<td></td>
<td>.990</td>
<td>.003</td>
</tr>
<tr>
<td></td>
<td>.006</td>
<td>.001</td>
</tr>
<tr>
<td>d3</td>
<td>.984</td>
<td>.978</td>
</tr>
<tr>
<td></td>
<td>.975</td>
<td>.001</td>
</tr>
<tr>
<td></td>
<td>.002</td>
<td>.001</td>
</tr>
<tr>
<td>e1</td>
<td>.997</td>
<td>.987</td>
</tr>
<tr>
<td></td>
<td>.987</td>
<td>.987</td>
</tr>
<tr>
<td></td>
<td>.009</td>
<td>.001</td>
</tr>
<tr>
<td>e2</td>
<td>.999</td>
<td>.746</td>
</tr>
<tr>
<td></td>
<td>.990</td>
<td>.008</td>
</tr>
<tr>
<td></td>
<td>.000</td>
<td>.000</td>
</tr>
<tr>
<td>sch1</td>
<td>.648</td>
<td>.561</td>
</tr>
<tr>
<td></td>
<td>.340</td>
<td>.028</td>
</tr>
<tr>
<td></td>
<td>.008</td>
<td>.005</td>
</tr>
<tr>
<td>sch2</td>
<td>.926</td>
<td>.681</td>
</tr>
<tr>
<td></td>
<td>.736</td>
<td>.638</td>
</tr>
<tr>
<td></td>
<td>.126</td>
<td>.007</td>
</tr>
</tbody>
</table>

the reduced concordance in identical twins. We obtain a quite flat posterior distribution. A wide range of allele frequencies are reasonable in light of the data set. However, low values are more probable, for low allele frequencies allow more variation in the penetrances, and thus gather more probability mass.

For the double-recessive data set (D) the posterior of the allele frequencies displays a complex structure (Figure 7.5). Different dependency curves correspond to different patterns of $0/1$-penetrances, as displayed in Table 7.4. For the more accurate data set (Dh) the posterior mass is concentrated on the uppermost curve, corresponding to the penetrances of the
7.4 Conclusions

...generating model but letting allele frequencies vary. When the data size is smaller (D1), allele frequencies take values more independently. For the double-dominant data set (E) the sample consists of three distinct high-density areas. The middle one corresponds to the penetrances of the generating model; for the other two curves the penetrance \( f_{AaBb} \) is flipped from near 1 to almost 0 (see Table 7.4). For the data set F the posterior distribution of the allele frequencies is flatter (results not shown). This is not a surprise, as for this data set, well-fitting models are not forced to 0/1-penetrances.

**Diabetes and schizophrenia.** For diabetes the data supports one locus, and for schizophrenia the two-locus model is preferred (Table 7.2). However, on these data sets the differences are relatively small.

On DIA the mean estimates of the one-locus parameters \( q_A, f_{AA}, f_{Aa}, \) and \( f_{aa} \) were obtained as \(.044 \pm .009, .869 \pm .111, .039 \pm .009, \) and \(2.6e^{-4} \pm 2.1e^{-4} \), respectively. From the relatively small standard deviations we conclude that the mean estimates characterize the high-density region of the posterior rather well. The joint distribution of the two-locus allele frequencies seems to have a high density region around the values \( q_A = q_B = .05 \) (Figure 7.5).

On SCH we obtained two-locus models with allele frequency distribution similar to that obtained with the data set F. Two example hypotheses, sch1 and sch2, are given in Table 7.4. The penetrance patterns of sch3 and the generating model F are quite similar (see Table 7.1). However, being affected requires neither double \( AA \) nor \( BB \), for the penetrances \( f_{AaBb}, f_{Aabb}, \) and \( f_{aaBb} \) are nonzero, though low. In sch2 the penetrance is high among the genotypes with at least two disease alleles, but a calculation reveals that most of the affected individuals are expected to carry a single \( A \) or \( B \).

7.4 Conclusions

The experiments provide evidence for the correctness of the implementation and for the efficiency of the method. First, the sum of the marginal likelihoods over small data sets yielded the expected result. Second, the results for selected synthetic data set showed that extreme boundaries will be successfully explored by the MC^3 algorithm. The symmetry of the two-locus allele frequency distributions speaks for relatively rapid mixing and convergence. However, the method is highly computer-intensive—an analysis on a single data set may take several hours depending on the number of iter-
ations and the number of coupled chains. In these experiments relatively large values were used.

Statistically, the inversion from risk data to genetic hypotheses seems to be a complex task. The fact that the model parameters are not identifiable makes the analysis challenging. Although the posterior distribution is a valid representation of updated knowledge, it may be problematic to summarize.

We focused on the inference of the number of loci. Results on small data sets are interesting, as they declare that the two-locus hypothesis rarely gets the highest posterior probability. One might first think that one locus fits best to all reasonable data sets that satisfy the inequalities of recurrence risks (Theorem 6.57 in Chapter 6), whereas the more complex three-locus hypothesis is more plausible on improbable data. But this idea is flawed. Namely, a closer look at the results reveals that, for example, on the data \((M_P, M_O, M_S, M_I) = (0, 1, 2, 3)\) the posterior order of the number loci is \((3, 2, 1)\), and on \((M_P, M_O, M_S, M_I) = (3, 2, 1, 0)\) the order is the reverse one, \((1, 2, 3)\). A somewhat surprising result is also that the distributions of the marginal likelihood are identical for different number of loci.

The analysis of Bayesian power showed a limit on the inference of the number of loci. Even in the case of a very large data set, the (prior) probability of inferring the correct number of loci is around 0.6. An open issue is how much hard or soft constraints on the prior would help in this. Some hints are, however, provided by the results on the special synthetic data sets. We saw that when the penetrance is complete or close to it, then the posterior probability mass concentrates on the correct number of loci.

We also studied posterior patterns of allele frequencies and penetrances. Inferring these parameters from data is seldom possible. Due to unidentifiability a continuum of parameter values may give equal fit to given data. The simulation study on large data sets reflected this fact. Namely, the results suggest that the expected (differential) entropy will not tend to \(-\infty\) as the data size grows, but that is has a finite lower bound. This would mean that the expected information has an upper bound. Yet, when the penetrance in the generating model was complete (A, B, D, E), we observed clear patterns. In the case of one locus (A and B), the four parameters were identifiable. In the case of two loci (D and E), the posterior mass concentrated on a couple of penetrance configurations and a few “dependence curves” of allele frequencies.

The results on diabetes and schizophrenia data are not genetically very informative. The diabetes data set provided mild evidence for the one-locus hypothesis, but note that no background knowledge of this trait was
incorporated into the prior. Indeed, the current knowledge is that there are several susceptibility loci (see, e.g., [Onk02] and reference therein). For schizophrenia the two-locus hypothesis is most plausible in light of the data. Yet, we notice that the recurrence risk estimates do not satisfy the inequalities of recurrence risks (Theorem 6.57 in Chapter 6), as the sibling risk is lower than the offspring risk. This suggests that the epistatic Mendelian model may not be appropriate in this case. More probably, the observed distortion may be due to ascertainment bias in the data collection procedure.

In summary, the presented results show that Bayesian analysis of risk data is computationally feasible and may provide useful information. However, the amount of information seems to depend on how complete the penetrance of the trait is, i.e., the concordance in identical twins. Further experiments are needed to characterize the role of the data size in these cases. This might be best carried out by analyzing Bayesian power and expected information gain under more informative priors.
Figure 7.5: A scatter plot of the frequencies $q_A$ (horizontal axis) and $q_B$ (vertical axis) of two-locus models for one- and two-locus data sets, and for DIA and SCH. Parameter values of the models that are indicated by arrows are shown in Table 7.4.
Chapter 8

Discussion

In this thesis we briefly discussed some fundamental issues of Bayesian modeling, reviewed the general sum-product framework, and described the tempered Monte Carlo method. Finally, these general methods were applied to the analysis of recurrence risks.

The contributions of the thesis include a unifying view on transformation problems, and specially, on the Yates algorithm and the fast Möbius transform. We also showed that the tempered integration method, in conjunction with the Metropolis-coupled Markov chain Monte Carlo algorithm, is well suited to integration tasks that typically appear in Bayesian statistics. Most of this generic development found use in the genetic application. We also introduced the notion of Bayesian power.

The main messages of this work emerge from the interactions of generic and application-specific issues. The primary message concerns computational problems. The sum-product framework and the variable elimination algorithm are useful building blocks in many computational problems. Yet, there may exist different problem representations that lead to equally fast algorithms. Which one is the fastest may not be clear, when the efficiency is measured as a function of two or more problem parameters. The recurrence risk computation is a good example of such a problem.

A secondary message concerns reasoning under uncertainty. One of the most appealing features of the Bayesian paradigm is that the model can be arbitrarily complex. However, one ought to be careful, as the power of the model depends on the strength of prior beliefs. We demonstrated the computational issues related to these features in the genetic risk analysis application, where unidentifiability is present.

The analysis of genetic risk data can be viewed as an example problem. Typically, risk data are alone not sufficient to identify the properties of the trait; rather, it should be combined with other types of data. In this work,
our purpose was not to investigate what type of data one should collect, but to study how a given risk data set is best analyzed. From the methodological point of view, the genetic data analysis task served as a large case study.

More generally, the prospects of the Bayesian machinery, armed with efficient summation algorithms, seem excellent regarding scientific data analysis. The rapid growth of computing power has made it possible to handle complex models, multivariate systems where the data analyst is forced to consider marginal summaries, instead of point hypotheses. Sound marginalization can only become more important in the future. A good example of such a problem domain is structure discovery in Bayesian networks. Promising general methods that combine exact and approximate summation have been proposed recently in [FK03]. However, it seems that a great deal of interesting algorithmic work still remains to be done in this context.

We finish by recalling some open research problems we had to pass by in the present work. Concerning the Bayesian paradigm, we sketched, in Section 2.3, an approximation approach to manage the annoying problem of wrong models; it would be interesting to develop this further. The relationship of randomized approximation algorithms and Bayesian approaches—briefly mentioned in Sections 2.4 and 4.2—is also interesting from the point of view of computational complexity. Regarding sum-product algorithms on rings, we observed that the ordinary variable elimination algorithm can be accelerated by fast rectangular matrix multiplication. An open problem is whether similar algebraic techniques provide us with even faster algorithms in special cases, e.g., in those stated in Section 3.6. An interesting question is also whether the sum-product framework and the variable elimination algorithm can be extended to automatically manage different problem representations (cf. the convolution problem and the fast Fourier transform). This might offer also an answer to the open question whether the presented two algorithms for the computation of recurrence risks can be derived from some unifying problem representation, and whether they are optimal. Finally, there are numerous directions in which the genetic model considered in Chapters 5, 6, and 7 can be extended. These include, e.g., allowing for linkage between loci and constructing the prior of penetrance as a mixture of simple and complex distributions. Of course, such extensions also lead to new algorithmic issues.
References


REFERENCES


Reports may be ordered from: Department of Computer Science, Library (A 214), P.O. Box 26, FIN-00014 University of Helsinki, FINLAND.


A-2002-3  J. Vilo: Discovery from biosequences. 149 pp. (Ph.D. thesis)