Tel Aviv University
Sackler Faculty of Exact Sciences
School of Computer Science

Computational Problems in Modern Human Genetics

THESIS SUBMITTED FOR THE DEGREE OF
"DOCTOR PHYLOSOPHY"

by

Gad Kimmel

The work on this thesis was carried out
under the supervision of **Prof. Ron Shamir**

Submitted to the Senate of Tel Aviv University
September 2006

Acknowledgments

First and foremost, I want to thank my adviser, Prof. Ron Shamir, for his support, guidance, encouragement, and constructive criticism throughout the course of this work. I am indebted to him for giving me the opportunity to enter into the exciting world of computer science and computational biology. I have learned from him at all levels, and I hope to carry with me his values of scientific integrity and persistence.

I want to thank Hilla, my best friend and wife, for walking beside me all these years, and for her support and encouragement whenever needed. Next, I would like to thank my parents - Sara and Jacob Kimmel, for their love and for the hope they gave me, especially in rainy days. Also, I want to thank my brother, Prof. Ron Kimmel, who assisted me significantly in choosing the right path, when being in professional decision junctions.

I want to thank Prof. Isaac Meilijson for many helpful discussions and for broadening my horizons in probability and statistics. Additionally, I want to thank Irit Gat-Viks, Amos Tanay and Ofir Davidovich, for all our fruitful dialogs.

Last, but not least, I would like to thank all my collaborators: Dr. Roded Sharan (Tel Aviv University); Dr. Eran Halperin (International Computer Science Institute, Berkeley); Prof. Eitan Friedman, Inbar Gal and Marie Koren (Sheba Medical Center, Tel-Hashomer); Dr. Arie Levine and Dr. Esther Leshinsky-Silver (Wolfson Medical Center); Dr. Amir Karban (Rambam Medical Center); Prof. Jacqui Beckmann (Lausanne); Prof. Doron Lancet and Dr. Edna Ben-Asher (Weizmann Institute of Science); and Prof. Margret Hoehe (Max Plank Institute).

Preface

This thesis is based on the following collection of seven articles that were published throughout the PhD period in scientific journals and in reviewed proceedings of conferences.

1. **Computational problems in noisy SNP and haplotype analysis: block scores, block identification and population stratification.**
   Gad Kimmel, Roded Sharan and Ron Shamir.
   Published in *Proceedings of the Workshop on Algorithms in Bioinformatics (WABI 03)* [27] and in *INFORMS Journal on Computing* [28].

2. **The incomplete perfect phylogeny haplotype problem.**
   Gad Kimmel and Ron Shamir.
   Published in *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes* [21] and in *Journal of Bioinformatics and Computational Biology (JBCB)* [25].

3. **Maximum likelihood resolution of multi-block genotypes.**
   Gad Kimmel and Ron Shamir.
   Published in *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 04)* [22].

4. **GERBIL: genotype resolution and block identification using likelihood.**
   Gad Kimmel and Ron Shamir.
   Published in *Proceedings of the National Academy of Sciences of the United States of America (PNAS)* [24].

5. **A block-free hidden Markov model for genotypes and its application to disease association.**

Gad Kimmel and Ron Shamir.
Published in *Journal of Computational Biology* [23].

6. **Tag SNP selection in genotype data for maximizing SNP prediction accuracy.**
   Eran Halperin*, Gad Kimmel* and Ron Shamir (* equal contribution).
   Published in *Bioinformatics* journal supplement for the proceedings of *The 13th Annual International Conference on Intelligent Systems for Molecular Biology (ISMB 2005)* [16].

7. **A fast method for computing high significance disease association in large population-based studies.**
   Gad Kimmel and Ron Shamir.
   An invited oral presentation in *The Biology of Genomes* meeting, Cold Spring Harbor Laboratory, 2006. Published in *American Journal of Human Genetics* [26].

Abstract

Most of genetic variation among human individuals is due to single nucleotide polymorphisms (SNPs). The knowledge of genome variation is expected to play a key role in disease association studies. Hence, the identification and analysis of SNPs is currently a major goal of the international scientific community. In this thesis, we studied several of the major computational problems that arise in the analysis of SNP data. We used computational techniques from graph theory, probability and statistical theory and integrated them with biological principles to develop models for these problems. We used our methods to study extensive human SNP data.

We first studied haplotype block partitioning. Given a set of haplotypes, our objective was to find a block partitioning minimizing the total number of distinct haplotypes in blocks. We showed that the problem is NP-hard when there are errors or missing data, and provided approximation algorithms for several problem variants. On real data, we generated a more concise block description than previous approaches.

The next step was to study the phasing problem. Initially, we analyzed the perfect phylogeny haplotype problem. We proved that this problem is NP-complete when some of the data entries are missing. We developed an algorithm that takes an expected polynomial time, under a reasonable probabilistic model for genotype generation. In tests on simulated data, our algorithm quickly resolved the genotypes under high rates of missing entries.

To obtain more accurate phasing, we developed a new algorithm that performs genotype phasing and block partitioning in one process. We defined a stochastic model for blocks of recombination-poor regions, in which haplotypes are generated from a small number of core haplotypes, allowing for mutations, rare recombinations and errors. We developed an EM method for that model,

which outperformed most of the phasing algorithms when published. Moreover, our algorithm could handle very large datasets with many hundreds of genotypes, while the time required by other accurate methods to handle such data sets was prohibitive. In a subsequent work, we developed an improved model, which offers a compromise between rigid block structure and no structure altogether: It reflects a general blocky structure of haplotypes, but also allows for "exchange" of haplotypes at non-boundary SNP sites; it also accommodates rare haplotypes and mutations. We tested the model on many different data sets of genotypes. In comparison to three other models, its accuracy was the highest.

Next, we studied the tag SNP selection problem. We defined a new natural measure for evaluating the prediction accuracy of a set of tag SNPs, and used it to develop a new method for tag SNPs selection. We compared our method to two state of the art tag SNP selection algorithms on a large number of different genotype data sets. Our method consistently found tag SNPs with considerably better prediction ability than the other methods.

In our last study we developed a faster algorithm for calculating accurate p-values in case-control association studies. We developed a method based on importance sampling and exploiting the decay in linkage disequilibrium along the chromosome. Our algorithm is 3-5 orders of magnitude faster than the standard permutation test, which is used for evaluating the significance, while preserving the same accuracy and robustness. The method significantly increases the problem size range for which accurate, meaningful association results are attainable.

Contents

Chapter 1

Introduction

1.1 General Background

The availability of a nearly complete human genome sequence makes it possible to look for telltale differences between DNA sequences of different individuals on a genome-wide scale, and to associate genetic variation with medical conditions. In order to achieve this goal, researchers concentrate on positions along the DNA sequence, which show variability in their nucleic acid contents across the population. Such sites are called *single nucleotide polymorphisms* (SNPs). Millions of SNPs have already been detected [47, 56], out of an estimated total of 10 millions common SNPs. The sequence of alleles in contiguous SNP positions along a chromosomal region is called a *haplotype*. The conflated information of two haplotypes obtained from a person is called a *genotype*.

Common diseases such as cancer, stroke, heart disease, diabetes and asthma usually result from the combined effects of a number of genetic variants and environmental factors. Several preliminary studies [36, 38, 46] have demonstrated that the risk of contracting common diseases is influenced by genetic variants that are relatively common in populations. Not enough data are yet available to evaluate the generality of this hypothesis, but more and more widely distributed genetic variants associated with common diseases are being discovered. Therefore, the identification and analysis of SNPs and haplotypes is currently a major effort of the international community [60].

1.2 Blocks of High Linkage Disequilibrium

SNPs that are in close physical proximity to each other are often correlated. This correlation is measured by the *linkage disequilibrium* (LD) between the SNPs [44]. There are several different methods to evaluate the LD. If A, a are two possible alleles of one locus, B, b are two possible alleles of another locus, and $p(\cdot)$ is the probability, then the LD is usually represented by D, D' and r^2:

$$D = p(AB) - p(A)p(B),$$

$$D' = D/D_{max},$$

where

$$D_{max} = \begin{cases} \min\{p(A)p(b), p(a)p(B)\} & D \geq 0 \\ \max\{-p(A)p(B), -p(a)p(b)\} & D < 0 \end{cases},$$

and

$$r^2 = \frac{D^2}{p(A)p(a)p(B)p(b)}.$$

LD tends to decay with distance, so that a lower LD value is usually observed between loci that are farther apart. This is mainly explained by the fact that the probability for a recombination event along the genealogy of the population is larger when the distance between the SNPs is larger.

A major recent discovery is that haplotypes tend to be preserved along relatively long genomic stretches, with recombination occurring primarily in narrow regions called *hot spots* [10, 42]. The regions between two neighboring hot spots are called *blocks*, and the number of distinct haplotypes within each block that are observed in a population is very limited: typically, some 70-90% of the haplotypes within a block belong to very few (2-5) *common haplotypes* [42]. The remaining haplotypes are called *rare haplotypes*. While the block model is only an approximation of biological reality, this finding is very important to disease association studies, since once the blocks and common haplotypes are identified, one can hopefully obtain a much stronger association between a haplotype and a disease phenotype.

Several studies have concentrated on the problem of block identification in a given collection of haplotypes: Zhang et al. [63, 65] sought a block partitioning that minimizes the number of tag SNPs (roughly speaking, this is a set of

sites with the property that the combination of alleles in it uniquely identifies the alleles at all other sites). Koivisto et al. [30] used a minimum description length (MDL) criterion for block definition. All these studies used the same basic dynamic programming approach of [63] to the problem, but differed in the optimization criterion used within the dynamic programming computation.

1.3 Genotypes and Phasing

The block partitioning problem is intertwined with another problem in diploid organisms. Such organisms (including humans) have two near-identical copies of each chromosome. Most techniques for determining SNPs do not provide the haplotype information separately for each of the two copies. Instead, they generate for each site *genotype* information, i.e., an unordered pair of allele readings, one from each copy [47].

Hence, given the genotype data {A,A} {A,C} {C,G} for three SNP sites in a certain individual, there are two possible haplotype pair solutions: (ACC and AAG), or (ACG and AAC). A genotype with two identical bases in a site is called *homozygous* in that site, while if it has two different bases it is called *heterozygous* in that site. The genotype in the example above is homozygous for the allele A in the first site, and heterozygous in the second and third sites. The process of inferring the haplotypes from the genotypes is called *phasing* or *resolving*.

In the absence of additional information, each genotype with h heterozygous sites can be resolved in 2^{h-1} different ways. Resolving is done simultaneously in all the available genotypes and is based on some assumptions on how the haplotypes were generated. The first approach to haplotype resolution was Clark's parsimony-based algorithm [4]. Likelihood-based Expectation - Maximization (EM) algorithms [9, 34] gave better results. Stephens et al. [54] and Niu et al. [40] proposed MCMC-based methods which gave promising results. All of those methods assumed that the genotype data correspond to a single block with no recombination events. Hence, for multi-block data the block structure must be determined separately.

A novel combinatorial model was suggested by Gusfield [14]. According to this model, the resolution must produce haplotypes that define a *perfect phylogeny tree*. Gusfield provided an efficient yet complex algorithm for the problem. Simpler, direct efficient algorithms under this model were later developed [8, 1]. Eskin

et al. [8] showed good performance with low error rates on real genotypes.

In real genotype data (e.g., refs [42, 10, 7]) some of the data entries are often missing, due to technical causes. Current phasing algorithms that are based on perfect phylogeny require complete genotypes. This situation raises the following algorithmic problem: Complete the missing entries in the genotypes and then resolve the data, such that the resulting haplotypes define a perfect phylogeny tree. We call this problem *incomplete perfect phylogeny haplotype* (IPPH). It was posed by Halldòrsson et al. [15]. In order to deal with such incomplete data, Eskin et al.[8] used a heuristic to complete the missing entries, and showed very good results. However, having an algorithm for optimally handling missing data entries should allow more accurate resolution.

The IPPH problem has two variants: *rooted* (or *directed*) and *unrooted* (or general). In the rooted version, one haplotype is given as part of the input. This haplotype is referred to as the root of the tree, even though it may not be the real evolutionary root of the tree. This holds, since each of the haplotypes can be used as a root in the perfect phylogeny tree [13]. The unrooted version is a more faithful formulation of the practice in biology, since in phasing, the root of the haplotypes is not given.

While elegant and powerful, the perfect phylogeny approach has certain limitations: first, it assumes that the input data admit a perfect phylogeny tree. This assumption is often violated in practice, due to data errors and rare haplotypes. In fact, Eskin at al. show that in the real data that they analyzed, a block does not necessarily admit a perfect phylogeny tree. Second, the model requires partition of data into blocks by other methods. Third, the solution to the problem may not be unique and there may be several (or many) indistinguishable solutions. (These limitations were addressed heuristically in [8]).

Another common approach for phasing is to define a stochastic generation model for the haplotypes, and to resolve its parameters. Many of the studies that use this approach [9, 34, 54, 40] make the Hardy-Weinberg assumption [17] that mating is random. In a specific block, for a population of size n, the likelihood function is:

$$L = \prod_{i=1}^{n} \alpha_1^i \alpha_2^i,$$

where α_j^i is the probability of the j-th haplotype of the i-th individual in the population (j equals 1 or 2). However, since the haplotypes of an individual are

unknown, the equation becomes:

$$L = \prod_{i=1}^{n} \sum_{h_1+h_2=g_i} \alpha_{h_1}\alpha_{h_2},$$

where α_x is the probability of haplotype x in the population, g_i is the genotype of the i-th person, and "$h_1 + h_2 = g_i$" denotes the set of all pairs of haplotypes possible to generate genotype g_i. Note that the above equation applies for a specific block.

Expanding this idea, several authors suggested methods of performing phasing and block partitioning as one process. Greenspan and Geiger [12] proposed a new method and algorithm, called *HaploBlock*, which performs resolution while taking into account the block structure. The method is based on a Bayesian network model. Stephens et al. [52] presented the *PHASE* algorithm, in which the phasing process is performed under the assumption of the coalescent model [29] as a prior in short segments. Very good results were reported.

1.4 Tag SNPs

The total cost of a study grows with the number of SNPs typed. Therefore, to save resources, one wishes to reduce the number of SNPs typed per individual. This is usually done by choosing an appropriate small subset of the SNPs, called *tag SNPs*, that could predict the rest of the SNPs with a small error. Thus, when preforming a disease association study, the geneticist would experimentally test for association by only typing the tag SNPs, thereby considerably saving resources (or, alternatively, increasing the power of the statistical tests by increasing the number of individuals). Hence, a key problem is to find a set of tag SNPs of minimum size that would have a very good prediction ability.

Finding a high-quality set of tag SNPs is a challenging task for several reasons. One of the main challenges is that the haplotype information is usually not given, and one has information on genotypes only. As mentioned in Section 1.3 there are computational tools that use the correlations between neighboring SNPs in order to predict the phase information. Their accuracy depends on the proximity and correlation of the SNPs typed. When a set of tag SNPs is chosen and then typed, the rest of the SNPs are not measured and instead must be predicted from this information. The accuracy of such prediction is limited, since the correlation

between the tag SNPs is not necessarily as strong as the correlation between SNPs that are in close proximity to each other. Most of the extant methods for tagging SNPs, that aim to explicitly predict individual SNPs, use the haplotypes of the tag SNPs.

Another issue that is crucial in the search for tag SNPs is the definition of an adequate measure of the prediction quality. Many of the current tag SNP selection methods partition the region into blocks of limited diversity (e.g., [63, 65, 64]), and find a set of tag SNPs that aims to predict the common haplotypes of each block. The most apparent disadvantages of such an approach are the lack of cross-block information and the dependency of the tag SNPs choice on the block definition.

1.5 Association Studies

Linking genetic variation and personal health is one of the major challenges and opportunities facing scientists today. It was listed as one of the "125 big questions that face scientific inquiry over the next quarter-century" [5]. The accumulating information on this variation is making large scale, genome-wide disease association studies possible for the first time. Preliminary studies have shown that the knowledge on genome variation is expected to play a key role in disease association studies [36, 38, 46]. The objective of such studies is to find genetic factors correlated with complex disease. In these studies, the DNA of individuals from two populations, healthy individuals and carriers of the disease, is sampled. When differences between the SNP and haplotype structures of the two populations are revealed, they may lead to identifying the genetic source of the disease.

The next few years carry the promise of very large association studies that will use SNPs extensively [49]. There are already reported studies with 400-800 genotypes [39], and studies with thousands of genotypes are envisioned [39]. High throughput genotyping methods are progressing rapidly [55]. The number of SNPs typed is also likely to increase with technology improvements: DNA chips with 500,000 SNPs are already commercially available [58]. Hence, it is essential to develop computational methods to handle such large data sets. Our focus in this part of the thesis is on improving a key aspect in the mathematical analysis of disease association studies.

There are two major types of association studies: *family-based*, in which the control individuals are the parents of the affected individuals, and *case-control* or *population-based*, in which the study samples are collected from unrelated individuals from a population.

The most widely used test for the family-based case is the transmission disequilibrium test (TDT), introduced by Spielman et al. [51]. This test has been extended by Spielman et al. [50] to multi-allelic markers. The test statistic is derived from the probability of transmitting alleles from the parents to the affected child. This statistic is χ^2 distributed.

In our research, we focused on population-based association studies, in which unrelated individual are genotyped. The test for association in this case is usually based on the difference in allele frequency between case and control individuals. For a single SNP, a common test, suggested by Olsen et al. [41], is based on building a contingency table of alleles vs. disease phenotypes (i.e., case - control), and then calculating a χ^2 distributed statistic. When multiple markers in a chromosomal region are tested, several studies suggested the use of generalized linear models [61, 48, 33]. Such methods must assume a specific distribution of the trait given the SNPs, and this assumption does not always hold. Typically, a Bonferroni correction for the p-value is employed to account for multiple testing. However, this correction does not take into account the dependence of strongly linked maker loci, and may lead to over-conservative conclusions. This problem worsens when the number of tests grows, due to a larger number of sites.

To cope with these difficulties, Zhang et al. [62] suggested a Monte Carlo procedure to evaluate the overall p-value of the association between the SNPs data and the disease: The χ^2 value of each marker is calculated, and the maximum value over all markers, denoted $\mathrm{CC}_{\max}$, is chosen as the test statistic. Then, the same statistic is calculated for many data sets with the same genotypes and randomly permuted labels of the case and control individuals. The fraction of times that this value exceeds the original $\mathrm{CC}_{\max}$ is used as the p-value. A clear advantage of this test is that no specific distribution function is assumed, and a random model is simply generated by permutations of the case / control labels. Additionally, the test handles multiple-testing directly and avoids the bias of correction. Consequently, it is widely used, and, for instance, is implemented in the state of the art software package Haploview [59] developed by the HapMap project.

The permutation test can be readily generalized to handle association between haplotypes and the disease, e.g., by adding artificial loci for block haplotypes [7, 24] with states corresponding to common haplotypes. Similarly, one can add loci interactions as artificial loci, whose states are the allele combinations.

Running time is a major obstacle in preforming permutation tests. The time complexity of the algorithm is $O(N_S nm)$, where N_S is the number of permutations, n is the number of samples, and m is the number of loci. To search for p-values as low as α, about $1/\alpha$ permutations are needed. Hence, in an association study that contains 1,000 cases and 1,000 controls, with 10,000 loci, to reach a p-value of 10^{-6}, over 10^{13} basic computer operations are required, or over 30 CPU days using a standard computer. Scaling up to larger studies with 100,000 loci or more is completely out of reach.

When SNP interactions are also considered, time complexity is an even greater concern. There are several possible biological configurations by which two or more loci can interact. Several statistical studies focus on modeling loci interactions that have little or no marginal effects at each locus [18, 6, 37]. Recently, Marchini et al. [35] addressed the question of designing association studies, given the plausibility of interactions between genetic loci with non-negligible marginal effects. In all of these studies the multiple-testing cost of fitting interaction models is much larger than that for the single-locus analysis. Moreover, the dependency among different tests is higher, so the disadvantage of the conservative Bonferroni correction is exacerbated. For example, when testing all possible pairwise loci interactions, a quadratic number of tests has to be applied, and applying Bonferroni correction would artificially decrease the test power. In this case the permutation test is of even higher value. Unfortunately, the running time is linearly correlated with the number of tests, which causes this algorithm to become prohibitively slow even with a few hundred SNPs.

1.6 Summary of Articles Included in this Thesis

1. **Computational problems in noisy SNP and haplotype analysis: block scores, block identification and population stratification.**
Gad Kimmel, Roded Sharan and Ron Shamir.
Published in *Proceedings of the Workshop on Algorithms in Bioinformatics (WABI 03)* [27] and in *INFORMS Journal on Computing* [28].

In this article, we studied several problems arising in haplotype block partitioning. Our objective was to find a solution minimizing the total number of distinct haplotypes in blocks. We showed that the problem is NP-hard when there are errors or missing data, and provided approximation algorithms for several problem variants. We also gave an algorithm that solves the problem with high probability under a probabilistic model that allows noise and missing data. In addition, we studied the multi-population case, where one has to partition the haplotypes into populations and seek a different block partition in each one. We provided a heuristic for that problem and used it to analyze simulated and real data. On simulated data, our blocks resembled the true partition more than the blocks generated by the LD-based algorithm. On single-population real data, we generated a more concise block description than do extant approaches, with better average LD within blocks. The algorithm also gave promising results on real two-population genotype data.

2. **The incomplete perfect phylogeny haplotype problem.**
Gad Kimmel and Ron Shamir.
Published in *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes* [21] and in *Journal of Bioinformatics and Computational Biology (JBCB)* [25].

In this paper, we proved that the perfect phylogeny haplotype problem is NP-complete when some of the data entries are missing, even when the phylogeny is rooted. We defined a biologically motivated probabilistic model for genotype generation and for the way missing data occur. Under this model, we developed an algorithm that takes an expected polynomial time. In tests on simulated data, our algorithm quickly resolved the genotypes under high rates of missing entries.

3. **Maximum likelihood resolution of multi-block genotypes.**
Gad Kimmel and Ron Shamir.
Published in *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 04)* [22].

In this article, we developed a new algorithm to handle genotype phasing and block partitioning in one process. Our analysis was based on a stochastic model for blocks, in which haplotypes are generated from a small number of core haplotypes, allowing for mutations, rare recombinations and errors. In our model,

common haplotypes are redefined in a probabilistic setting. The model allows errors and rare haplotypes, and the algorithm is particularly tailored to the practical situation in which the number of common haplotypes is small. We formulated genotype resolution and block partitioning as a maximum likelihood problem, and solved it by an EM algorithm. We applied our algorithm to several examples of real biological SNP data, and it outperformed two state of the art phasing algorithms.

4. **GERBIL: genotype resolution and block identification using likelihood.**
Gad Kimmel and Ron Shamir.
Published in *Proceedings of the National Academy of Sciences of the United States of America (PNAS)* [24].

This work is a direct continuation of the former one. Here, we improved the methods of the previous paper for genotype phasing. Specifically, we used several mathematical techniques for modeling the data more accurately, such as minimum description length (MDL). The main focus in this paper was to test the algorithm on a large number of biological data sets from different sources, and to compare its accuracy and speed to other phasing algorithms. The algorithm was implemented in a software package called GERBIL (GEnotype Resolution and Block Identification using Likelihood) which is efficient and simple to use. We tested GERBIL on large-scale sets of genotypes from four sources. It outperformed two state-of-the-art phasing algorithms. The PHASE algorithm (version 2.0.2) was slightly more accurate than GERBIL when allowed to run with default parameters, but required two orders of magnitude more time. When using comparable running times, GERBIL was consistently more accurate. For data sets with hundreds of genotypes, the time required by PHASE becomes prohibitive. We concluded that GERBIL has a clear advantage for studies that include many hundreds of genotypes, and in particular for large-scale disease studies.

5. **A block-free hidden Markov model for genotypes and its application to disease association.**
Gad Kimmel and Ron Shamir.
Published in *Journal of Computational Biology* [23].

In this paper, we presented a new stochastic model for genotype generation.

The model offers a compromise between rigid block structure and no structure altogether: It reflects a general blocky structure of haplotypes, but also allows for "exchange" of haplotypes at non-boundary SNP sites; it also accommodates rare haplotypes and mutations. The model can be viewed as a natural generalization of the one presented in the two previous works. We inferred the parameters of the model by an Expectation - Maximization algorithm. The algorithm was implemented in a software package called HINT (Haplotype INference Tool), and tested on 58 data sets of genotypes. To evaluate the utility of the model in association studies, we used biological data to create a simple disease association search scenario. When comparing HINT to three other models, HINT predicted association most accurately.

After this work was accepted for publication, a similar model was independently published by Rastas et al. [45] and by Stephens et al. [53]. Both of these works used this model for performing more accurate and fast phasing of genotypes.

6. **Tag SNP selection in genotype data for maximizing SNP prediction accuracy.**
Eran Halperin*, Gad Kimmel* and Ron Shamir (* equal contribution).
Published in *Bioinformatics* journal supplement for the proceedings of *The 13th Annual International Conference on Intelligent Systems for Molecular Biology (ISMB 2005)* [16].

Studies seeking disease association are often limited by the cost of genotyping SNPs. Therefore, it is essential to find a small subset of informative SNPs (tag SNPs) that may be used as good representatives of the rest of the SNPs. In this article, we defined a new natural measure for evaluating the prediction accuracy of a set of tag SNPs, and used it to develop a new method for tag SNPs selection. Our method is based on a novel algorithm that predicts the values of the rest of the SNPs given the tag SNPs. In contrast to most previous methods, our prediction algorithm uses the genotype information and not the haplotype information of the tag SNPs. Our method is very efficient, and it does not rely on having a block partition of the genomic region. We compared our method to two state of the art tag SNP selection algorithms on 58 different genotype data sets from four different sources. Our method consistently found tag SNPs with considerably better prediction ability than the other methods.

7. **A fast method for computing high significance disease association in large population-based studies.**
Gad Kimmel and Ron Shamir.
An invited oral presentation in *The Biology of Genomes* meeting, Cold Spring Harbor Laboratory, 2006. Published in *American Journal of Human Genetics* [26].

In this paper, we presented a faster algorithm for calculating the accurate p-value of a case-control association permutation test. Unlike several previous methods, we do not assume a specific distribution function of the traits given the genotypes. Our method is based on importance sampling and on accounting for the decay in linkage disequilibrium along the chromosome. The algorithm is dramatically faster than the standard permutation test. For example, when testing marker-trait association in simulations with three thousands SNPs and one thousand of cases and controls, it was over 5,000 times faster. On 10,000 SNPs from Chromosome 1, a speed-up of more than 20,000 was achieved. Our method significantly increases the problem size range for which accurate, meaningful association results are attainable.

Chapter 2

Articles

INFORMS Journal on Computing
Vol. 16, No. 4, Fall 2004, pp. 360–370
ISSN 0899-1499 | EISSN 1526-5528 | 04 | 1604 | 0360

DOI 10.1287/ijoc.1040.0088

Computational Problems in Noisy SNP and Haplotype Analysis: Block Scores, Block Identification, and Population Stratification

Gad Kimmel
School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel, kgad@tau.ac.il

Roded Sharan
International Computer Science Institute, 1947 Center St., Suite 600, Berkeley, California 94704, USA, roded@icsi.berkeley.edu

Ron Shamir
School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel, rshamir@tau.ac.il

The study of haplotypes and their diversity in a population is central to disease-association research. We study several problems arising in haplotype block partitioning. Our objective function is the total number of distinct haplotypes in blocks. We show that the problem is NP-hard when there are errors or missing data, and provide approximation algorithms for several of its variants. We also give an algorithm that solves the problem with high probability under a probabilistic model that allows noise and missing data. In addition, we study the multipopulation case, where one has to partition the haplotypes into populations and seek a different block partition in each one. We provide a heuristic for that problem and use it to analyze simulated and real data. On simulated data, our blocks resemble the true partition more than the blocks generated by the LD-based algorithm of Gabriel et al (2002). On single-population real data, we generate a more concise block description than do extant approaches, with better average LD within blocks. The algorithm also gives promising results on real two-population genotype data.

Key words: haplotype; block; genotype; SNP; subpopulation; stratification; algorithm; complexity
History: Accepted by Harvey J. Greenberg, Guest Editor; received August 2003; revised March 2004; accepted April 2004.

1. Introduction

The availability of a nearly complete human genome sequence makes it possible to look for telltale differences between DNA sequences of different individuals on a genome-wide scale, and to associate genetic variation with medical conditions. The main source of such information is single nucleotide polymorphisms (SNPs). Millions of SNPs have already been detected (Sachidanandam et al. 2001, Venter et al. 2001), out of an estimated total of 10 million common SNPs (Kruglyak and Nickerson 2001). This abundance is a blessing, as it provides very dense markers for association studies. Yet, it is also a curse, as the cost of typing every individual SNP becomes prohibitive. Haplotype blocks allow researchers to use the plethora of SNPs at a substantially reduced cost.

The sequence of alleles in contiguous SNP positions along a chromosomal region is called a *haplotype*. A major recent discovery is that haplotypes tend to be preserved along relatively long genomic stretches, with recombination occurring primarily in narrow regions called *hot spots* (Gabriel et al. 2002, Patil et al. 2001). The regions between two neighboring hot spots are called *blocks*, and the number of distinct haplotypes within each block that are observed in a population is very limited: typically, some 70% to 90% of the haplotypes within a block belong to very few (two to five) *common haplotypes* (Patil et al. 2001). The remaining haplotypes are called *rare haplotypes*. This finding is very important to disease-association studies because once the blocks and common haplotypes are identified, one can hopefully obtain a much stronger association between a haplotype and a disease phenotype. Moreover, rather than typing every individual SNP, one can choose few representative SNPs from each block that suffice to determine the haplotype. Using such *tag SNPs* allows a major saving in typing costs.

Due to their importance, blocks have been studied quite intensively recently. Daly et al. (2001) and Patil et al. (2001) used a greedy algorithm to find a partition into blocks that minimizes the total number of

SNPs that distinguish a prescribed fraction of the haplotypes in each block. Zhang et al. (2002) provided a dynamic-programming algorithm for the same purpose. Koivisto et al. (2003) provided a method based on minimum description length to find haplotype blocks. Bafna et al. (2003) proposed a combinatorial measure for comparing block partitions and suggested a different approach to find tag SNPs that avoids the partition into blocks. For an excellent, recent review on computational aspects of haplotype analysis, see Halldorsson et al. (2003).

In this paper we address several problems that arise in haplotype studies. Our starting point is a very natural optimization criterion: we wish to find a block partition that minimizes *the total number of distinct haplotypes* that are observed in all the blocks. This criterion for evaluating a block partition follows naturally from the above-mentioned observation: within blocks in the human genome, only a few common haplotypes are observed (Patil et al. 2001, Daly et al. 2001, Gabriel et al. 2002). The same criterion is used in the pure-parsimony approach for haplotype inference, where the problem is to resolve genotypes into haplotypes, using a minimum number of distinct haplotypes (Gusfield 2003). In this case, the problem was shown to be NP-hard (Hubbell 2003, cf. Halldorsson et al. 2003). This criterion was also proposed by Gusfield (2001) as a secondary criterion in refinements to Clark's inference method (Clark 1990). Minimizing the total number of haplotypes in blocks can be done in polynomial time (if there are no data errors) using a dynamic-programming algorithm. As we shall show, the problem becomes hard when errors are present or some of the data are missing. In fact, the problem of scoring a single given block turns out to be the bottleneck. Note that in practice, one has to account for rare haplotypes and hence minimize the total number of *common* haplotypes.

The input to all the problems we address is a *haplotype matrix A* with columns corresponding to SNPs in their order along the chromosome and rows corresponding to individual chromosomal segments typed. Because virtually all SNP sites have two alleles, we adopt the common assumption that the matrix is binary after we transform the two distinct alleles at each site arbitrarily to 0 and 1. A_{ij} is the allele type of chromosome i in SNP j. The first set of problems that we study concerns the scoring of a single block in the presence of errors or missing data. In one problem variant, we wish to find a minimum number of haplotypes such that by making at most E changes in the matrix, each row vector is transformed into one of them. We call this problem *total block errors* (TBE). We show that the problem is NP-hard, and provide a polynomial 2-approximation algorithm to a variant of TBE, where one wishes to minimize the total number of errors induced by the solution and the number of common haplotypes is bounded. In a second problem, we wish to minimize the number of haplotypes when the *maximum* number of errors between a given row and its (closest) haplotype is bounded by e. We call this problem *local block errors* (LBE). This problem is shown to be NP-hard too, and we provide a polynomial algorithm (for fixed e) that guarantees a logarithmic approximation factor. In a third variant, some of the data entries are missing (manifested as *question marks* in the block matrix), and we wish to complete each of them by zero or one so that the total number of resulting haplotypes is minimized. Again, we show that this *incomplete haplotypes* (IH) problem is NP-hard. To overcome the hardness we resort to a probabilistic approach. We define a probabilistic model for generating haplotype data, including errors, missing data, and rare haplotypes, and provide an algorithm that scores a block correctly with high probability under this model.

Another problem that we address is stratifying the haplotype populations. It has been shown that the block structure in different populations is different (Gabriel et al. 2002). When the partition of the sample haplotypes into subpopulations is unknown, determining a single block structure for all the haplotypes can create artificial solutions with far too many haplotypes. We define the *minimum block haplotypes* (MBH) problem, where one has to partition the haplotyped individuals into subpopulations and provide a block structure for each one so that the total number of distinct haplotypes over all subpopulations and their blocks is minimum. We show that MBH is NP-hard, but we also provide a heuristic for solving it in the presence of errors, missing data, and rare haplotypes. The algorithm uses ideas from the probabilistic analysis.

We applied our algorithm to several synthetic and real datasets. We show that the algorithm can identify the correct number of subpopulations in simulated data, and that it is robust to noise sources. On simulated data, when compared to the LD-based algorithm of Gabriel et al. (2002), we show that our algorithm forms a partition into blocks that is much more faithful to the true one. On a real dataset of Daly et al. (2001) we generate a more concise block description than do extant approaches, with a better average value of high LD-confidence fraction within blocks. As a final test, we applied our MBH algorithm to the two largest subpopulations reported in Gabriel et al. (2002). As these were genotype data, we treated heterozygotes as missing data. Nevertheless, the algorithm determined that there are two subpopulations and correctly classified over 95% of the haplotypes.

The paper is organized as follows. In §2 we study the complexity of scoring a block under various noise sources and present our probabilistic scoring algorithm. In §3 we study the complexity of the MBH problem and describe a practical algorithm for solving it. Section 4 contains our results on simulated and real data.

A preliminary version of the results of this paper is to appear in Proceedings of the Third Workshop on Algorithms in Bioinformatics (WABI) (Kimmel et al. 2003).

2. Scoring Noisy Blocks

In this section we study the problem of minimizing the number of distinct haplotypes in a single block under various noise sources. This number will be called the *score* of the block. The scoring problem arises as a key component in block partitioning in single- and multiple-population situations.

The input is a haplotype matrix A with n rows (haplotypes) and m columns (SNPs). A may contain errors (where 0 is replaced by 1 and vice versa), resulting from point mutations or measurement errors, and missing entries, denoted by "?". Clearly, if there are no errors or missing data then a block can be scored in time proportional to its size by a hashing algorithm. Below we define and analyze several versions of the scoring problem that incorporate errors into the model. We assume until §2.4 that there are no rare haplotypes. In the following we denote by v_i the ith row vector (haplotype) of A, and by $V = \{v_1, \ldots, v_n\}$ the set of all n row vectors.

2.1. Minimizing the Total Number of Errors

First we study the following problem: We are given an integer E, and wish to determine the minimum number of (possibly new) haplotypes, called *centroids*, such that by changing at most E entries in A, every row vector is transformed into one of the centroids. Formally, let $h(\cdot, \cdot)$ denote the Hamming distance between two vectors. Define the following problem:

Problem 1 (Total Block Errors (TBE)). Given a binary haplotype matrix A and an integer E, find a minimum number k of centroids $v_1, \ldots, v_k$, such that $\sum_{u \in V} \min_i h(u, v_i) \leq E$.

Determining if $k = 1$ can be done trivially in $O(nm)$ time by observing that the minimum number of errors is obtained when choosing v_1 to be the consensus vector of the rows of A. The general problem, however, is NP-hard, as shown below:

Theorem 1. *TBE is NP-hard.*

Proof. We provide a reduction from VERTEX COVER (Garey and Johnson 1979). Given an instance $(G = (W = \{w_1, \ldots, w_m\}, F = \{e_1, \ldots, e_n\}), k)$ of VERTEX COVER, where w.l.o.g. $k < m - 1$, we form an instance $(A, k + 1, E)$ of TBE. A is an $(n + mn^2) \times m$ matrix, whose rows are constructed as follows:

(1) For each edge $e_i = (s, t) \in F$, we form a binary vector v_{e_i} with 1 in positions s and t, and 0 in all other positions.

(2) For vertex $w_i \in W$ define the *vertex vector* u_i as the vector with 1 in its ith position, and 0 otherwise. For each $w_i \in W$ we form a set U_i of n^2 identical copies of u_i.

Finally, define $E = n + n^2(m - k)$. We shall prove that G has a vertex cover of size at most k if and only if there is a solution to TBE on A with at most $k + 1$ centroids and E errors.

($\Rightarrow$) Suppose that G has a vertex cover $\{w_1, \ldots, w_t\}$ with $t \leq k$. Take some cover with $t = k$. Partition the rows of A into the following subsets: for $1 \leq i \leq t$ the ith subset will contain all vectors corresponding to edges that are covered by v_i (if an edge is covered by two vertices, choose one arbitrarily), along with the n^2 vectors in U_i. Its centroid will be w_i. The $(t+1)$st subset will contain all vectors corresponding to vertices of G that are not members of the vertex cover, with its centroid being the all-0 vector. It is easy to verify that the number of errors induced by this partition is exactly $n + n^2(m - k) = E$.

($\Leftarrow$) Suppose that A can be partitioned into at most $t+1$ subsets with corresponding centroids (with $t \leq k$) such that the number E^* of induced errors is at most E. In particular, examine a partition that induces a minimum number of errors. W.l.o.g., we can assume that for each i all vectors in U_i belong to the same set in the partition. For each vertex $i \in W$, the set U_i induces at least n^2 errors, unless u_i is one of the centroids. Let l be the number of centroids that correspond to vertex vectors. Then the number E' of errors induced by the remaining $m - l$ sets of vertex vectors is at least $(m-l)n^2$. But because $E' \leq E^*$, it follows that $(m - l)n^2 \leq E = (m - k)n^2 + n$. Hence, $k \leq l + 1/n$ and by integrality $k \leq l$. Now, $l \leq t + 1 \leq k + 1$. Suppose to the contrary that $l = k + 1$. Because the Hamming distance of any two distinct vertex vectors is 2, we get $E' \geq 2(m-k-1)n^2 > E$ (because $m > k+1$), a contradiction. Thus, $l = k$. We claim that these k vertices form a vertex cover of G. By the argument above, every other vertex vector must belong to the $(k+1)$st subset and, moreover, its centroid must be the all-0 vector. Consider a vector w corresponding to an edge (u, w). If w is assigned to the $(k+1)$st subset, it adds 2 to E^*. Similarly, if w is assigned to one of the first k subsets corresponding to a vertex v, and $u, w \neq v$, then w adds 2 to E^*. Because there are n edges and the assignment of vertex vectors induced $E' = n^2(m-k) \geq E - n$ errors, each edge can induce at most one error. Hence, each edge induces exactly one error, implying that every edge is incident to one of the k vertices. □

Due to the hardness of TBE, we resort to enumerative approaches. We study the optimization version where E is to be minimized. A straightforward approach is to enumerate the centroids in the solution and assign each row vector of A to its closest centroid. Suppose there are k centroids in an optimum solution. Then the complexity of this approach is $O(kmn2^{mk})$, which is feasible only for very small m and k. In the following we present an alternative approach. We devise a $(2-2/n)$-approximation algorithm, which takes $O(n^2m+kn^{k+1})$ time.

To describe the algorithm and prove its correctness we use the following lemma, that focuses on the problem of seeking a single centroid $v \in W$ for the set of vectors $W=\{v_1,\ldots,v_n\}$. Denote $\tilde{v}_b \equiv \arg\min_{v\in\{0,1\}^m}\sum_{i=1}^n h(v,v_i)$, and let $E \equiv \sum_{v\in W} h(v,\tilde{v}_b)$.

Lemma 2. *Let* $v_b = \arg\min_{v\in W}\sum_{i=1}^n h(v,v_i)$. *Then* $\sum_{i=1}^n h(v_b,v_i) \le (2-2/n)E$.

Proof. Define $s \equiv \sum_{1\le i<j\le n} h(v_i,v_j)$. We first claim that $s \le E(n-1)$. Then,

$$s=\sum_{i<j} h(v_i,v_j) \le \sum_{i<j}[h(v_i,\tilde{v}_b)+h(\tilde{v}_b,v_j)]$$
$$=(n-1)\sum_i h(v_i,\tilde{v}_b)=(n-1)E.$$

The first inequality follows because the Hamming distance satisfies the triangle inequality. The last equality follows by using $\tilde{v}_b$ as the centroid. This proves the claim.

By the definition of v_b, for every $v_c \ne v_b$ we have

$$\sum_{v_i\in V} h(v_b,v_i) \le \sum_{v_i\in V} h(v_c,v_i).$$

Summing the above inequality for all n vectors, noting that $h(v,v)=0$, we get

$$n\sum_{v_i\in V} h(v_b,v_i) \le 2\sum_{1\le i<j\le n} h(v_i,v_j)=2s\le 2E(n-1). \quad \square$$

Theorem 3. *TBE can be* $(2-2/n)$*-approximated in* $O(n^2m+kn^{k+1})$ *time.*

Proof. Algorithm: Our algorithm enumerates all possible subsets of k rows in A as centroids, assigns each other row to its closest centroid, and computes the total number of errors in the resulting solution.

Approximation Factor. Consider two (possibly equal) partitions of the rows of A: $P_{\text{alg}}=(A_1,\ldots,A_k)$, the one returned by our algorithm; and $P_{\text{best}}=(\hat{A}_1,\ldots,\hat{A}_k)$, a partition that induces a minimum number of errors. For $1\le i\le k$ denote

$$v_b^i=\arg\min_{v\in A_i}\sum_{v_j\in A_i} h(v,v_j), \quad \hat{v}_b^i=\arg\min_{v\in \hat{A}_i}\sum_{v_j\in \hat{A}_i} h(v,v_j).$$

The number of errors induced by P_{alg} and P_{best} are

$$E_{\text{alg}}=\sum_{i=1}^k\sum_{v\in A_i} h(v_b^i,v) \quad \text{and} \quad E_{\text{best}}=\sum_{i=1}^k\sum_{v\in \hat{A}_i} h(\hat{v}_b^i,v),$$

respectively. Finally, let $n_i=|\hat{A}_i|$ and denote by e_i the minimum number of errors induced in subset $\hat{A}_i$, by the optimal solution. In particular, $\sum_{i=1}^k n_i = n$ and $\sum_{i=1}^k e_i = E$.

Because our algorithm checks all possible solutions that use k of the original haplotypes as centroids and chooses a solution that induces a minimal number of errors, $E_{\text{alg}} \le E_{\text{best}}$. By Lemma 2, $\sum_{v\in\hat{A}_i} h(\hat{v}_b^i,v) \le (2-2/n_i)e_i$ for every $1\le i\le k$. Summing this inequality over all $1\le i\le k$ we get

$$E_{\text{alg}} \le E_{\text{best}}=\sum_{i=1}^k\sum_{v\in\hat{A}_i} h(\hat{v}_b^i,v) \le \sum_{i=1}^k\left(2-\frac{2}{n_i}\right)e_i$$
$$\le \sum_{i=1}^k\left(2-\frac{2}{n}\right)e_i=\left(2-\frac{2}{n}\right)E.$$

Complexity. As a preprocessing step we compute the Hamming distance between every two rows in $O(n^2m)$ time. There are $O(n^k)$ possible sets of centroids. For each centroid set, assigning rows to centroids and computing the total number of errors takes $O(kn)$ time. The complexity follows. $\square$

We note that Ostrovsky et al. (2002) presented a probabilistic algorithm for the above problem, with an approximation ratio of $(1+4\sqrt{\epsilon})^2$, where $\frac{1}{4}\ge\epsilon>0$.

2.2. Handling Local Data Errors

In this section we treat the question of scoring a block when the *maximum* number of errors between a haplotype and its centroid is bounded. Formally, we study the following problem.

Problem 2 (Local Block Errors (LBE)). Given a block matrix A and an integer e, find a minimum number k of centroids $v_1,\ldots,v_k$ and a partition $P=(V_1,\ldots,V_k)$ of the rows of A, such that $h(u,v_i)\le e$ for every i and every $u\in V_i$.

Theorem 4. *LBE is NP-hard even when* $e=1$.

Proof. We use the same construction as in the proof of Theorem 1. We claim that the VERTEX COVER instance has a solution of cardinality at most k if and only if the LBE instance has a solution of cardinality at most $k+1$, such that at most one error is allowed in each row. The "only if" part is immediate from the proof of Theorem 1. For the "if" part, observe that any two vectors corresponding to a pair of independent edges cannot belong to the same subset in the partition, and so is the case for a vertex vector and any vector corresponding to an edge that is not incident on that vertex. This already implies a vertex cover of size at most $k+1$. Because $m>k+1$ there must be a subset in the partition that contains at least two vectors corresponding to distinct vertices.

But then either it contains no edge vector, or it contains exactly one edge vector and the vectors corresponding to its endpoints. In any case we obtain a vertex cover of the required size. □

THEOREM 5. *There is an $O(\log n)$ approximation algorithm for LBE that takes $O(n^2m^e)$ time.*

PROOF. Our approximation algorithm for LBE is based on a reduction to SET COVER. Let V be the set of row vectors of A. Define the *e-set* of a vector v as the set of vectors of the same length that have Hamming distance at most e to v. Denote this e-set by $e(v)$. Let U be the union of all e-sets of row vectors of A. We reduce the LBE instance to a SET COVER instance $(V, \mathcal{S})$, where $\mathcal{S} \equiv \{e(v) \cap V\colon v \in U\}$. Clearly, there is a 1-1 correspondence between solutions for the LBE instance and solutions for the SET COVER instance, and that correspondence preserves the cardinality of the solutions. We now apply an $O(\log n)$-approximation algorithm for SET COVER (see, e.g., Cormen et al. 1990) to $(V, \mathcal{S})$ and derive a solution to the LBE instance, which is within a factor of $O(\log n)$ of optimal. The complexity follows by observing that $|U| = O(nm^e)$. □

2.3. Handling Missing Data

In this section we study the problem of scoring an *incomplete matrix*, i.e., a matrix in which some of the entries may be missing. The problem is formally stated as follows.

PROBLEM 3 (INCOMPLETE HAPLOTYPES (IH)). Given an incomplete haplotype matrix A, complete the missing entries so that the number of haplotypes in the resulting matrix is minimum.

THEOREM 6. *IH is NP-hard.*

PROOF. We present a reduction from GRAPH COLORING (Garey and Johnson 1979). Given an instance $(G = (W, E), k)$ of GRAPH COLORING we build an instance (A, k) of IH as follows. Let $W = \{1, \ldots, n\}$. Each $i \in W$ is assigned an n-dimensional row vector v_i in A with 1 in the ith position, 0 in the jth position for every $(i, j) \in E$, and "?" in all other positions.

Given a k-coloring of G, let $W_1, \ldots, W_k$ be the corresponding color classes. For each class $W_i = \{v_{j_1}^{(i)}, \ldots, v_{j_i}^{(i)}\}$ we complete the ?s in the vectors corresponding to its vertices as follows. Each ? in one of the columns $v_{j_1}^{(i)}, \ldots, v_{j_i}^{(i)}$ is completed to 1, and all the others are completed to 0. The resulting matrix contains exactly k distinct haplotypes. Each haplotype corresponds to a color class, and has 1 in position i if and only if i is a member of the color class.

Conversely, given a solution to IH of cardinality at most k, each of the solution haplotypes corresponds to a color class in G. This follows because any two vectors corresponding to adjacent vertices must have a column with both 0 and 1 and, thus, represent two different haplotypes. □

2.4. A Probabilistic Algorithm

In this section we define a probabilistic model for the generation of haplotype block data. The model is admittedly naive, in that it assumes equal allele frequencies and independence between different SNPs and distinct haplotypes. However, as we shall see in §§3 and 4, it provides useful insights towards an effective heuristic that performs well on real data. We give a polynomial algorithm that computes the optimal score of a block under this model with high probability (w.h.p.). Our model allows for all three types of confusing signals mentioned earlier: rare haplotypes, errors, and missing data.

Denote by T the hidden true haplotype matrix, and by A the observed one. Let T' be a submatrix of T, which contains one representative of each haplotype in T (common and rare). We assume that the entries of T' are drawn independently according to a Bernoulli distribution with parameter 0.5. T is generated by duplicating each row in T' an arbitrary number of times. This completes the description of the probabilistic model for T. Note that we do not make any assumption on the relative frequencies of the haplotypes. We now introduce errors to T by independently flipping each entry of T with probability $\alpha < 0.5$. Finally, each entry is independently replaced with a ? with probability p. Let A be the resulting matrix, and let A' be the submatrix of A induced by the rows in T'. Under these assumptions, the entries of A' are independently identically distributed as follows: $A'_{ij} = 0$ with probability $(1-p)/2$, $A'_{ij} = 1$ with probability $(1-p)/2$ and $A'_{ij} = ?$ with probability p.

We say that two vectors x and y have a *conflict* in position i if one has value 1 and the other has value 0 in that position. Define the dissimilarity $d(x, y)$ of x and y as the number of their conflicting positions (in the absence of ?s, this is just the Hamming distance). We say that x is *independent* of y and denote it by $x \parallel y$, if x and y originate from two different haplotypes in T. Otherwise, we say that x and y are *mates* and denote it by $x \approx y$. Intuitively, independent vectors will have higher dissimilarity compared to mates. In particular, for any i:

$$\begin{aligned} p_I &\equiv \text{Prob}(x_i = y_i \mid x \parallel y;\, x_i, y_i \in \{0,1\}) = 0.5, \\ p_M &\equiv \text{Prob}(x_i = y_i \mid x \approx y;\, x_i, y_i \in \{0,1\}) \\ &= \alpha^2 + (1-\alpha)^2 > 0.5. \end{aligned} \tag{1}$$

PROBLEM 4 (PROBABILISTIC MODEL BLOCK SCORING (PMBS)). Given an incomplete haplotype block matrix A, find a minimum number k of centroids $v_1, \ldots, v_k$, such that under the above probabilistic model, with high probability, each vector $u \in A$ is a mate of some centroid.

Score(A):

1. Let V be the set of rows in A.
2. Initialize a heap S.
3. **While** $V \neq \emptyset$ **do**:
   (a) Choose some $v \in V$.
   (b) $H \leftarrow \{v' \in V \mid d(v, v') \leq t^*\}$.
   (c) $V \leftarrow V \setminus H$.
   (d) Insert(S,$|H|$).
4. Output S.

Figure 1 An Algorithm for Scoring a Block Under a Probabilistic Model of the Data

Note. Procedure insert(S, s) inserts a number s into a heap S.

Our algorithm for scoring a block A under the above probabilistic model is described in Figure 1. It uses a threshold t^* on the dissimilarity between vectors to decide on mate relations. We set t^* to be the average of the expected dissimilarity between mates and of the expected dissimilarity between independent vectors (see proof of Theorem 7). The algorithm produces a partition of the rows into mate classes of cardinalities $s_1 \geq s_2 \geq \cdots \geq s_l$. Given any lower bound γ on the fraction of rows that need to be covered by the common haplotypes, we give A the score $h = \arg\min_j \sum_{i=1}^{j} s_i \geq \gamma n$. We prove below that w.h.p. h is the correct score of A.

THEOREM 7. *If $m = \omega(\log n)$ then w.h.p. the algorithm computes the correct score of A.*

PROOF. We prove that w.h.p. each mate relation decided by the algorithm is correct. Applying a union bound over all such decisions will give the required result. Fix an iteration of the algorithm at which v is the chosen vertex and let $v' \neq v$ be some row vector in A. Let X_i be a binary random variable that is 1 if and only if v_i and v'_i are in conflict. Clearly, all X_i are independent identically distributed Bernoulli random variables. Define $X \equiv d(v, v') = \sum_{i=1}^{m} X_i$ and $f \equiv (1-p)^2$. Using (1) we conclude:

$$(X \mid v' \parallel v) \sim \text{Binom}(m, f(1-p_I)),$$
$$(X \mid v' \approx v) \sim \text{Binom}(m, f(1-p_M)).$$

We now require the following Chernoff bound (cf. Alon and Spencer 2000). If $Y \sim \text{Binom}(n, s)$ then for every $\epsilon > 0$ there exists $c_\epsilon > 0$ that depends only on ϵ, satisfying:

$$\text{Prob}[|Y - ns| \geq \epsilon ns] \leq 2e^{-c_\epsilon ns}.$$

Let $\mu = mf(1-p_M)$. Define $\epsilon \equiv [(1-p_I) - (1-p_M)]/(2(1-P_M))$ and $t^* \equiv \epsilon\mu$. Applying the Chernoff bound and using the assumption that $m = \omega(\log n)$, we have that for all $c > 0$:

$$\text{Prob}(X > t^* \mid v' \approx v) \leq 2e^{-c_\epsilon m} < \frac{1}{n^c},$$
$$\text{Prob}(X \leq t^* \mid v' \parallel v) < \frac{1}{n^c}.$$

Because we check whether $d(v, v') < t^*$ a total of $O(n^2)$ times, by applying a union bound we conclude that the probability that throughout the algorithm some implied mate relation is incorrect and is bounded by a polynomial in $1/n$. □

When using the algorithm as part of a practical heuristic (see §3), we do not report the rare haplotypes. Instead, we report only the smallest number of the most abundant haplotypes as computed by the algorithm that together capture a fraction γ of all haplotypes.

3. The Multipopulation Case

Suppose that the matrix A contains haplotypes from several homogeneous populations. The partitioning into blocks can differ among populations (Gabriel et al. 2002). Here, we study the question of reconstructing the partition of the rows of A into sets called *subpopulations*, and the columns in each set into blocks, such that the sum of the scores of the submatrices corresponding to these blocks is minimized.

PROBLEM 5 (MINIMUM BLOCK HAPLOTYPES (MBH)). Given a haplotype matrix A, find a partition of its rows into subpopulations so that the total number of block haplotypes is minimized.

In practice, we usually have full information on the population from which each of the haplotypes originates. However, in certain situations there may be a hidden stratification of a population that can affect the conclusions of association studies on it. Problem 5 aims to address such situations.

3.1. Minimum Block Haplotypes

For a haplotype matrix A and a subset S of its rows, we denote by H_S^A the (minimum) total number of block haplotypes in an optimal partition of S into blocks. Our goal is to find $H^A = H_V^A$. Given a partition $P = (P_1, \ldots, P_r)$ of the rows of A into subpopulations, we let $H^A(P) = \sum_{i=1}^{r} H_{P_i}^A$, that is, the (minimum) total number of block haplotypes in an optimal partition of each subpopulation into blocks. In the following we omit the superscript A when it is clear from the context. Given a partition P, $H(P)$ can be polynomially computed in the noiseless case using a simple adaptation of the dynamic-programming algorithm of Zhang et al. (2002). However, the general MBH problem is NP-hard.

THEOREM 8. *MBH is NP-hard.*

PROOF. We provide a reduction from VERTEX COVER (Garey and Johnson 1979). Let $(G=(V,E),k)$ be an instance of VERTEX COVER where $|V|=n$, $|E|=m$, and w.l.o.g. $n<m$. We build an instance $(A, n(8m+4+2m^2)+12m+2k)$ of (the decision version of) MBH as follows. We associate with the vertices and edges of G row vectors of dimension $c=(2n+1)m^{10}$. These vectors will constitute the matrix A. Each of the row vectors v is partitioned into segments where the segment of length m^{10} between positions $i^- \equiv (i-1)2m^{10}+1$ and $i^+ \equiv (i-1)2m^{10}+m^{10}$ corresponds to vertex i. The m^{10} last positions in v are called its *tail*.

The content of each segment will be a periodic binary sequence. For an integer k let S_k be the sequence $(0,\ldots,0,1)$ of length k where $S_0=(0)$ and $S_1=(1)$. For convenience we denote S_k also as S_k^1, and use S_k^{-1} to denote the complement of that sequence. Each of the vector segments consist of repetitions of some S_k or its complement. We denote by $S_k(l)$ the sequence formed by concatenating copies of S_k up to a total length of l where the last copy may be truncated.

For an ordered sequence of integers $1=i_1<\cdots<i_{l+1}=c+1$, inducing a partition of $[1,\ldots,c]$, we define the following vector set:

$$U_{i_1,\ldots,i_{l+1}}(k_1,\ldots,k_l) \equiv \bigcup_{r_1,\ldots,r_l\in\{1,-1\}} \left(S_{k_1}^{r_1}(i_2-i_1),\ldots,S_{k_l}^{r_l}(i_{l+1}-i_l)\right).$$

In words, $U_{i_1,\ldots,i_{l+1}}(k_1,\ldots,k_l)$ is a set of 2^l vectors of dimension c, where the sth vector contains in its tth segment copies of $S_{k_t}^r$ with $r=1$ iff the tth bit of s is 0.

With each vertex v_i we associate the set of $2\cdot(2\cdot 4m)\cdot 2\cdot 2m^2=64m^3$ vectors:

$$V_i=\bigcup_{1\le j\le 4m,\, im^2\le k<(i+1)m^2} U_{1,i^-,i^+ +1,c-m^{10}+m^9 i,c+1}(0,j,0,k).$$

Thus, each vertex vector has four segments: until position i^- it is all zeros or all ones; between i^- and i^+ it has one of $4m$ possible sequences or their complements; until the beginning of its tail it is again all zeros or all ones; and then at a unique position, which depends on the vertex identity, starts one of $2m^2$ possible tail sequences for that vertex.

With each edge e_l: $1\le l\le m$ connecting vertices i and j, where $i<j$, we associate a set of $2\cdot(2\cdot 4)\cdot 2\cdot(2\cdot 4)\cdot 2=512$ vectors

$$E_l=\bigcup_{p=4l-3}^{4l} U_{1,i^-,i^+ +1,j^-,j^+ +1,c+1}(0,p,0,p,0).$$

Thus, each edge vector contains one of eight possible sequences in its (i^-,i^+) and (j^-,j^+) segments, and these sequences are unique for each edge.

By construction, $H_{V_i}=2+8m+2+2m^2=8m+4+2m^2$ and $H_{E_l}=16+6=22$. We now prove that G has a vertex cover of size at most k if and only if A has a partition P with $H(P)\le n(8m+4+2m^2)+12m+2k$.

($\Rightarrow$) W.l.o.g., let $\{1,\ldots,t\}$ be a vertex cover of size $t\le k$ for G. Let C_i be the set of edges covered by vertex i (for an edge covered by two vertices, choose the one with smaller index) where $C_i=\varnothing$ for $i>t$. Define $A_i\equiv V_i\cup\bigcup_{j\in C_i}E_j$ for $1\le i\le n$. Let $P=(A_1,\ldots,A_n)$. We shall prove that $H(P)$ is of the required size. Fix i and let $C_i=\{e_1,\ldots,e_p\}$ where e_j connects i to s_j and, w.l.o.g., $i<s_1<\cdots<s_p$. We claim that $H_{A_i}=(8m+4+2m^2)+12p+2\delta$ where δ is an indicator that equals 1 if and only if $i\le t$. Consider the partition of A_i into the following blocks: $(1,i^-1)$, (i^-,i^+), (i^++1, s_1^-1), $(s_1^-,s_1^+),\ldots,(s_{p-1}^++1,s_p^-1)$, (s_p^-,s_p^+), $(s_p^++1,\ c-m^{10}+m^9-1)$, $(c-m^{10}+m^9,c)$. Due to V_i, A_i has two haplotypes in the first block, $8m$ haplotypes in the second block (which corresponds to the segment of vertex i), two haplotypes in the segment before last, and $2m^2$ haplotypes in the tail block. In addition, if we add the sets E_j one by one to the same subpopulation, then every such set, corresponding to the edge (i,s_j), adds two new blocks and 12 haplotypes (two haplotypes in $((j-1)^++1,j^-1)$ and $8+2$ in (j^-,j^+)). The only exception is $j=1$, for which two more haplotypes are added in the tail segment. Thus, if $|C_i|=p>0$ then $H_{A_i}=(8m+4+2m^2)+12p+2$ and if A_i contains no edge vectors then $H_{A_i}=8m+4+2m^2$. The claim follows.

($\Leftarrow$) Suppose that A has a partition $P=(A_1,\ldots,A_t)$ so that $H(P)\le n(8m+4+2m^2)+12m+2k$. In particular, examine the partition P^* for which $H\equiv H(P^*)$ is minimal. W.l.o.g. every one of V_i and E_j is completely contained in some A_k. We first claim that no set in the partition contains both V_i and V_j for $i\ne j$. Suppose this is not the case. Define a new partition P' in which V_j is moved into a new set. Then $H-H(P')\ge(2m^2+2)-8m-4>0$ where the first term is due to the tail segments of i and j and the second is due to edge vectors corresponding to edges incident on j that are possibly present in the same partition set as V_i and V_j. Thus, we arrive at a contradiction.

Now consider an edge l connecting vertices i and j, and let $A_r\supseteq E_l$. We claim that $V_i\subset A_r$ or $V_j\subset A_r$ (in P^*). To see that, observe that in the first case l adds at most 14 haplotypes to H (similar to the argument in the "only if" part of the proof), while in the second case it adds at least 16 haplotypes to H because each of the segments (i^-,i^+) and (j^-,j^+) contains eight unique haplotypes.

Finally, suppose there are t sets in P^* that contain edge vectors. Then $H\ge n(8m+4+2m^2)+12m+2t$, implying that $t\le k$ and G has a vertex cover of size at most k. □

3.2. A Polynomial Case

We now give a polynomial algorithm for a restricted version of MBH in which each subpopulation is required to be a contiguous set of rows. We call this variant *minimum contiguous block haplotypes* (*MCBH*). Its solution may be useful for designing heuristics that permute the matrix rows for local improvement. For clarity, in the discussion below we shall assume that there exists an oracle that scores a given block in $O(1)$ time. Denote the optimal solution of MCBH on A by H^A.

THEOREM 9. *MCBH can be solved in $O(n^2m^2)$ time.*

PROOF. Algorithm: Let A be an input haplotype matrix. We give a dynamic-programming procedure to solve MCBH. A key component of the algorithm is a dynamic-programming algorithm, which computes the score for a given subpopulation S in a straightforward manner, similar to Zhang et al. (2002). Let T_i^S, $0 \le i \le m$, be the minimum number of block haplotypes in the submatrix of A induced on the rows in S and the columns $1, \ldots, i$, where $T_0^S = 0$. For a pair of columns i, j let B_{ij}^S be the score of the block induced by the rows in S and the columns in $\{i, \ldots, j\}$. Then the following recursive formula can be used to compute T_m^S:

$$T_i^S = \min_{0 \le j \le i-1} T_j^S + B_{ji}^S.$$

We now use a second dynamic-programming algorithm to compute H^A. Define P_i, $0 \le i \le n$ as the minimum number of block haplotypes in any row partition of $A_{\{1,\ldots,i\}}$. Clearly, $P_0 = 0$ and $P_n = H^A$. The computation of P_i uses the following recursive formula:

$$P_i = \min_{1 \le j \le i} P_{j-1} + T_m^{\{j,\ldots,i\}}.$$

Complexity. Computing T_m^S for any S takes $O(m^2)$ time. Hence, computing H^A takes $O(n^2m^2)$ time in total. □

3.3. A Heuristic

Next, we present an efficient heuristic for MBH. The algorithm has three components: a block-scoring procedure, a dynamic-programming algorithm to find the optimum block structure for a single subpopulation, and a simulated-annealing algorithm to find an optimum partition into homogeneous subpopulations. We describe these components below.

The dynamic-programming component is as described in the first part of the proof of Theorem 9. For scoring a block within the dynamic-programming procedure we use the probabilistic algorithm described in §2.4 with a small modification: instead of using a fixed threshold t^*, we compute a different threshold $t^*_{v,v'}$ for every two vectors v, v'. This is done by counting the number l of positions in which neither of the vectors has ?, and setting $t^*_{v,v'} = \frac{1}{2}l[(1-p_M)+(1-P_I)]$. Scoring an $n \times t$ block takes $O(tnk)$ time where k is a bound on the number of common haplotypes. Hence, the dynamic program takes $O(mb^2nk)$ total time where b is an upper bound on the allowed block size. Additional saving may be possible by precomputing the pairwise distances of rows in contiguous matrix segments of size up to b.

The goal of the annealing process is to optimize the partition of the haplotypes into subpopulations. We define a *neighboring partition* as any partition that can be obtained from the current one by moving one haplotype from one group to another. The process proceeds through a sequence of neighboring partitions depending on their scores and the temperature parameter in a standard annealing fashion. A crucial factor in obtaining a good solution is the initialization of the annealing process. We perform the initialization as follows. We compute pairwise similarities between every two haplotypes. The similarity S_{uv} of vectors u and v is calculated as follows. Initially we set $S_{uv} = 0$. We then slide a window of size $w = 20$ along u and v (20 is the average size of a block). For each position i we check whether $d((u_i, \ldots, u_{i+w-1}), (v_i, \ldots, v_{i+w-1})) \le w\alpha$ (for a parameter α). If this is the case, we increment S_{uv} and jump to $i + w$ for the next iteration. Otherwise, we jump to $i + 1$. The intuition is that rows from the same subpopulation should be more similar in blocks in which they share the same haplotypes and, thus, have a better chance to hit good windows and accumulate a higher score in the scan. Next we cluster the haplotypes based on their similarity values using the K-means algorithm (MacQueen 1965). The resulting partition is taken as the starting point for the annealing process. To determine the number of subpopulations K, we try several choices and pick the one that results in the lowest score.

The running time of the practical algorithm is dominated by the cost of each annealing step. Because this step changes the haplotypes of two subpopulations only, it suffices to recompute the scores of these subpopulations only.

4. Experimental Results

4.1. Simulations

We applied our heuristic algorithm to simulated and real haplotype data. First we conducted extensive simulations to check the ability of our algorithm to detect subpopulations and recognize their block structure. Our simulation setup is as follows. We generated simulated haplotype matrices with 100 haplotypes and 300 SNPs. The number of subpopulations varied in the simulations. Subpopulations were of equal sizes. For each subpopulation we generated block

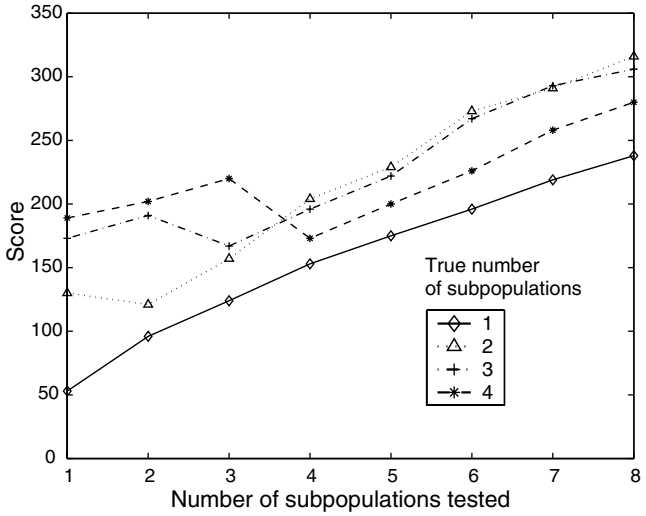


Figure 2 Simulation Results: Determining the Number of Subpopulations

Note. For each simulated matrix, containing one to four subpopulations, the score assigned by the algorithm to partitions (y axis) with different numbers of subpopulations (x axis). Simulations were performed with error level 1% and no missing entries.

boundaries using a Poisson process with rate 20. Each block within a subpopulation contained two to five common haplotypes, covering 90% of the block's rows (with the remaining 10% being rare haplotypes). Within each block of each subpopulation, the haplotype matrix was created according to the probabilistic model described in §2.4. Errors and missing data were introduced with varying rates of up to 30%.

As a first test we simulated several matrices with one to four subpopulations and applied our algorithm with K ranging from 1–8. For each K we computed the score of the partition obtained (as described in §3.3). In each of the simulations the correct number got the lowest score (Figure 2). Next we simulated several matrices with three subpopulations and different levels of errors and missing data. Table 1 summarizes our results in correctly assigning haplotypes to subpopulations (the set with the largest overlap with the true subpopulation was declared correct). It can be seen that the MBH algorithm gives highly accurate results for missing data and error levels up to 10%.

Table 1 Accuracy of Haplotype Classification by the MBH Algorithm for Different Noise Levels (Data Are for Three Subpopulations)

% Errors	% Missing entries	% Correct classifications
0	0	99
5	5	98
10	10	95
15	15	84
20	20	71

For comparison, we also implemented the LD-based algorithm of Gabriel et al. (2002) for finding blocks. We compared the block structures produced by our algorithm and by the LD-based algorithm to the correct one, using an alignment score similar to the one used in comparison of two DNA restriction enzyme maps (Waterman 1995, §9.10). The score of two partitions P_1 and P_2 of m SNPs is computed as follows. We form two vectors of size $m-1$, in which 1 in position i denotes a block boundary between SNPs i and $i+1$, and 0 denotes that the two SNPs belong to the same block. We then compute an alignment score of these vectors using an affine gap penalty model with penalties 3, 2, and 0.5 for mismatch, gap open, and gap extension, respectively, and a match score of zero.

We simulated one population with 3,000 haplotypes, computed its block structure with both algorithms, and compared them to the true one. We repeated this experiment with different error and missing-data rates. The results are shown in Figure 3. It can be observed that our algorithm yields partitions that are closer to the true ones, particularly as the rate of errors and missing data rises. An example of the actual block structures produced is shown in Figure 4.

4.2. Real Data

We applied our algorithm to two published datasets. The first dataset of Daly et al. (2001) consists of 258 haplotypes and 103 SNPs. We applied our block partitioning algorithm with the following parameters: the maximal allowed error ratio between two vectors to be considered as resulting from a single haplotype was 0.02. In addition, we allowed up to 5% rare

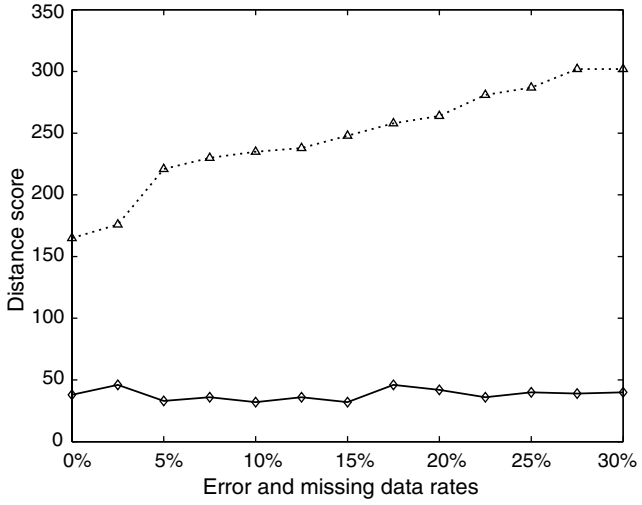


Figure 3 Accuracy in Block Reconstruction by our Algorithm (Solid Line) and the Algorithm of Gabriel et al. (2002) (Dashed Line)

Note. y axis: the score of aligning the reconstructed structure with the correct one. x axis: the noise rate.

Figure 4 **An Example of the Block Structures Produced for an Error Rate of 1% by Our Algorithm (Bottom), the LD-Based Algorithm of Gabriel et al. (2002) (Top), and the True Solution (Middle)**

Note. Each block boundary is denoted by a vertical line.

haplotypes, i.e., in scoring a block we sought the minimum number of different haplotypes that together cover at least 95% of the rows.

In order to assess our block partitioning and compare it to the one reported by Daly et al. (2001), we calculated LD-based measures for both partitions. Specifically, we calculated the LD-confidence values between every pair of SNPs inside the same block, using a χ^2 test, as follows. For a pair i, j of SNPs, let $P_{a,b}$, where $a, b \in \{0, 1\}$, be the frequency of occurrence of a in position i and b in position j of a haplotype. Let p_0, p_1 (q_0, q_1) denote the frequencies of haplotypes with 0 and 1 in the ith (jth) SNP, respectively. Define $D \equiv P_{00}P_{11} - P_{01}P_{10}$. D is a measure of linkage disequilibrium, and $nD^2/(p_0p_1q_0q_1)$ is distributed as χ^2, with one degree of freedom.

For each block, we calculated the fraction of SNP pairs in the block whose LD-confidence value exceeded 95% (*high LD pairs*). The average fraction over all blocks was computed as the ratio of the total number of high LD pairs inside blocks to the total number of SNP pairs within blocks.

A comparison between our block partition to the one obtained by Daly et al. (2001) is presented in Table 2. Overall, the two block partitions have similar boundaries and similar scores. The average fraction of high LD pairs in blocks for our partition was 0.823. For the partition of Daly et al. (2001) the average fraction was 0.796. Another partition was produced for these data by Eskin et al. (2003) based on minimizing the number of representative SNPs. Their partition

Table 2 **Comparison Between the Blocks of Daly et al. (2001) and the Blocks Generated by Our Algorithm**

Daly et al. blocks	Fraction of high LD pairs	Our blocks	Fraction of high LD pairs
1: 1–9	0.78	1: 1–15	0.81
2: 10–15	1		
3: 16–24	0.78	2: 16–24	0.78
4: 25–35	0.95	3: 25–36	0.94
5: 36–40	0.70	4: 37–44	0.68
6: 41–45	1		
7: 46–77	0.77	5: 45–67	0.84
		6: 68–78	0.71
8: 78–85	0.50	7: 79–81	0.33
9: 86–91	0.93	8: 82–90	0.89
10: 92–98	0.95	9: 91–95	1
11: 99–103	1	10: 96–103	0.75
Average	0.796		0.822

Table 3 **Separation to Subpopulations and Block Finding on Different Regions in Part of the Data of Gabriel et al. (2002)**

Chromosome: region	#SNPs	Discovered blocks	% Correct classifications
1: 3a	119	1: 1–35, 36–119 2: 1–46, 47–119	95
2: 8a	73	1: 1–73 2: 1–73	99
6: 24a	121	1: 1–52, 53–121 2: 1–44, 45–121	98
8: 29a	104	1: 1–27, 28–104 2: 1–40, 41–104	100
9: 32a	110	1: 1–25, 26–110 2: 1–38, 39–110	99
14: 41a	141	1: 1–48, 49–63, 64–141 2: 1–12, 13–63, 64–141	100

Note. Includes subpopulations A and D

contained 11 blocks and its average fraction of high LD pairs was 0.814.

The second dataset we analyzed, of Gabriel et al. (2002) contains unresolved genotype data. In order to apply our algorithm to these data, we transformed them into haplotypes by treating heterozygous SNPs as missing data. Notably, the fraction of heterozygous sites was relatively small, so the loss in information was moderate. We considered the two largest populations in the data, A (Europeans) and D (individuals from Yoruba), consisting of 93 and 90 samples, respectively. Each population was genotyped in ~60 different regions in the genome. We analyzed six of those regions that contained over 70 SNPs. In all cases we were able to detect two different populations in the data and classify correctly over 95% of the haplotypes. The results are shown in Table 3. The results with three populations were poorer, due to the smaller size of the third population.

5. Concluding Remarks

We have introduced a simple and intuitive measure for scoring and detecting blocks in a haplotype matrix: the total number of distinct haplotypes in blocks. Using this measure along with several error models, we have studied the computational problems of scoring of a block, and of finding an optimal block structure. Most versions of the scoring problem that address imperfect data are shown to be NP-hard. A similar situation occurred with the f score function of Zhang et al. (2002). We devised several algorithms for different variants of the problem. In particular, we gave a simple algorithm, which, under an appropriate probabilistic model, scores a block correctly with high probability in the presence of errors, missing data, and rare haplotypes.

Note that our measure is adequate only when the ratio of the number n of typed individuals to the

number m of SNPs is not too extreme. When n is very small and m is large, our measure might be optimized by the trivial solution of a single block.

In simulations, our score leads to more accurate block detection than does the LD-based method of Gabriel et al. (2002). While the simulation setup is quite naive, it seems to act just as favorably for the LD-based methods. The latter methods apparently tend to over-partition the data into blocks, as they demand a very stringent criterion between every pair of SNPs in the same block. This criterion is very hard to satisfy as block size increases, and the number of pairwise comparisons grows quadratically. On the data of Daly et al. (2001) we generated a slightly more concise block description than do extant approaches, with a somewhat better fraction of high-LD pairs.

We also treated the question of partitioning a set of haplotypes into subpopulations based on their different block structures, and devised a practical heuristic for the problem. On a genotype dataset of Gabriel et al. (2002) we were able to identify two subpopulations correctly, in spite of ignoring all heterozygous types. A principled method of dealing with genotype data remains a computational challenge. While in some studies the partition into subpopulations is known, others may not have this information, or further, finer partition may be detectable using our algorithm. In our model we implicitly assumed that block boundaries in different subpopulations are independent. In practice, some boundaries may be common due to the common lineage of the subpopulations. A more detailed treatment of the block boundaries in subpopulations should be considered when additional haplotype data reveal the correct way to model this situation.

Acknowledgments

R. Sharan was supported by a Fullbright grant. R. Shamir was supported by a grant from the Israel Science Foundation (Grant 309/02). The authors thank Chaim Linhart and Dekel Tsur for their comments on the manuscript.

References

Alon, N., J. H. Spencer. 2000. *The Probabilistic Method.* John Wiley and Sons, Inc., New York.

Bafna, V., B. V. Halldorsson, R. Schwartz, A. Clark, S. Istrail. 2003. Haplotyles and informative SNP selection algorithms: Don't block out information. *Proc. Seventh Annual Internat. Conf. Res. Comput. Molecular Biol.* (*RECOMB*). The Association for Computing Machinery. New York, 19–27.

Clark, A. 1990. Inference of haplotypes from PCR-amplified samples of diploid populations. *Molecular Biol. Evolution* **7** 111–122.

Cormen, T. H., C. E. Leiserson, R. L. Rivest. 1990. *Introduction to Algorithms.* MIT Press, Cambridge, MA.

Daly, M. J., J. D. Rioux, S. F. Schaffner, T. J. Hudson, E. S. Lander. 2001. High-resolution haplotype structure in the human genome. *Nature Genetics* **29**(2) 229–232.

Eskin, E., E. Halperin, R. M. Karp. 2003. Large scale reconstruction of haplotypes from genotype data. *Proc. Seventh Annual International Conf. Res. Comput. Molecular Biol.* (*RECOMB*). The Association for Computing Machinery. New York, 104–113.

Gabriel, S. B., S. F. Schaffner, H. Nguyen, J. M. Moore, J. Roy, B. Blumenstiel, J. Higgins, M. DeFelice, A. Lochner, M. Faggart, S. N. Liu-Cordero, C. Rotimi, A. Adeyemo, R. Cooper, R. Ward, E. S. Lander, M. J. Daly, D. Altshuler. 2002. The structure of haplotype blocks in the human genome. *Science* **296** 2225–2229.

Garey, M. R., D. S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Co., San Francisco, CA.

Gusfield, D. 2001. Inference of haplotypes in samples of diploid populations: Complexity and algorithms. *J. Comput. Biol.* **8**(3) 305–323.

Gusfield, D. 2003. Haplotype by pure parsimony. *Proc. Fourteenth Annual Sympos. Combin. Pattern Matching (CPM), Morelia, Mexico.* Springer, Berlin, 144–155.

Halldorsson, B. V., V. Bafna, N. Edwards, R. Lippert, S. Yooseph, S. Istrail. 2003. Combinatorial problems arising in SNP. *Discrete Math. Theoret. Comput. Sci. Lecture Notes in Computer Science*, No. 2731. Springer-Verlag, Heidelberg, Germany, 26–47.

Hubbell, E. 2003. Finding a parsimony solution to haplotype phase is NP-hard. Unpublished manuscript, Affymetrix Inc., Santa Clara, CA.

Kimmel, G., R. Sharan, R. Shamir. 2003. Identifying blocks and subpopulations in noisy SNP data. *Proc. Third Workshop Algorithms in Bioinformatics* (*WABI*). Springer-Verlag, Berlin, 303–319.

Koivisto, M., M. Perola, T. Varilo, W. Hennah, J. Ekelund, M. Lukk, L. Peltonen, E. Ukkonen, H. Mannila. 2003. An MDL method for finding haplotype blocks and for estimating the strength of haplotype block boundaries. *Proc. Pacific Sympos. Biocomputing* (*PSB*), *Big Island of Hawaii*, *Hawaii*, Vol. 8. World Scientific, Singapore, 502–513.

Kruglyak, L., D. A. Nickerson. 2001. Variation is the spice of life. *Nature Genetics* **27** 234–236.

MacQueen, J. 1965. Some methods for classification and analysis of multivariate observations. *Proc. Fifth Berkeley Sympos. Math. Statist. Probab.*, University of California Press, Berkeley, CA, 281–297.

Ostrovsky, R., Y. Rabani. 2002. Polynomial time approximation schemes for geometric k-clustering. *J. Assoc. Comput. Mach.* **49** 139–156.

Patil, N., A. J. Berno, D. A. Hinds, W. A. Barrett, J. M. Doshi, C. R. Hacker, C. R. Kautzer, D. H. Lee, C. Marjoribanks, D. P. McDonough, B. T. Nguyen, M. C. Norris, J. B. Sheehan, N. Shen, D. Stern, R. P. Stokowski, D. J. Thomas, M. O. Trulson, K. R. Vyas, K. A. Frazer, S. P. Fodor, D. R. Cox. 2001. Blocks of limited haplotype diversity revealed by high-resolution scanning of human chromosome 21. *Science* **294** 1719–1723.

Sachidanandam, R., D. Weissman, S. C. Schmidt, J. M. Kakol, L. D. Stein, G. Marth, S. Sherry, J. C. Mullikin, B. J. Mortimore, D. L. Willey, S. E. Hunt, C. G. Cole, P. C. Coggill, C. M. Rice, Z. Ning, J. Rogers, D. R. Bentley, P. Y. Kwok, E. R. Mardis, R. T. Yeh, B. Schultz, L. Cook, R. Davenport, M. Dante, L. Fulton, L. Hillier, R. H. Waterston, J. D. McPherson, B. Gilman, S. Schaffner, W. J. Van Etten, D. Reich, J. Higgins, M. J. Daly, B. Blumenstiel, J. Baldwin, N. Stange-Thomann, M. C. Zody, L. Linton, E. S. Lander, D. Altshuler. 2001. A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms. *Nature* **291** 1298–2302.

Venter, J. Craig, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt, J. D. Gocayne et al. 2001. The sequence of the human genome. *Science* **291** 1304–1351.

Waterman, M. S. 1995. *Introduction to Computational Biology: Maps, Sequences and Genomes.* Chapman and Hall.

Zhang, K., M. Deng, T. Chen, M. S. Waterman, F. Sun. 2002. A dynamic programming algorithm for haplotype block partitioning. *Proc. National Acad. Sci. USA* **99** 7335–7339.

Journal of Bioinformatics and Computational Biology
Vol. 3, No. 2 (2005) 359–384

Imperial College Press
www.icpress.co.uk

THE INCOMPLETE PERFECT PHYLOGENY HAPLOTYPE PROBLEM

GAD KIMMEL
School of Computer Science, Tel Aviv University
Tel Aviv 69978, Israel
kgad@tau.ac.il

RON SHAMIR
School of Computer Science, Tel Aviv University
Tel Aviv 69978, Israel
rshamir@tau.ac.il

Received 11 May 2004
Revised 21 August 2004
Accepted 30 August 2004

The problem of resolving genotypes into haplotypes, under the perfect phylogeny model, has been under intensive study recently. All studies so far handled missing data entries in a heuristic manner. We prove that the perfect phylogeny haplotype problem is NP-complete when some of the data entries are missing, even when the phylogeny is rooted. We define a biologically motivated probabilistic model for genotype generation and for the way missing data occur. Under this model, we provide an algorithm, which takes an expected polynomial time. In tests on simulated data, our algorithm quickly resolves the genotypes under high rates of missing entries.

Keywords: Haplotype; haplotype block; genotype; SNP; algorithm; complexity; genotype phasing; haplotype resolution; perfect phylogeny.

1. Introduction

A central current challenge in human genome research is to learn about DNA differences among individuals. This knowledge will hopefully lead to finding the genetic causes of complex and multi-factorial diseases. The distinct single-base sites along the DNA sequence that show variability in their nucleic acids contents across the population, are called *single nucleotide polymorphisms* (SNPs). Millions of SNPs have already been detected,[22] and it is estimated that the total number of common SNPs is 10 million.[18]

In diploid organisms (e.g. humans), there are two nearly-identical copies of each chromosome. Most techniques for determining SNPs provide a pair of readings, one from each copy. However, they cannot identify which of the two chromosomes each reading came from.[14] The goal of *phasing* (or *resolving*) is to infer that missing

information. The original conflated data from both chromosomes are called the *genotype* of the individual, and is represented by a set of two nucleotide readings for each site. The two separated sequences corresponding to the two chromosomes of an individual are called his/her *haplotypes*. If the two bases in a site are identical (resp., different), the site is called *homozygous* (resp., *heterozygous*). For recent reviews on biological and computational aspects of haplotype analysis, see Halldorsson *et al.*[11] and Hoehe *et al.*[14]

Resolving the genotypes is a central problem in haplotyping. It has been argued that more accurate association studies can be performed once the genotypes are resolved.[4,15] In the absence of additional information, each genotype can be resolved in 2^{h-1} different ways, where h is the number of heterozygous sites in the genotype. To find the correct way, resolution is done simultaneously on all the available genotypes, and according to a model. A pioneering approach to haplotype resolution was Clark's parsimony-based algorithm.[3] A likelihood-based EM algorithm[6,19] gave better results. Stephens *et al.*[24] and Niu *et al.*[20] proposed MCMC-based methods which gave promising results. All of these methods assumed that the genotype data correspond to a single block with no recombination events. Hence, for multi-block data the block structure must be determined separately.

Recently, a new combinatorial formulation of the phasing problem was suggested by Gusfield.[10] According to this model, phasing must be done so that the resulting haplotypes define a *perfect phylogeny tree*. This model assumes that in the studied region along the chromosome, recombination did not occur, and the infinite site assumption holds.[10] Gusfield showed how to solve the problem efficiently, and simpler algorithms were subsequently developed by Bafna *et al.*[2] and Eskin *et al.*[5] Eskin *et al.*[5] showed good resolving results with small error rates on real genotypes. They also reported that their algorithm was faster and more accurate in practical settings than the method by Stephens *et al.*[24]

In real genotype data (e.g., Daly *et al.*,[4] Gabriel *et al.*[7] and Patil *et al.*[21]) some of the data entries are often missing, due to technical causes. Current phasing algorithms that are based on perfect phylogeny require complete genotypes. This situation raises the following algorithmic problem: Complete the missing entries in the genotypes and then resolve the data, such that the resulting haplotypes define a perfect phylogeny tree. We call this problem *incomplete perfect phylogeny haplotype* (IPPH). It was posed by Halldorsson *et al.*[11] In order to deal with such incomplete data, Eskin *et al.*[5] used a heuristic to complete the missing entries, and showed very good results. However, having an algorithm for optimally handling missing data entries should allow more accurate resolution. In this paper we address the IPPH problem.

A special case of IPPH was studied in phylogeny by Pe'er *et al.*[16] In the *incomplete directed perfect phylogeny* problem, the input is an $n \times m$ species-characters matrix. The characters are binary and directed, i.e., a species can only gain characters, and certain characters are missing in some species. The question is whether one

can complete the missing states in a way admitting a perfect phylogeny. Pe'er *et al.* provided a near optimal $\tilde{O}(nm)$ time algorithm for the problem.[a] This problem is a special case of IPPH in which all the sites in all genotypes are homozygous, and the root is known.

The IPPH problem has two variants: *rooted* (or *directed*) and *unrooted* (or general). In the rooted version, one haplotype is given as part of the input. This haplotype is referred to as the root of the tree, even though it may not be the real evolutionary root of the tree. This holds since each of the haplotypes can be used as a root in the perfect phylogeny tree.[9] The unrooted version is a more direct formulation of the practice in biology, since in phasing, the root of the haplotypes is not given. However, we argue that the more restricted rooted version is of practical importance: Though theoretically finding a root might take an exponential time, in practice, there is often one genotype that is both complete and homozygous in all sites, which can be used as a root. As we shall demonstrate in Sec. 5 on simulated and real biological data, virtually always at least one such genotype exists. If there is no such genotype, one can use a genotype with few undetermined sites and enumerate the values in these sites. In the rare cases that this too is not feasible, one can physically separate the two chromosomes of a single individual and sequence one haplotype, as was done in Patil *et al.*[21] This procedure is considerably more expensive than standard genotyping techniques, but it will be performed only for one individual, so the price is small. Thus, both variants of IPPH are biologically important.

In this paper, we show that rooted IPPH is NP-complete. The hardness of unrooted IPPH follows immediately from the hardness of determining the compatibility of unrooted partial binary characters (incomplete haplotype matrix).[23] This was observed first by R. Sharan (private communication). However, this result does not imply the hardness of rooted version. In fact, our proof for rooted IPPH is quite involved.

To cope with the theoretical hardness of IPPH, we invoke a probabilistic approach. We define a stochastic model for generating the haplotypes and for the way missing entries occur in them. The model assumptions are mild and seem to hold for biological data. In addition, we assume that the number of sites m grows much more slowly than the number of genotypes n. Specifically, we assume that $m = O(n^5)$. As m is bounded by the block size which in practice is not more than a modest constant (10–30), this condition also holds in practice. We design an algorithm that always finds the correct solution, and under the assumptions above takes an expected time of $\tilde{O}(m^2 n)$. A similar probabilistic approach leading to comparable results was developed simultaneously and independently by Halperin and Karp.[12]

To test our algorithm, we applied it to simulated data using biologically realistic values of the parameters, and calculated an upper bound Γ on the main factor in the

[a]We use $\tilde{O}$ notation to suppress polylogarithmic factors in presenting complexity bounds. Formally, $\tilde{O}(g(n)) := \{f(n) \mid \exists n_0 > 0, \exists c > 0, \exists d > 0, \forall n \geq n_0 : \ 0 \leq f(n) \leq c[\log n]^d g(n)\}$.

running time. Γm gives a bound on the number of times the polynomial algorithm by Pe'er *et al.*[16] would be invoked to complete the calculation. Γ may be exponential, but under the model assumptions it was shown to have an expected polynomial time. On data with 200 genotypes and 30 sites, we show that on average $\Gamma < 4000$ even when only two haplotypes are present and the rate of missing entries is 50%. For a more realistic case of five haplotypes and 20% missing entries, $\mathbb{E}[\Gamma] < 100$. Hence, the algorithm runs in modest time even far beyond the range of its provable performance.

The paper is organized as follows: Sec. 2 presents definitions and preliminaries. Section 3 shows the hardness result. Section 4 presents the algorithm and the probabilistic analysis, and finally Sec. 5 summarizes our experimental results.

A preliminary version of this study was published in the *Proceedings of the Second RECOMB Satellite Meeting on SNPs and Haplotypes*.[17]

2. Preliminaries

In this section we provide basic definitions, lemmas and observations that are needed for our analysis. Figure 1 demonstrates the main definitions.

Given n genotypes, the haplotype inference problem is to find n pairs of haplotypes vectors that could have generated the genotypes vectors. Formally, the input is an $n \times m$ *genotype matrix* M, with $M[i,j] \in \{0,1,2\}$. The ith row $M[i,*]$ describes the ith genotype. The jth column describes the alleles in the jth location: 0 or 1 for two homozygous alleles, and 2 for a heterozygous site. A $2n \times m$ binary matrix M' is an *expansion* of the genotype matrix M if each row $M[i,*]$ expands to two rows denoted by $M'[i,*]$ and $M'[i',*]$, with $i' = n + i$, satisfying the following: for every i, if $M[i,j] \in \{0,1\}$, then $M[i,j] = M'[i,j] = M'[i',j]$; if $M[i,j] = 2$, then $M'[i,j] \neq M'[i',j]$. M' is also called a *haplotype matrix* corresponding to M.

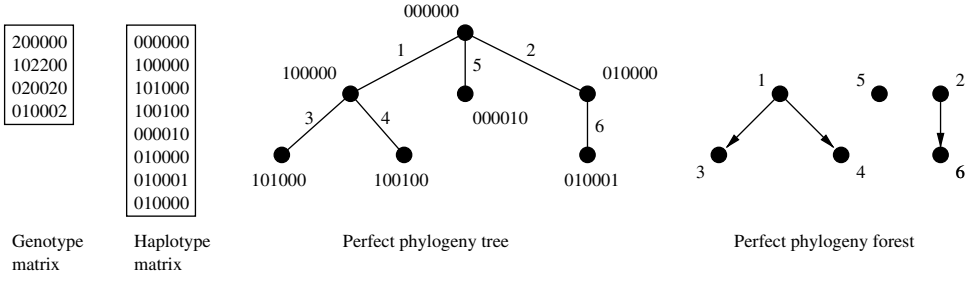


Fig. 1. Genotypes, haplotypes and trees. A genotype matrix M, a haplotype matrix M' that is an expansion of M, the perfect phylogeny tree of M', and the corresponding perfect phylogeny forest.

Definition 1. Perfect Phylogeny Tree for a Matrix
A *perfect phylogeny* for a $k \times m$ haplotype matrix M' is a tree T with a root r, exactly k leaves and integer edge labels, and a binary label vector $(l^v(1) \cdots l^v(m))$ for each node v, that obeys the following properties:

1. Each of the rows in M' is the label of exactly one leaf of T.
2. Each of the columns labels exactly one edge of T.
3. Every edge of T is labeled by one column.
4. For any node v, $l^v(i) \neq l^r(i)$ if and only if i labels an edge on the unique path from the root to v. Hence, given the root label, the root-node paths provide a compact representation of all node labels.

An equivalent definition appeared in Bafna *et al.*[2] Note that we disallow edges with multiple labels, and replace them by paths with a single label per edge.

Problem 1. The Perfect Phylogeny Haplotype Problem (PPH)[10]
Given a matrix M, find an expansion M' of M which admits a perfect phylogeny.

We now define a generalization of PPH that allows missing data entries. The input to our problem is an *incomplete genotype matrix*, i.e., a matrix M with $M[i, j] \in \{0, 1, 2, ?\}$, where "?" indicates a missing data entry. The process of replacing each "?" by 0,1 or 2 is called *completing* the matrix M.

Problem 2. Incomplete Perfect Phylogeny Haplotype Problem (IPPH)
Given an incomplete genotype matrix M, can one complete M, so that there exists an expansion M' of M, which admits a perfect phylogeny?

The following definitions are implicit in Bafna *et al.*[2] and Eskin *et al.*[5]

Definition 2. Perfect Phylogeny Forest
Let M be a haplotype matrix, and let $P = (V_P, E_P)$ be a perfect phylogeny tree corresponding to M. The *perfect phylogeny forest* of P is a directed forest $F = (V_F, E_F)$ whose vertices are the edges of P, and for $u, v \in V_F$, u is a parent of v in F if and only if the edge corresponding to u in P is a parent of the edge corresponding to v in P.

Hence, the vertices of perfect phylogeny forest correspond to M''s columns, and reflect the order of mutations in the phylogeny tree. Clearly, every perfect phylogeny tree can be converted into a perfect phylogeny forest and vice versa. Thus, M' admits a perfect phylogeny tree iff it admits a perfect phylogeny forest. For a column $j \in \{1, 2, \ldots, m\}$ of M', we denote by u_j its corresponding vertex in the perfect phylogeny forest.

For a perfect phylogeny forest F, we say that two vertices are in *parenthood relation* if one is an ancestor of the other. Otherwise, we say that they are in *brotherhood relation*. Note that brothers can either be in different connected components, or be in the same component and have the root on the path connecting them.

The following special case of IPPH will be a main subject of our investigation.

Problem 3. Incomplete Perfect Phylogeny Haplotype, rooted version (ROOTED-IPPH)
Given an incomplete genotype matrix M and a haplotype r, can one complete M, such that there exists an expansion M' of M, which admits an perfect phylogeny, with r as a root?

In this problem, we can assume w.l.o.g. that the input root haplotype is $r_0 = (0, \ldots, 0)$.[9] The following lemma explains the connection between F and M', and is key for our construction:

Lemma 1. (*Bafna et al.*,[2] *Eskin et al.*[5]) *Let M' be a haplotype matrix, and let $F = (V_F, E_F)$ be a perfect phylogeny forest, corresponding to a perfect phylogeny tree with the root $r_0 = (0, 0, \ldots, 0)$. F is a perfect phylogeny forest of M' iff for all $u_a, u_b \in V_F$ and for every haplotypei* :

(1) *If u_a is an ancestor of u_b then $M'[i, a] = 1$ or $M'[i, b] = 0$.*
(2) *If u_a and v_b are in brotherhood relation, then $M'[i, a] = 0$ or $M'[i, b] = 0$.*

In the rest of this section, we provide our own definitions, building on those introduced above, and prove several lemmas which will be needed for our analysis.

Definition 3. Constrained Mixed Graph
A *constrained mixed graph* (CMG) is a triplet $G_c = (V, E, X)$, where $G = (V, E)$ is a graph and $X = \{X_1, X_2, \ldots, X_p\}$, where for each i: $X_i \subseteq V$. The sets X_i are called *XOR relations*. G has four types of edges: undirected, dashed undirected, directed and dashed directed.

Definition 4. Parenthood Connected Components
Two vertices u and v in a constrained mixed graph are in the same *parenthood connected component* if there exists a path between u and v consisting only of undirected or directed edges (a parenthood relation). Note that edge directions are not important in this definition.

Definition 5. Constrained Mixed Completion Graph
For a constrained mixed graph $G_c = (V, E, X)$, we define its *constrained mixed completion graph* $G' = (V, E')$ to be a complete graph (with a single edge for each pair $u, v \in E$), where E' contains two types of edges: directed and dashed undirected. The edge types induce a labeling L: $E' \to \{0, 1\}$, where a directed edge is labeled with 0, and dashed undirected edge is labeled with 1. G' must maintain all the following properties:

(1) All G' edges maintain the following properties:
   (a) If $e : (u, v) \in E$ is an undirected edge then E' must contain a directed edge from u to v or from v to u.
   (b) Directed edges and dashed undirected edges in G are unchanged in G'.
   (c) If $e : (u, v) \in E$ is a dashed directed edge from u to v then the corresponding $e' : (u, v) \in E'$ must be a dashed undirected edge or a directed edge from u to v.

(2) G' contains a spanning forest $F = (V, E_F \subseteq E')$, consisting of directed edges only, such that:

   (a) If node $u \in V$ is an ancestor of $v \in V$ in F, then there is a directed edge from u to v in G'.
   (b) For any two nodes in V, if neither node is an ancestor of the other in F, then they are connected by a dashed undirected edge in G'.

(3) For each XOR relation X_i, for every three vertices: $x_{i,a}, x_{i,b}, x_{i,c} \in X_i$, the following holds: $L(x_{i,a}, x_{i,b}) \oplus L(x_{i,b}, x_{i,c}) \oplus L(x_{i,a}, x_{i,c}) = 0$.[b]

Problem 4. Constrained Mixed Graph Completion Problem (CMGC)
Given a constrained mixed graph G, provide a constrained mixed completion graph of G, if such a graph exists.

An example of the CMGC problem is presented in Fig. 2. The decision version of the CMGC problem is to decide whether there exists a constrained mixed completion graph G' for G. An important property of the constrained mixed completion graph, is that it can be viewed as a directed spanning forest F, with additional edges between nodes, according to the relation of those nodes in the forest: a dashed undirected edge for a brotherhood relation, and a directed edge for a parenthood relation.

The following notation is adopted from Eskin *et al.*:[5] $c(M, x)$ is defined as the set of rows of M containing the value x in column c. Let c, c' be columns and let x, y be elements of $\{0, 1\}$. The pair c, c' *induces* (x, y) in M if $((c(M, x) \cap c'(M, y)) \cup (c(M, x) \cap c'(M, 2)) \cup (c(M, 2) \cap c'(M, y)) \neq \emptyset$. Let $R(M, c, c')$ be the set of pairs (x, y) such that (c, c') induces (x, y) in M. Note that $R(M, c, c')$ does not contain pairs with "?", but only "0" and "1". Observe that from our assumption that the root is $(0, \ldots, 0)$, it follows that $R(M, c, c')$ contains (0,0) for every c, c'.

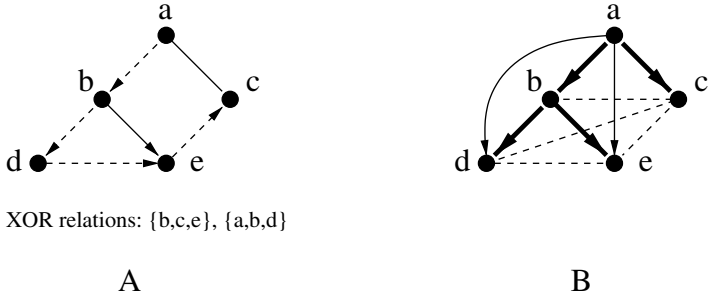


Fig. 2. Example of CMGC problem. A: an instance of a graph for CMGC with XOR relations. B: a possible solution for this instance. The edges of the forest appear in bold.

[b]The operator $\oplus$ denotes the boolean xor operator.

Let c, c' be two columns such that $c(M,2) \cap c'(M,2) \neq \emptyset$. Let M' be an expansion of the M, after completing the missing entries, which admits a perfect phylogeny. We say that M' resolves the pair of columns (c, c') *unequally* if $\{(0,0),(0,1),(1,0)\} = R(M', c, c')$ and *equally* if $\{(0,0),(1,1)\} = R(M', c, c')$. According to Lemma 1, M' must resolve the pair (c, c') either equally or unequally, and can not resolve the pair in both ways.

For an incomplete genotype matrix M, we build a constrained mixed graph $G_c(M)$, where each column in M has a corresponding vertex in G_c. The edges represent the possible relations of the columns in the perfect phylogeny forest, and are determined according to lemma 1: For each two vertices u_a, u_b:

(1) If $R(M,a,b)\backslash\{(0,0)\} = \{(1,1),(1,0)\}$ then u_a is an ancestor of u_b in F. The edge (u_a, u_b) is set as a directed edge from u_a to u_b.
(2) If $R(M,a,b)\backslash\{(0,0)\} = \{(1,1)\}$ then u_a, u_b are in parenthood relation in F, but it is unknown which of the vertices is the ancestor. The edge (u_a, u_b) is set as an undirected edge.
(3) If $R(M,a,b)\backslash\{(0,0)\} = \{(1,0),(0,1)\}$ then u_a, u_b are in brotherhood relation in F. The edge (u_a, u_b) is set as a dashed undirected edge.
(4) If $R(M,a,b)\backslash\{(0,0)\} = \{(1,0)\}$ then either u_a is an ancestor of u_b in F, or u_a, u_b are in brotherhood relation in F. The edge (u_a, u_b) is set as a dashed directed edge from u_a to u_b.
(5) If $R(M,a,b)\backslash\{(0,0)\} = \emptyset$ then the relation of u_a, u_b in F is unknown. In that case: $(u_a, u_b) \notin E$. The labeling of unlabeled edges corresponds to selecting the type of edge in the completion of G_c for solid undirected and for dashed directed edges.

In addition, for each row i, the set of columns $a_1, \ldots, a_t$, such that $M[i,a_1] = \cdots = M[i,a_t] = 2$, imply a XOR relation on the corresponding vertices $u_{a_1}, \ldots, u_{a_t}$. Each pair of vertices of G_c is labeled with $L : (u_a, u_b) \to \{0,1,?\}$ as follows: A solid (directed or undirected) edge, i.e., a parenthood relation, is labeled with 0; dashed undirected edge, i.e., a brotherhood relation, is labeled with 1; and all other cases, i.e., an unknown relation, are labeled with "?". The last set is called *unlabeled pairs*.

Step 1: Primary Label Completion
A *primary label completion of* $G_c(M)$ is an assignment of a label s to unlabeled pairs of vertices, by performing the following step as long as possible: Find three vertices $x_{i,a}, x_{i,b}, x_{i,c} \in X_i$, such that $L(x_{i,a}, x_{i,b})$ and $L(x_{i,b}, x_{i,c})$ are set and $L(x_{i,a}, x_{i,c})$ is not, and assign: $L(x_{i,a}, x_{i,c}) = L(x_{i,a}, x_{i,b}) \oplus L(x_{i,b}, x_{i,c})$.

Define U_{G_c} to be the set of unlabeled pairs after primary label completion was performed. U_{G_c} is independent of the order of choosing the triplets.[2]

Step 2: Secondary Label Completion A *secondary label completion of a constrained mixed graph* $G_c(M)$ is an assignment of a label in {0,1} to pairs

$(u_a, u_b) \in U_{G_c}$, such that for each XOR relation X_i, for every three vertices $x_{i,a}, x_{i,b}, x_{i,c} \in X_i$, the condition: $L(x_{i,a}, x_{i,b}) \oplus L(x_{i,b}, x_{i,c}) \oplus L(x_{i,a}, x_{i,c}) = 0$ is satisfied.

After the secondary label completion, we can perform a label resolution using the incomplete genotype matrix, which is defined as follows: Given an incomplete genotype matrix M, an *expansion* for M is in an incomplete haplotype matrix M' which satisfies the expansion rules for complete matrices, and also preserves all "?" values. Formally, each row $M[i, *]$ expands to two rows denoted by $M'[i, *]$ and $M'[i', *]$, such that for every i, if $M[i, j] \in \{0, 1, ?\}$, then $M[i, j] = M'[i, j] = M'[i', j]$; if $M[i, j] = 2$, then either $M'[i, j] = 0$ and $M'[i', j] = 1$, or $M'[i, j] = 1$ and $M'[i', j] = 0$.

Step 3: Label Resolution

A *label resolution* of a genotype matrix M is an expansion of M to an incomplete haplotype matrix M', according to the label function L: For every two columns a, b, if there exists i, such that $M[i, a] = M[i, b] = 2$, if $L(u_a, u_b) = 0$ resolve (a, b) equally and if $L(u_a, u_b) = 1$ resolve (a, b) unequally. The output of this process is an incomplete haplotype

Label resolution of an incomplete genotype matrix can be done by algorithm $\mathcal{E}2M$ proposed by Bafna *et al.*[2] Observe that any submatrix $M[i, (a, b)]$, where $M[i, a]$ and $M[i, b]$ are not both equal to 2, has a unique expansion in any incomplete haplotype matrix. Hence, for such a submatrix, the resolution is not influenced by the label function.

Primary label completion was suggested by Bafna *et al.*[2] as part of an algorithm for complete genotype matrix phasing. Interestingly, Bafna *et al.* proved that once primary label completion is performed, for any possible (legal) secondary label completion of U_{G_c}, label resolution of the genotype matrix results in a haplotype matrix, which admits a perfect phylogeny. This is true for a complete genotype matrix (with no missing entries), but not for the incomplete case. A simple example for that is an incomplete haplotype matrix that does not admit a perfect phylogeny (see e.g., Pe'er *et al.*,[16] Fig. 2). Now consider such a matrix to be the input genotype matrix, by duplicating each haplotype to form a fully homozygous genotype. Here, no primary and secondary resolution is needed, since there are no heterozygous sites in the matrix. Thus, every secondary resolution results in an incomplete haplotype matrix (namely, the same input matrix), which does not admit a perfect phylogeny. The following lemma describes a weaker connection between secondary label completion and the solution of IPPH.

Lemma 2. *Suppose M is an incomplete genotype matrix that has a completion to a genotype matrix M^*. Suppose further that M^* has an expansion M' that admits a perfect phylogeny. Then there exists some secondary label completion of U_{G_c}, such that a label resolution of the incomplete genotype matrix M gives an incomplete haplotype matrix, that can be completed to M'.*

Proof. Let C be the set of columns of M. The perfect phylogeny implies some label function f_L on the pairs of the vertices of $G_c(M)$, i.e., $\forall i, j \in C$: $f_L(u_i, u_j) \in \{0, 1\}$. This complete label function can not contradict the XOR relations of $G_c(M)$ (for proof, see Bafna *et al.*[2]). Next, primary label completion of $G_c(M)$, for the known pairs, must match the labels in f_L, as there is only one possible primary label completion. Then, we can chose the following secondary label completion: $(u_i, u_j) \in U_{G_c}$: $L(u_i, u_j) = f_L(u_i, u_j)$, which obviously gives an equivalent label function to f_L. Thus, using this secondary label completion of M, a label resolution of the incomplete genotype matrix M gives an incomplete haplotype matrix, that can be completed to M'. □

3. The Hardness Result

In this section, we show that ROOTED-IPPH is NP-complete. Clearly, the problem belongs to NP. To prove NP-hardness, we will show the following polynomial reductions: 3-SAT $\propto$ CMGC $\propto$ ROOTED-IPPH.

We note that this also implies an alternative proof of the hardness of unrooted IPPH: to form the reduction ROOTED-IPPH $\propto$ IPPH, given an instance (M, r) of ROOTED-IPPH, we simply add the genotype row r to M. The resulting matrix M^* is the input to IPPH. In a solution to the latter, there will be a leaf labeled with r, and thus it solves the former problem. Conversely, if M has a solution with root r then it is also a solution for M^*. The same idea was used for another purpose by Bafna *et al.*[2]

We first prove the reduction from CMGC:

Theorem 1. *CMGC* $\propto$ ROOTED-IPPH

Proof. Given a constrained mixed graph $G_c = (V, E, X)$ for the CMGC problem, we build a matrix M, and set $r = (0, 0, \ldots, 0)$. (M, r) will serve as input for ROOTED-IPPH. Let $|X| = p$. M has dimensions $(2|E|+p)\times|V|$. For each $e \in E$ there are two corresponding rows, and their indices are denoted by N_e^0 and N_e^1. For each $X_i \in \{X_i\}_{1\le i\le p}$ there is one row with index N_{X_i}. The column $i \in \{1, 2, \ldots, |V|\}$ corresponds to vertex u_i in G_c.

The construction of M is as follows:

(1) For each $e = (u_a, u_b) \in E$, we add two rows $M[N_e^0, *]$ and $M[N_e^1, *]$, such that $\forall u_c \in V\backslash\{u_a, u_b\}$, $M[N_e^0, c] = M[N_e^1, c] =?$, and:
   (a) If e is an undirected edge then $M[N_e^0, a] = 0,\ M[N_e^0, b] = 0,\ M[N_e^1, a] = 1,\ M[N_e^1, b] = 1$.
   (b) If e is a dashed undirected edge then $M[N_e^0, a] = 0, M[N_e^0, b] = 1,\ M[N_e^1, a] = 1,\ M[N_e^1, b] = 0$.
   (c) If e is a directed edge from u_a to u_b then $M[N_e^0, a] = 1,\ M[N_e^0, b] = 0,\ M[N_e^1, a] = 1,\ M[N_e^1, b] = 1$.

(d) If e is a dashed directed edge from u_a to u_b then $M[N_e^0, a] = 0,\ M[N_e^0, b] = 0,$ $M[N_e^1, a] = 1,\ M[N_e^1, b] = 0.$

(2) For each $\{X_i\}_{1 \le i \le p}$, we add one row $M[N_{X_i}, *]$, such that $\forall u_j \in X_i$: $M[N_{X_i}, j] = 2$ and $\forall u_k \in V \backslash X_i$: $M[N_{X_i}, k] =?$. □

This completes the description of the reduction. Clearly the reduction is polynomial.

(⇒)

Suppose that ROOTED-IPPH(M,r) = TRUE, i.e., M has an expansion M' that admits a perfect phylogeny tree, with r_0 as a root. Thus, M' has a directed perfect phylogeny forest $F = (V_F, E_F)$. Let $\hat{F} = (V_F, \hat{E}_F)$ be a complete graph, where for each $u, v \in V_F$, we add a directed edge from u to v if u is an ancestor of v in F, or a dashed undirected edge if neither node is an ancestor of the other.

We claim that $\hat{F}$ is a constrained mixed completion graph of G_c. This is proven by checking that all three properties of $\hat{F}$ as a constrained mixed completion graph of graph G_c hold (compare Definition 5). Property 2 holds since by the construction of $\hat{F}$, F is a rooted spanning forest of $\hat{F}$ as required. In order to prove property 3, we use Lemma 2 in Bafna *et al.*[2]: the structure of the rows $\{M[N_{X_i}, *]\}_{1 \le i \le p}$ forces that for each of the XOR relations, for every three vertices $x_{i,a}, x_{i,b}, x_{i,c} \in (X_i \subseteq V_F)$, the equation $L(x_{i,a}, x_{i,b}) \oplus L(x_{i,b}, x_{i,c}) \oplus L(x_{i,a}, x_{i,c}) = 0$ holds. Finally, property 1 holds, since for an edge $e \in E$, the values in the two corresponding rows $\{M[N_e^j, *]\}_{j \in \{0,1\}}$ are determined in step 1 of the construction of M: The edge (u, v) in graph G_c determines the possible relations of u and v in F. Since, by the assumption, M has an expansion M', that admits a perfect phylogeny forest F, it follows that for each $u, v \in F$, the edge $e' = (u, v) \in E_F$ must be set according to $e = (u, v) \in E$ in G_c: If e is an undirected edge then e' must be a directed edge; if e is a dashed undirected edge then e' must be a dashed undirected edge; if e is a directed edge from u to v then e' must be a directed edge from u to v; and if e is a dashed directed edge from u to v then e' must be a dashed undirected edge or a directed edge from u to v. This proves property 1. Thus, $\hat{F}$ is the constrained mixed completion graph of G_c, and CMGC(G_c) = TRUE.

(⇐)

Suppose that CMGC (G_c) = TRUE, i.e., there exists a constrained mixed completion graph G' for G_c. According to the second property of G', there exists a directed forest $F = (E_F, V)$, which spans V. Due to the third property, the completion of edges in G_c, does not violate the XOR relations. We create an expansion M' of M as follows: resolve the "2" of the genotypes in those rows, according to G': for two vertices $\{u_a, u_b \in X_i\}_{1 \le i \le p}$, in case $M[N_{X_i}, a] = M[N_{X_i}, b] = 2$, if there is an undirected dashed edge between $u_a, u_b \in V$, then resolve the pair of columns (a, b) unequally, and if there is an directed edge between $u_a, u_b \in V$, then resolve the

submatrix equally. Since those edges are completed in G' according to XOR relations (see Definition 5, property 3), each of the "2"s in these rows can be resolved accordingly.

We denote the remaining $2|E|\times|V|$ matrix by M^*. Note that $M^*[i,j] \in \{0,1,?\}$. We call the $\{0,1\}$ entries *"constants"*, and the "?" entries *"variables"*. We denote the set of column indices of constants in row i by C_i, and the set of column indices of variables in this row by V_i. Complete the variables entries in the matrix M^* to create a matrix M^{**} as follows:

$$M^{**}[i,j]_{j\in V_i} = \begin{cases} 1 & \text{if } \exists c \in C_i \text{ s.t. } M^*[i,c] = 1 \wedge u_j \text{ is an ancestor of } u_c \\ 0 & \text{otherwise} \end{cases}$$

M^{**} is a binary matrix and an expansion of M. We claim that M^{**} admits a perfect phylogeny forest. Moreover, this forest is F. This will be proven by showing that the relation of each two columns in M^{**} does not contradict the relation of their corresponding nodes in F. Thus, according to Lemma 1, F is a perfect phylogeny forest of M^{**}.

Consider two vertices $u_a, u_b \in V$ and their corresponding columns a, b in M^{**}. For each row i, we examine the three possible cases for M^{**}:

(1) $u_a, u_b \in C_i$

$M(i,a)$ and $M(i,b)$ were set according to the edge $(u_a, u_b) \in E$, which by definition of G', does not contradict F.

(2) $u_a \in C_i, u_b \in V_i$

First, suppose $M^{**}[i,a] = 0$: If $M^{**}[i,b]$ is set to 0, then there is no contradiction for any relations of u_a and u_b in F. Otherwise, if $M^{**}[i,b]$ is set to 1, then there exists $c \in C_i$ such that $M'[i,c] = 1$ and u_b is an ancestor of u_c. Suppose, on the contrary, that u_a, u_b contradict F in row i. This means, that there are two rows j,k such that $M^*[j,a] = 1$, $M^*[j,b] = 0$, $M^*[k,a] = 1$, $M^*[k,b] = 1$, i.e., according to these rows, u_a is an ancestor of u_b in F. Since u_b is an ancestor of u_c, u_a must be an ancestor of u_c. However, according to the construction of M, u_a cannot be an ancestor of u_c, since $M^{**}[i,a] = 0$ and $M^{**}[i,c] = 1$ and $a, c \in C_i$.

Second, suppose $M^{**}[i,a] = 1$: If $M^{**}[i,b]$ is set to 0, clearly u_b is not an ancestor of u_a, so M^{**} does not contradict F. Otherwise, if $M^{**}[i,b]$ is set to 1, then there exists $c \in C_i$, such that $M^{**}[i,c] = 1$ and u_b is an ancestor of u_c. In case $c = a$, u_a and u_b can not be in a brotherhood relation. In case $c \neq a$, u_a and u_c are in parenthood relation, and since u_b is an ancestor of u_c, it follows that u_a and u_b cannot be in a brotherhood relation.

It follows that, in this case, M^{**} does not contradict F.

(3) $u_a, u_b \in V_i$

First, suppose that $M'[i,a]$ and $M'[i,b]$ are both set to 0. Obviously, the submatrix does not contradict F.

Second, suppose w.l.o.g. that $M'[i,a]$ is set to 0 and $M'[i,b]$ is set to 1. There exists $c \in C_i, c \neq a$ such that $M'[i,c] = 1$ and u_b is an ancestor of u_c.

Suppose, on the contrary, that u_a, u_b contradict F in row i. This means, that there are two rows j, k such that $M^*[j,a] = 1$, $M^*[j,b] = 0$, $M^*[k,a] = 1$, $M^*[k,b] = 1$, i.e., according to these rows, u_a is an ancestor of u_b in F. Since u_b is an ancestor of u_c, u_a must be an ancestor of u_c. However, in that case, $M'[i,a]$ should have been set to 1.

Third, suppose that $M'[i,a]$ and $M'[i,b]$ are both set to 1. There exist $c_a, c_b \in C_i$ such that $M'[i,c_a] = 1$, $M'[i,c_b] = 1$ and u_a is an ancestor of u_{c_a} and u_b is an ancestor of u_{c_b}. Clearly, u_{c_a} and u_{c_b} are in parenthood relation, so w.l.o.g. suppose that u_{c_a} is an ancestor of u_{c_b}. Thus, both u_a and u_b are ancestors of u_{c_b}, and it follows that u_a and u_b cannot be in brotherhood relation.

It follows that, in this case, M^{**} does not contradict F.

Theorem 2. 3-SAT $\propto$ CMGC.

Proof. For a 3-SAT instance Φ we build a CMGC graph G_c. Denote the variables of Φ by $\{Y_i\}_{1\le i\le t}$ and the clauses by $\{C_j\}_{1\le j\le s}$. Our construction will be formed from four types of CMG sub-instances. First we define these four graph structures:

Variable base graph contains two vertices denoted by x_0^i and x_1^i, with no edge between them. This graph is denoted by Var_i.

Clause base graph (see Fig. 3) contains 6 vertices denoted by $\{c_t^j\}_{0\le t\le 5}$. The edges are indicated in Fig. 3. This graph is denoted by Cl_j.

Positive variable connector (see Fig. 3) contains 12 vertices denoted by $\{a_t^{i,j}\}_{0\le t\le 5}$, and $\{b_t^{i,j}\}_{0\le t\le 5}$. The edges are indicated in Fig. 3. This graph is denoted by Pos.

Negative variable connector (see Fig. 3) contains 8 vertices denoted by $\{d_t^{i,j}\}_{0\le t\le 3}$ and $\{e_t^{i,j}\}_{0\le t\le 3}$. The edges are indicated in Fig. 3. This graph is denoted by Neg.

The XOR relations constrain the ways to complete the variable connectors. In fact, that there are two possible ways to complete the positive variable connector and the negative variable connector with undirected edges. Both of the ways for both types of connectors are presented in Fig. 4. An important key to understanding the reduction, is that in the positive connector, the type (dashed or non-dashed) of edge $(a_0^{i,j}, b_0^{i,j})$ is the same as the type of the edge $(a_5^{i,j}, b_5^{i,j})$. In the negative connector, the type of edge $(d_0^{i,j}, e_0^{i,j})$ is the opposite from the edge $(d_3^{i,j}, e_3^{i,j})$. These two types will play the role of True and False in the reduction.

The construction of G_c is done as follows:

1. For each variable $\{Y_i\}_{1\le i\le t}$ create a copy of a variable base graph Var_i.
2. For each clause $\{C_j\}_{1\le j\le s}$ create a copy of a clause base graph Cl_j.
3. For all $1 \le j \le s$, for all $1 \le k \le 3$ do:
4. if Y_i is the kth literal in clause C_j then do:
   create a copy of positive variable connector with superscripts i, j.

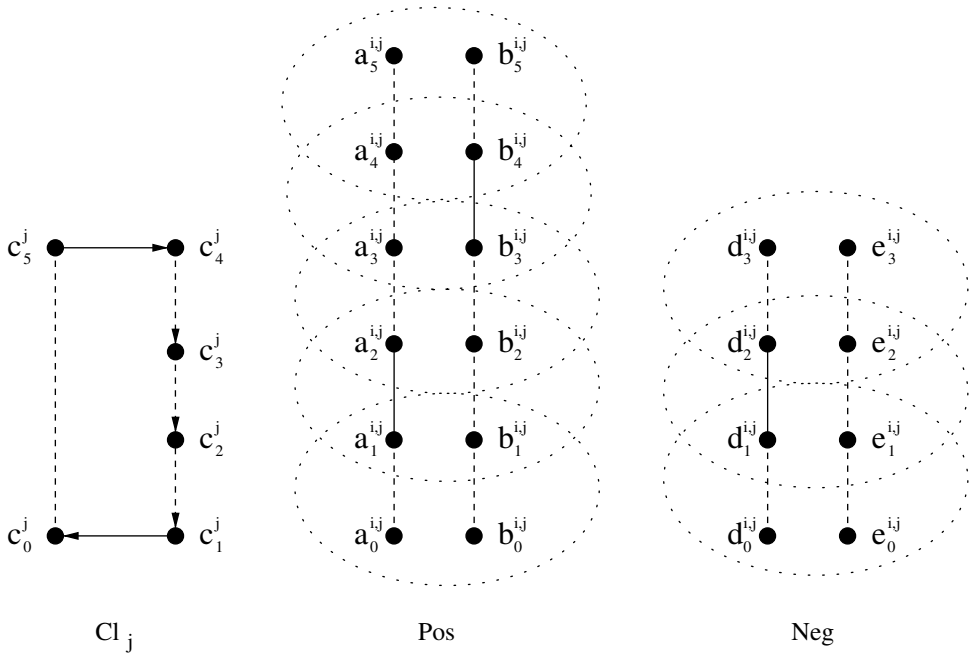


Fig. 3. The building blocks of the reduction. Clause base graph (left), positive variable connector (middle), and negative variable connector (right). In each case, the circled vertex sets represent XOR relations. Edge types (directed, undirected, solid, dashed) are as shown in the graphs.

identify $a_0^{i,j}$ with x_0^i and $b_0^{i,j}$ with x_1^i.
identify $a_5^{i,j}$ with c_k^i and $b_5^{i,j}$ with c_{k+1}^i.

5. if $\neg Y_i$ is the kth literal in clause C_j then do:
create a copy of negative variable connector with superscripts i, j.
identify $d_0^{i,j}$ with x_0^i and $e_0^{i,j}$ with x_1^i.
identify $d_3^{i,j}$ with c_k^i and $e_3^{i,j}$ with c_{k+1}^i.

This concludes the reduction which is clearly polynomial. For convenience, we also call an undirected dashed edge a *positive edge*, and a directed or undirected (solid) edge a *negative edge*.

($\Rightarrow$)

Suppose that 3-SAT(Φ) = TRUE. There exists a satisfying truth assignment τ: $(Y_i) \rightarrow \{T, F\}$ for Φ. For each variable graph $\{Var_i\}_{1 \leq i \leq t}$ complete the edge according to the assignment, in the following way: For every $1 \leq i \leq t$: (x_0^i, x_1^i) is determined to be a positive edge if $\tau(Y_i) = T$, or a negative edge, otherwise. Now, resolve the XOR relations in all the variable connectors. In each of the clause base graphs Cl_j, at least one of the three edges (c_1^j, c_2^j), (c_2^j, c_3^j), and (c_3^j, c_4^j), is a positive edge. It follows that in each clause base graph there is more than one parenthood connectivity component. Each such component has only solid edges

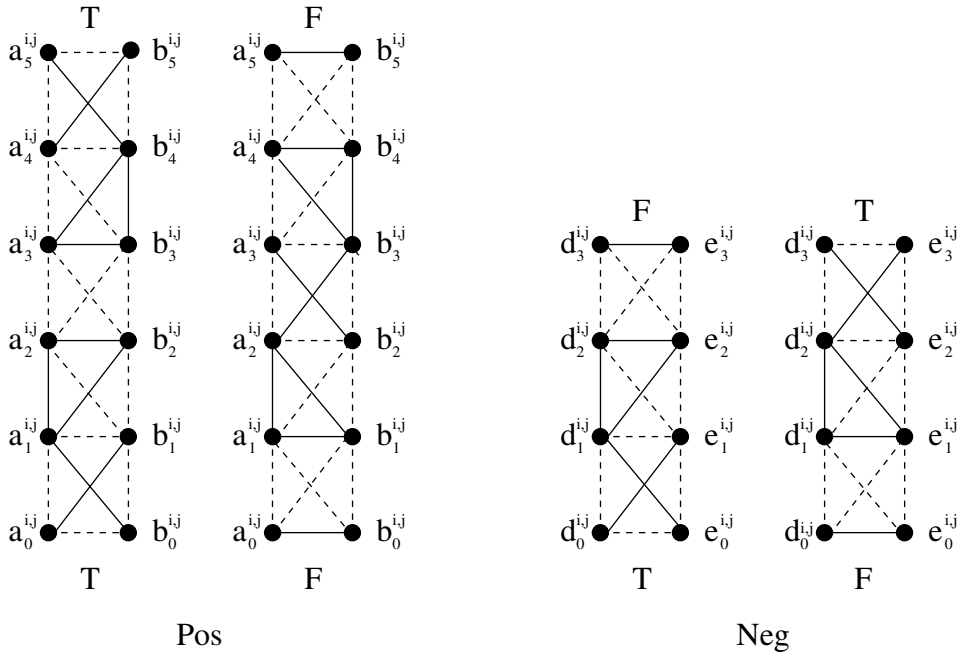


Fig. 4. Completion of variable positive and negative connectors. Note that in *Pos* (left) the completion propagates the type of the edge from the bottom to the top. In *Neg* (right) the types at the top and the bottom are reversed.

between members, and there is a directed edge between vertices c_a^j and c_b^j, only if $a = b + 1$. It follows that a directed tree can be built in each of the components of a clause base graph, under the constrains of G_c.

In addition, any of the two possible completions of each of the variable connectors, for any assignment, provides parenthood connectivity components in the variable connectors as follows: each component is a connected component in the subgraph of the solid edges only. These components can be directed in a transitive fashion (see Fig. 4). Thus, in each variable connector, one can form a directed sub-tree in each component, according to G_c constrains. Note that subgraphs of two different variable connectors Con_1 and Con_2 may be in the same parenthood connectivity component. This may happen only when two variable connectors are connected to the same clause base graph, to edges (c_1^j, c_2^j) and (c_3^j, c_4^j) respectively, and when (c_2^j, c_3^j) is a directed solid edge and (c_1^j, c_2^j) and (c_3^j, c_4^j) are undirected dashed edges. In this case, there is only one directed edge that connects Con_1 and Con_2, so trees T_1 and T_2 can be built on Con_1 and Con_2 separately and directed using the common edge, and then T_1 and T_2 can be united to a spanning directed tree on $Con_1 \cup Con_2$.

It follows that the graph can be divided into h parenthood connectivity components $\{R_i\}_{1 \le i \le h}$, where a directed spanning tree T_i can be built in each of this

components, under the constrains of G_c. Since each of the trees is in a different parenthood connectivity component, $\bigcup_{i=1}^{h} T_i$ is a directed forest spanning on G_c vertices. The constrained mixed completion graph can now be accomplished simply by completing the rest of the missing edges in each parenthood connectivity component according to its spanning tree, and between the components, by undirected dashed edges. It follows that CMGC $(G_c) =$ TRUE.

($\Leftarrow$)

Suppose that 3-SAT(Φ) = FALSE. Then for each truth assignment to the variables at least one of the clauses has *all* literals assigned to be FALSE. This implies that in any completion of G_c, there will be always one clause base graph Cl_j, such that all the three edges: (c_1^j, c_2^j), (c_2^j, c_3^j) and (c_3^j, c_4^j), are negative, i.e., solid directed edges. Thus c_5^j must be an ancestor of c_0^j in the forest. But this contradicts the undirected dashed edge between c_0^j and c_5^j, so a spanning forest which satisfied G_c constraints does not exist. Thus, CMGC $(G_c) =$ FALSE. □

4. An Algorithmic Solution for IPPH

In spite of the negative results of Sec. 3, we provide an efficient algorithmic approach to IPPH. We propose a probabilistic model for data generation and argue that the model holds for biological data. Under this model, we provide an algorithm that takes an expected polynomial time for both the rooted and the unrooted versions of IPPH. A similar probabilistic approach leading to comparable results was developed simultaneously and independently by Halperin and Karp.[12]

Pe'er *et al.*[16] suggested an algorithm that requires $\tilde{O}(mn)$ time for solving the rooted version of perfect phylogeny with missing data on an $n \times m$ haplotype matrix. Let the input incomplete haplotype matrix be $\tilde{M}$, with $\tilde{M}[i,j] \in \{0,1,?\}$, and let the root be r. We denote by $\mathsf{IDP}(\tilde{M}, r)$ the completed matrix obtained by performing this algorithm on $\tilde{M}$. We also use $\mathsf{IDP}(\tilde{M})$ to denote $\mathsf{IDP}(\tilde{M}, r_0)$. We use $h(\cdot,\cdot)$ to denote the Hamming distance between two binary vectors. We use $\sigma_0(j)$ and $\sigma_1(j)$ for the numbers of 0s and 1s in the jth column of M, respectively.

Suppose the root r_0 is known. Given an incomplete matrix M, we build a constrained mixed graph, as described in Sec. 2. We then perform primary label completion. According to Lemma 2, if M can be completed to M^* so that there exists an expansion M' of M^* that admits a perfect phylogeny, then there exists some secondary label completion of U_{G_c}, that can form the basis of completion of M^*. Thus, the computational challenge is to find such a secondary label completion. Suppose we were able to guess the correct secondary label completion. In that case, let $\tilde{M}$ be the resulting *incomplete* haplotype matrix, generated by performing label resolution accordingly. A completion of $\tilde{M}$ can be done in polynomial time by computing $\mathsf{IDP}(\tilde{M})$. Hence, the bottleneck step is finding a secondary label completion.

Due to the hardness result in Sec. 3, a polynomial time algorithm for finding the correct secondary label completion does not exist, unless P = NP. However,

by making several assumptions on the properties of the genotype data, this can be performed by a polynomial expected time algorithm. We now describe these assumptions, and for each one, we provide its biological motivation:

(1) Each entry value in the original genotype matrix is replaced by "?" with probability $\tilde{p}$, independently of the other values. This assumption makes sense as missing data entries are caused by technical problems in the biological experiment, that tend to generate independent "misses" ("?"s).

The same value $\tilde{p}$ may be used for all entries. One may claim that occasionally different SNPs may have different probability for a missing entry, due to distinct difficulties in sequencing different regions in the human genome. In that case, we denote by $\tilde{p_i}$ the probability for a missing entry in the ith SNP and set $\tilde{p}$ to be: $\tilde{p} \equiv \max_i\{\tilde{p_i}\}$.

(2) Each haplotype h_i, which is a node in a perfect phylogeny tree, is chosen to be in a genotype with probability of α_i, independently. This assumption is also made in the Hardy–Weinberg equilibrium model.[13] Moreover, we assume that these probabilities do not depend on n or m.

(3) The number of columns m grows much more slowly than the number of rows n. Specifically, we use $m = o(n^{.5})$. This assumption applies in all biological scenarios: In future experiments, the number of genotypes is expected to be even larger than today, while m is not expected to grow substantially, since m is the size of a "block", i.e., a region in the chromosome where the number of recombination events in the sampled population is small. A constant bound on m is thus plausible, but for our analysis, a much weaker assumption than that is required.

Our algorithm was designed to solve IPPH under the assumptions above. Informally, algorithm Prob-IPPH(M) ignores the missing data entries in order to decide the relation between each two columns in the matrix. As we shall prove, if it is impossible to conclude the relation deterministically from the matrix, with high probability, a correct relation is obtained just by guessing.

Prob-IPPH(M):

1. Let $G_c(M) = (V, E, X)$ be the constrained mixed graph of M.
2. Let $\overline{r}$ be a vector such that $\overline{r}_j = 0$ if $\sigma_0(j) > \sigma_1(j)$ and 1 otherwise.
3. Perform primary label completion of G_c(M).
4. For $j = 0 \rightarrow m$
   For each possible root $r \in \{0,1\}^m$, such that $h(r, \overline{r}) = j$ do
   Relabel the matrix entries according to r, so that $r_0 = (0, 0, \ldots, 0)$ is the new root.
   For $i = 0 \rightarrow \binom{m}{2}$
   For each possible secondary label completion of U_{G_c},
   such that $|\{(u_a, u_b) : (u_a, u_b) \in U_{G_c} \ \wedge \ L((u_a, u_b)) = 0\}| = i$ do
   Perform label resolution of M, thereby obtaining $\widetilde{M}$.
   If IDP($\widetilde{M}$) is compatible then output IDP($\widetilde{M}$) and halt.
5. Output: "no solution".

Fig. 5. An algorithm for IPPH.

Theorem 3. *Under the assumptions of the model, algorithm* Prob-IPPH(M) *solves IPPH correctly within expected time of* $\tilde{O}(m^2 n)$.

Proof. Correctness: Algorithm Prob-IPPH(M) enumerates all possible roots and all possible relations between every pair of columns (parenthood or brotherhood). Thus, correctness trivially follows.
Complexity: Steps 1–3 can all be performed in $O(m^2 n)$ time. The time bottleneck is step 4. The algorithm can stop for any $0 \leq i \leq \binom{m}{2}$, $0 \leq j \leq m$. We denote by S_0, an upper bound on the running time of the algorithm, when it stops for $i = 0, j = 0$, and by E_{S_0} the event that the algorithm stops for $i = 0, j = 0$. Similarly, we denote by $\bar{S}_0$, an upper bound to the running time of the algorithm, when it does not stop for $i = 0$, $j = 0$ and by $E_{\bar{S}_0}$ the event that the algorithm does not stop for $i = 0, j = 0$. Trivial upper bounds for S_0 and $\bar{S}_0$ are:

$$\begin{aligned} S_0 &= \tilde{O}(m^2 n), \\ \bar{S}_0 &= \tilde{O}(m^2 n 2^{m^2}). \end{aligned} \tag{1}$$

Let $F_{i,j}$ be the set of rows a such that $M[a,i] = M[a,j] = 1$, or $M[a,i] = 1$ and $M[a,j] = 2$, or $M[a,i] = 2$ and $M[a,j] = 1$. Clearly, if $F_{i,j} \neq \emptyset$ then the columns i, j are in parenthood relation.

Definition 6. Informative and Enigmatic Pairs of Columns
A pair of columns i, j in an incomplete genotype matrix is called an *informative pair* if there is at least one row a, such that $a \in F_{i,j}$ in the original *complete* genotype matrix, i.e., the two corresponding vertices of the columns in the perfect phylogeny forest are in parenthood relation. The row a is called an *informative row w.r.t. columns* i, j.

A pair of columns i, j in an incomplete genotype matrix is called an *enigmatic pair* if the relation between i, j cannot be concluded directly from these columns, and there exists at least one row a, such that $M[a,i] =?$ or $M[a,j] =?$. Such a row a is called an *enigmatic row w.r.t. columns* i, j.

Let $I_{i,j}$ be the event that the pair of columns i, j is an informative pair, and let $E_{i,j}$ be the event that the pair of columns i, j is an enigmatic pair. Let $I_{i,j}^{(a)}$, denote the event that row $a \in F_{i,j}$. We denote the set of all *pairs* of haplotypes, which create a genotype which belongs to $F_{i,j}$ by $\mathcal{H}_{F_{i,j}}$. Now, according to assumption (2), the probability that row a belongs to $F_{i,j}$ is $\Pr[I_{i,j}^{(a)}] = \sum_{h_a, h_b \in \mathcal{H}_{F_{i,j}}} \alpha_a \alpha_b$. We denote $\Pr\left[I_{i,j}^{(a)}\right]$ by $q_{i,j}$.

Let $E_{i,j}^{(a)}$ denote the event that the row a is enigmatic w.r.t. the columns i, j. The probability of $E_{i,j}^{(a)}$, for all a, i, j is $p = 2\tilde{p}(1 - \tilde{p}) + \tilde{p}^2$.

We now investigate the event that both $I_{i,j}$ and $E_{i,j}$ occur. In other words, the columns i, j are both informative and enigmatic. In order for the columns i, j to be informative, for at least one row a the event $I_{i,j}^{(a)}$ must occur. In order for the columns i, j to be enigmatic, we require that for every row a, if the relation

(parenthood or brotherhood) between i, j can be concluded directly using this row, then $E_{i,j}^{(a)}$ occurs. Observe that since the event $I_{i,j}$ occurs, the relation of i, j must be parenthood. To conclude, in order that both events $I_{i,j}$ and $E_{i,j}$ take place, two conditions must hold:

(1) At least for one row a the event $I_{i,j}^{(a)}$ occurs.
(2) For every row a, if the event $I_{i,j}^{(a)}$ occurs, then the event $E_{i,j}^{(a)}$ also occurs.

Let $\mathcal{A}$ be the event $\left[\forall a\text{: } \left\{I_{i,j}^{(a)} \rightarrow E_{i,j}^{(a)}\right\}\right]$, and let $\mathcal{B}$ be the event $\left[\forall a : \left\{\neg I_{i,j}^{(a)}\right\}\right]$. First, observe that if event $\mathcal{B}$ occurs, i.e., the event $I_{i,j}^{(a)}$ does not occur for all rows a, then for each row a, the event $\left[I_{i,j}^{(a)} \rightarrow E_{i,j}^{(a)}\right]$ occurs. In other words, if event $\mathcal{B}$ occurs then event $\mathcal{A}$ also occurs. Thus: $\mathcal{B} \subseteq \mathcal{A}$. Second, note that event $\mathcal{A}$ is actually condition (2), and the complement of $\mathcal{B}$, denoted by $\bar{\mathcal{B}}$, is condition (1).

Hence, we can write:

$$[I_{i,j} \text{ and } E_{i,j}] \equiv \mathcal{A} \cap \bar{\mathcal{B}} \equiv \mathcal{A} \setminus \mathcal{B}.$$

Since $\mathcal{B} \subseteq \mathcal{A}$ then $\Pr[\mathcal{A} \setminus \mathcal{B}] = \Pr[\mathcal{A}] - \Pr[\mathcal{B}]$, and we can now calculate the joint probability $\Pr[I_{i,j}, E_{i,j}]$:

$$\begin{aligned}
\Pr[I_{i,j}, E_{i,j}] &= \Pr[\forall a : \{I_{i,j}^{(a)} \rightarrow E_{i,j}^{(a)}\}] - \Pr[\forall a : \{\neg I_{i,j}^{(a)}\}] \\
&= \Pr[\forall a : \{\neg I_{i,j}^{(a)} \vee E_{i,j}^{(a)}\}] - \Pr[\forall a : \{\neg I_{i,j}^{(a)}\}] \\
&= \Pr[\forall a : \{\neg (I_{i,j}^{(a)} \wedge \neg E_{i,j}^{(a)})\}] - \Pr[\forall a : \{\neg I_{i,j}^{(a)}\}] \\
&= [1 - q_{i,j}(1-p)]^n - (1 - q_{i,j})^n.
\end{aligned}$$

Next, we calculate the conditional probability of a pair of columns to be an informative pair, when the pair is known to be enigmatic:

$$\begin{aligned}
\Pr[I_{i,j} | E_{i,j}] &= \frac{\Pr[I_{i,j}, E_{i,j}]}{\Pr[E_{i,j}]} \\
&= \frac{[1 - q_{i,j}(1-p)]^n - (1 - q_{i,j})^n}{1 - (1-p)^n} \\
&\leq \frac{[1 - q_{i,j}(1-p)]^n}{1 - (1-p)^n}.
\end{aligned}$$

The probability for a pair to be not informative, when it is known to be enigmatic, is:

$$\begin{aligned}
\Pr[\neg I_{i,j} | E_{i,j}] &\geq 1 - \frac{[1 - q_{i,j}(1-p)]^n}{1 - (1-p)^n} \\
&= \frac{1 - (1-p)^n - [1 - q_{i,j}(1-p)]^n}{1 - (1-p)^n} \\
&\geq 1 - (1-p)^n - [1 - q_{i,j}(1-p)]^n. \qquad (2)
\end{aligned}$$

Note that $\Pr[\neg I_{i,j} | E_{i,j}]$ is the probability for a "success" with respect to columns i, j: Given that the pair i, j is enigmatic (i.e., we can not conclude its relation), the pair is not informative, which means that the relation must be brotherhood.

We use the following definitions:

$$u = \max_{i,j}\{1-p, 1-q_{i,j}(1-p)\}.$$

Due to assumption (2), $0 < u < 1$, and does not depend on n or m. When substituting (3) into inequality (2), we get:

$$\Pr[I_{i,j}|E_{i,j}] \leq 2u^n.$$

Since there are $\binom{m}{2}$ pairs of columns, the probability that at least one of the enigmatic pairs is an informative pair, can be bounded using a union bound:

$$\begin{aligned}\Pr[\exists i,j:\ I_{i,j}|E_{i,j}] &= \Pr\left[\bigcup_{i<j}(I_{i,j}|E_{i,j})\right] \\ &\leq \sum_{i<j}\Pr[I_{i,j}|E_{i,j}] \\ &\leq \binom{m}{2}2u^n.\end{aligned}$$

Thus, the complementary event, which represents "success", can be bounded by:

$$\Pr[\forall i,j:\ \neg I_{i,j}|E_{i,j}] \geq 1 - \binom{m}{2}2u^n.$$

If the relation between two columns cannot be concluded, then the algorithm starts with a guess of a brotherhood relation. Thus, an error might occur when $i = 0$, only if a pair is an informative pair. Since $m = \mathrm{o}(n^{.5})$, we replace n with c_1m^2, where c_1 is a constant. There exists m_0, such that $\forall m \geq m_0$ the probability that the algorithm finds the correct solution when $i = 0$, and when the root is known to be $\tilde{r}$, is:

$$\Pr[E_{S_0} \mid \text{root is } \tilde{r}] \geq 1 - m^2u^{c_1m^2}. \tag{3}$$

We now calculate the probability for an error in deciding the root, when $i = 0$. Denote by r the root calculated by the algorithm when $i = 0$. Let P_i^0, P_i^1 be the probabilities for 0 and 1 in the ith row, respectively. Hence $P_i^0 + P_i^1 = 1 - \bar{p}$, where $\bar{p}$ is the probability for "?" in a haplotype. A specific site in the genotype is missing if at least one out of the two corresponding sites in the haplotypes is missing. Thus, $\tilde{p} = 1 - (1-\bar{p})^2$ or, equivalently, $\bar{p} = 1 - \sqrt{1-\tilde{p}}$. Let n_i^0, n_i^1 be the number of 0 and 1 in the ith row, respectively. Without loss of generality, suppose that $P_i^0 < P_i^1$, then the root can be determined to be "1" in the ith component, according to the majority rule in determination of the root in perfect phylogeny (see Gusfield[9]). The probability for an error in the ith component can be bounded using Chernoff bound:[1]

$$\begin{aligned}\Pr[r_i \neq \tilde{r}_i] &= \Pr[n_i^0 > n_i^1 \mid P_i^0 < P_i^1] \\ &= \Pr[n_i^0 > \frac{1-\bar{p}}{2} \mid P_i^0 < P_i^1]\end{aligned}$$

$$
\begin{aligned}
&= \Pr[n_i^0 > nP_i^0 + nP_i^0\left(\frac{1-\bar{p}}{2P_i^0} - 1\right) \mid P_i^0 < P_i^1] \\
&\leq e^{-\frac{nP_i^0\left(\frac{1-\bar{p}}{2P_i^0}-1\right)^2}{4}} \\
&= e^{-bn},
\end{aligned}
$$

where $b = \frac{P_i^0}{4}(\frac{1-\bar{p}}{2P_i^0} - 1)^2$ is a constant. Using the union bound again, there exists m_0, such that $\forall m \geq m_0$ the probability for the root to be correct, when $i = 0$

$$\Pr[r = \tilde{r}] \geq 1 - me^{-bc_2m^2},$$

where c_2 is a constant. Now, we can bound the probability that the algorithm stops when $i = 0$:

$$
\begin{aligned}
\Pr[E_{S_0}] &\geq \Pr[E_{S_0}, r = \tilde{r}] \\
&= \Pr[E_{S_0} \mid r = \tilde{r}] \Pr[r = \tilde{r}] \\
&\geq (1 - m^2u^{c_1m^2})(1 - me^{-bc_2m^2}) \\
&\geq 1 - e^{-c_3m^2}, \qquad (4)
\end{aligned}
$$

where c_3 is a constant. Using inequality (1), we are now able to bound the expected running time of the algorithm:

$$
\begin{aligned}
\mathbb{E}[\text{running time}] &\leq \Pr[E_{S_0}]S_0 + (1 - \Pr[E_{S_0}])\bar{S}_0 \\
&\leq \tilde{O}(m^2n) + e^{-c_3m^2}\tilde{O}(m^2n2^{m^2}), \qquad (5)
\end{aligned}
$$

Since $m = \mathrm{o}(n^{.5})$ we can choose c_1 large enough (c_3 is larger when c_1 increases), such that the second summand vanishes for $n \to \infty$, and thus:

$$\mathbb{E}[\text{running time}] = \tilde{O}(m^2n).$$

Observe that in addition to proving that the expected running time is polynomial, we also showed that the running time is polynomial with high probability.

□

Note that the above analysis applies also when the root is known. In that case, obviously, we need not enumerate all possible roots, so the worst case running time can only improve. Asymptotically, the expected running time is the same.

5. Experimental Results

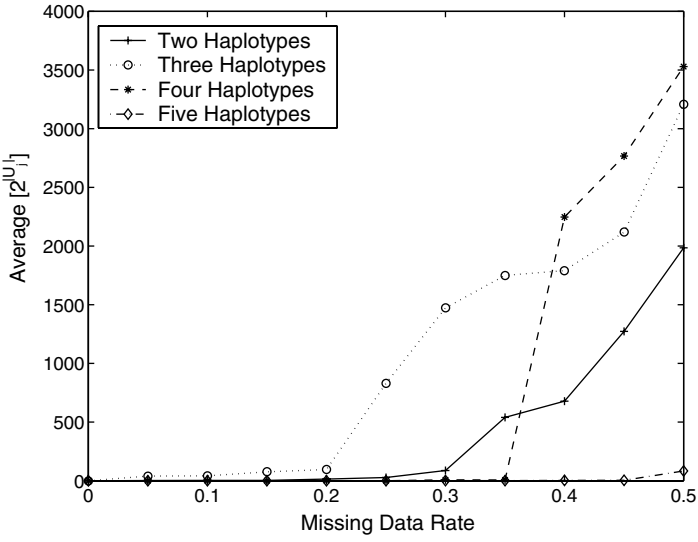
In order to assess our algorithm, we applied it on simulated data. The simulations used parameters which were adopted from several large scale biological studies.[4,7,12] By Theorem 3, the algorithm always outputs a correct solution. Although we proved that under our model assumptions the expected running time is $\tilde{O}(m^2n)$, we wanted to estimate the actual running time, under realistic biological parameters and beyond the range of the model assumptions. Specifically, we wanted to

calculate the expected number of different phylogenetic tree solutions for a given data set. The proof of Theorem 3 implies that $\Gamma = 2^{|U_{G_c}|}$ is an upper bound on the number of different phylogeny solutions, and the dominant factor in the complexity of the algorithm, in the rooted version of the problem.

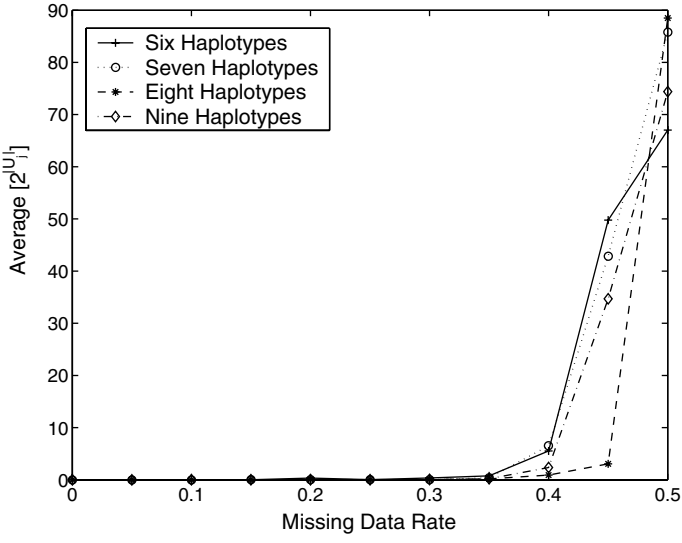
In each different experiment, we randomly generated $N = 10^5$ perfect phylogeny trees. We used the following procedure to generate a perfect phylogeny tree of haplotypes: We start with a binary root vector with $m = 30$ sites. Initially, no site is marked. In each step, we randomly pick a node from the current tree and an unmarked site, add a new child haplotype to that node in which only the state of that site is changed, and mark the site. For each tree, we randomly chose k haplotypes for reconstructing the genotypes, where $k = 2, 3, \ldots, 9$. We assigned frequencies, denoted by $\alpha_1, \alpha_2, \ldots, \alpha_k$, to the k chosen haplotypes, such that $\sum_{i=1}^{k} \alpha_i = 1$ and $\forall i : \alpha_i \geq 0.05$. For each tree, different frequencies were assigned. Next, we generated 200 genotypes according to the chosen haplotypes and their assigned frequencies. Introducing missing data entries to the genotypes was performed as follows: Each site in the genotypes data was flipped into a missing entry independently with probability p. Since we observed in real data $p \approx 0.1$,[4,7] we checked a wider range: $p = 0, 0.05, \ldots, 0.5$. Thus, for each sampled tree $T_j : j = 1, 2, \ldots, N$, we sampled one incomplete genotype matrix M_j of size 200×30. We applied our algorithm on each M_j. We denote $U_{G_c(M_j)}$ by U_j. After performing steps 1–3 of the algorithm, we stopped at $i = 0$ and calculated $2^{|U_j|}$. As was shown in Sec. 4, if the secondary label completion is known, it is possible in $\tilde{O}(m^2 n)$ time to output the solution to IPPH. Hence, completion of the algorithm, for each M_j, should take less than $2^{|U_j|}\tilde{O}(m^2 n)$ time. The dominating factor in the running time is the random variable $2^{|U_j|}$, whose expectation is approximated by: $\mathbb{E}[\Gamma] = \mathbb{E}[2^{|U_j|}] \approx \frac{1}{N}\sum_{j=1}^{N} 2^{|U_j|}$.

The results are presented in Fig. 6. In all experiments, $\mathbb{E}[\Gamma]$ was below 3500 (compared to a theoretical upper bound of $2^{\binom{m}{2}} = 2^{435}$). When the missing data rate is below 20%, $\mathbb{E}[\Gamma]$ was smaller than 100. Another observation, is that the larger the number of chosen haplotypes, the smaller the value of $\mathbb{E}[\Gamma]$. Notably, in all cases we found a correct root: either by finding at least one haplotype, which is homozygous with no missing entries in all sites, or by using the majority rule described in the algorithm.

To demonstrate that in real biological data, a root is readily available, we chose the genotype data of Daly *et al.*[4] This data set consists of 103 SNPs and 129 genotypes. We checked all possible $\binom{103}{2} = 5253$ blocks. In *all* the blocks of size 65 or smaller, there was always at least one genotype that was homozygous in all alleles and without any missing entries. This genotype is actually a haplotype, since it can be resolved in only one possible way, and hence, it can be used as a root. Since the size of a block is almost always smaller than 30, this naive simple method can be used for finding a root in biological data.



A



B

Fig. 6. Simulation results: both figures show the average of $2^{|U_j|}$ (y-axis), which represents the dominating factor in the running time of the algorithm for different missing data rates (x-axis). Each different line in the figures corresponds to a different number of haplotypes chosen from the tree (see legend).

6. Concluding Remarks

We investigated the incomplete perfect phylogeny haplotype problem. The goal is phasing of genotypes into haplotypes, under the perfect phylogeny model, where

some of the data are missing. We proved that the problem in its rooted version is NP-complete. We also provided a practical expected polynomial-time algorithm, under a biologically motivated probabilistic model of the problem. We applied our algorithm on simulated data, and concluded that the running time and the number of distinct candidate phylogeny solutions are relatively small, under a broad range of biological conditions and parameters, even when the missing data rate is 50%. An accurate treatment for phasing of genotypes with missing entries can therefore be obtained in practice. In addition, due to the small number of phylogenetic solutions observed in simulations, incorporation of additional statistical and combinatorial criteria with our algorithm is feasible.

After the completion of this study, Gramm *et al.*[8] reported on another investigation of the ROOTED-IPPH problem. They proved that this problem is NP-complete even when the phylogeny is a path and only one allele of every polymorphic site is present in the population in its homozygous state. This provides an alternative proof that the ROOTED-IPPH problem is NP-complete. They also give a linear-time algorithm for the problem for the special case, in which the phylogeny is a path.

Acknowledgments

This research was supported by the Israel Science Foundation (grant 309/02). We thank Roded Sharan for fruitful discussions.

References

1. Alon N, Spencer JH, *The Probabilistic Method*, John Wiley and Sons, Inc., 2000.
2. Bafna V, Gusfield D, Lancia G, Yooseph S, Haplotyping as perfect phylogeny: a direct approach, *J Comput Biol* **10**(3–4):323–340, 2003.
3. Clark A, Inference of haplotypes from PCR-amplified samples of diploid populations, *Mol Biol Evol* **7**(2):111–122, 1990.
4. Daly MJ, Rioux JD, Schaffner SF, Hudson TJ, Lander ES, High-resolution haplotype structure in the human genome, *Nature Genet* **29**(2):229–232, 2001.
5. Eskin E, Halperin E, Karp RM, Large scale reconstruction of haplotypes from genotype data, in *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)*, pp. 104–113, The Association for Computing Machinery, 2003.
6. Excoffier L, Slatkin M, Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population, *Mol Biol Evol* **12**(5):912–917, 1995.
7. Gabriel SB, Schaffner SF, Nguyen H, Moore JM, Roy J, Blumenstiel B, Higgins J, DeFelice M, Lochner A, Faggart M, Liu–Cordero SN, Rotimi C, Adeyemo A, Cooper R, Ward R, Lander ES, Daly MJ, Altshuler D, The structure of haplotype blocks in the human genome, *Science* **296**:2225–2229, 2002.
8. Gramm J, Nierhoff T, Sharan R, Tantau T, On the complexity of haplotyping via perfect phylogeny, in *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes*, pp. 35–46, 2004.
9. Gusfield D, Efficient algorithms for inferring evolutionary trees, *Networks* **21**: 19–28, 1991.

10. Gusfield D, Haplotyping as perfect phylogeny: conceptual framework and efficient solutions, in *Proceedings of the Sixth Annual International Conference on Research in Computational Molecular Biology (RECOMB 02)*, pp. 166–175, The Association for Computing Machinery, 2002.
11. Halldorsson BV, Bafna V, Edwards N, Lippert R, Yooseph S, Istrail S, Combinatorial problems arising in SNP, in *Proceedings of the Fourth International Conference on Discrete Mathematics and Theoretical Computer Science (DMTCS)*, volume 2731 of *Lecture Notes in Computer Science*, pp. 26–47. Springer, 2003.
12. Halperin E, Karp RM, Perfect phylogeny and haplotype assignment, in *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 04)*, pp. 10–19, The Association for Computing Machinery, 2004.
13. Hardy GH, Mendelian proportions in a mixed population, *Science* **18**:49–50, 1908.
14. Hoehe MR, Haplotypes and the systematic analysis of genetic variation in genes and genomes, *Pharmacogenomics* **4**(5):547–570, 2003.
15. Hoehe MR, Kopke K, Wendel B, Rohde K, Flachmeier C, Kidd KK, Berrettini WH, Church GM, Sequence variability and candidate gene analysis in complex disease: association of μ opioid receptor gene variation with substance dependence, *Hum Mol Genet* **9**:2895–2908, 2000.
16. Shamir R, Sharan R, Pe'er I, Pupko T, Incomplete directed perfect phylogeny, *SIAM J Comp*, **33**(3):590–607, 2004.
17. Kimmel G, Shamir R, The incomplete perfect phylogeny haplotype problem, in *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes*, pp. 59–70, 2004.
18. Kruglyak L, Nickerson DA, Variation is the spice of life, *Nature Genet* **27**:234–236, 2001.
19. Long J, Williams RC, Urbanek M, An EM algorithm and testing strategy for multiple-locus haplotypes, *Amer J Hum Genet* **56**(3):799–810, 1995.
20. Niu T, Qin ZS, Xu X, Liu JS, Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms, *Amer J Hum Genet* **70**(1):157–169, 2002.
21. Patil N, Berno AJ, Hinds DA, Barrett WA, Doshi JM, Hacker CR, Kautzer CR, Lee DH, Marjoribanks C, McDonough DP, Nguyen BT, Norris MC, Sheehan JB, Shen N, Stern D, Stokowski RP, Thomas DJ, Trulson MO, Vyas KR, Frazer KA, Fodor SP, Cox DR, Blocks of limited haplotype diversity revealed by high-resolution scanning of human chromosome 21, *Science* **294**:1719–1723, 2001.
22. Sachidanandam R, Weissman D, Schmidt SC, Kakol JM, Stein LD, Marth G, Sherry S, Mullikin JC, Mortimore BJ, Willey DL, Hunt SE, Cole CG, Coggill PC, Rice CM, Ning Z, Rogers J, Bentley DR, Kwok PY, Mardis ER, Yeh RT, Schultz B, Cook L, Davenport R, Dante M, Fulton L, Hillier L, Waterston RH, McPherson JD, Gilman B, Schaffner S, Van Etten WJ, Reich D, Higgins J, Daly MJ, Blumenstiel B, Baldwin J, Stange–Thomann N, Zody MC, Linton L, Lander ES, Altshuler D, A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms, *Nature* **291**:1298–2302, 2001.
23. Steel MA, The complexity of reconstructing trees from qualitative characters and subtrees, *J Classif* **9**:91–116, 1992.
24. Stephens M, Smith NJ, Donnelly P, A new statistical method for haplotype reconstruction from population data, *Amer J Hum Genet* **68**(4):978–989, 2001.

Dr. Gad Kimmel is a Ph.D. candidate at the Computer Science Tel Aviv University. He holds an MD degree from the Technion Institute of Haifa. His current interests are combining algorithmic and statistical theory in order to model mutations and recombination processes, and thereby develop improved methods for genotype analysis (e.g., haplotype inference, association studies, and tag SNPs selection).

Dr. Ron Shamir is a professor and head of the School of Computer Science, and the incumbent of the Raymond and Beverly Sackler Chair in Bioinformatics at Tel Aviv University. He holds a BSc in mathematics and physics from the Hebrew University and a PhD in operations research from UC Berkeley. His expertise covers graph theoretic methods for design and analysis of algorithms, specializing in problems in molecular biology. His current research focuses on developing computational tools for analysis of high-throughput heterogeneous genomic data, comparative genomics, genetic and regulatory networks and haplotyping. He is a member of the editorial board of several leading journals and series in computational biology, and on the steering committee of RECOMB. Dr. Shamir has published over 130 scientific publications.

Maximum Likelihood Resolution of Multi-block Genotypes

Gad Kimmel
School of Computer Science
Tel Aviv University
Tel Aviv 69978, Israel
kgad@tau.ac.il

Ron Shamir
School of Computer Science
Tel Aviv University
Tel Aviv 69978, Israel
rshamir@tau.ac.il

ABSTRACT

We present a new algorithm for the problems of genotype phasing and block partitioning. Our algorithm is based on a new stochastic model, and on the novel concept of probabilistic common haplotypes. We formulate the goals of genotype resolving and block partitioning as a maximum likelihood problem, and solve it by an EM algorithm. When applied to real biological SNP data, our algorithm outperforms two state of the art phasing algorithms. Our algorithm is also considerably more sensitive and accurate than a previous method in predicting and identifying disease association.

Categories and Subject Descriptors

J.3 [**Computer Applications**]: Life and Medical Sciences—*Biology and genetics*; G.3 [**Probability and Statistics**]: Probabilistic algorithms

General Terms

algorithms, haplotyping

Keywords

haplotype, haplotype block, genotype, SNP, algorithm, maximum likelihood, genotype phasing, haplotype resolution, disease association

1. INTRODUCTION

A major challenge after the completion of the human genome project is to learn about DNA differences among individuals. This knowledge can lead to better understanding of human genetics, and to finding the genetic causes for complex and multi-factorial diseases. Most DNA differences among individuals are single base sites, in which more than one nucleic acid can be observed across the population. Such differences and their sites are called *single nucleotide polymorphisms* (SNPs) [19, 7]. Usually only two alternative bases occur at a SNP site. Millions of SNPs have already been detected [20, 22], out of an estimated total of 10 millions common SNPs [13].

When studying polymorphism in the population, one looks only at SNP sites and disregards the long stretches of bases between them that are the same in the population. The sequence variants at each site are called the *alleles* at that site. The sequence of alleles in contiguous SNP sites along a chromosomal region is called a *haplotype*. Recent evidence indicates that haplotypes tend to be preserved along relatively long genomic stretches, with recombination occurring primarily in narrow regions called *hot spots* [7, 19]. The regions between two neighboring hot spots are called *blocks*. The number of distinct haplotypes within each block that are observed in a population is very limited: typically, some 70-90% of the haplotypes within a block are identical (or almost identical) to very few (2-5) distinct *common haplotypes* [19]. This finding is very important for disease association studies, since once the blocks and the common haplotypes are identified, one can, in principle, obtain a much stronger association between a haplotype and a disease phenotype.

Several studies have concentrated on the problem of block identification in a given collection of haplotypes: Zhang et al. [27, 28] sought a block partitioning that minimizes the number of tag SNPs (roughly speaking, this is a set of sites with the property that the combination of alleles in it uniquely identifies the alleles at all other sites, or a prescribed fraction of the haplotypes in that block). Koivisto et al. [12] used a minimum description length (MDL) criterion for block definition. Kimmel et al. [11] minimized the total number of common haplotypes, while allowing errors and missing data. All these studies used the same basic dynamic programming approach of [27] to the problem, but differed in the optimization criterion used within the dynamic programming computation.

The block partitioning problem is intertwined with another problem in diploid organisms. Such organisms (including humans) have two near-identical copies of each chromosome. Most techniques for determining SNPs do not provide the haplotype information separately for each of the two copies. Instead, they generate for each site *genotype* information, i.e., an unordered pair of allele readings, one from each copy [20].

Hence, given the genotype data {A,A} {A,C} {C,G} for three SNP sites in a certain individual, there are two possible haplotype pair solutions: (ACC and AAG), or (ACG and AAC). A genotype with two identical bases in a site is

RECOMB'04, March 27–31, 2004, San Diego, California, USA.

called *homozygote*, while a genotype with two different bases is called *heterozygote* in that site. The genotype in the example above is homozygote for the allele A in the first site, and heterozygote in the second and third sites. The process of inferring the haplotypes from the genotypes is called *phasing* or *resolving*.

In the absence of additional information, each genotype with h heterozygote sites can be resolved in 2^{h-1} different ways. Resolving is done simultaneously in all the available genotypes and is based on some assumptions on how the haplotypes were generated. The first approach to haplotype resolution was Clark's parsimony-based algorithm [3]. Likelihood-based EM algorithms [6, 15] gave better results. Stephens et al. [21] and Niu et al. [18] proposed MCMC-based methods which gave promising results. All of those methods assumed that the genotype data correspond to a single block with no recombination events. Hence, for multi-block data the block structure must be determined separately.

A novel combinatorial model was suggested by Gusfield [9]. According to this model, the resolution must produce haplotypes that define a *perfect phylogeny tree*. Gusfield provided an efficient yet complex algorithm for the problem. Simpler, direct efficient algorithms under this model were recently developed [5, 1]. Eskin et al. [5] showed good performance with low error rates on real genotypes.

While elegant and powerful, the perfect phylogeny approach has certain limitations: first, it assumes that the input data admit a perfect phylogeny tree. This assumption is often violated in practice, due to data errors and rare haplotypes. In fact, Eskin at al. show that in the real data that they analyzed, a block does not necessarily admit a perfect phylogeny tree. Second, the model requires partition of data into blocks by other methods. Third, the solution to the problem may not be unique and there may be several (or many) indistinguishable solutions. (These limitations were addressed heuristically in [5]). Recently, Greenspan and Geiger [8] proposed a new method and algorithm, called *HaploBlock*, which performs resolution while taking into account the blocks structure. The method is based on a Bayesian network model. Very good results were reported.

In this study we provide a new algorithm for block partitioning and phasing. Our algorithm is based on a new model for genotype generation. Our model is based on a haplotype generation model, parts of which were suggested by Koivisto et al. [12]. In our model, common haplotypes are redefined in a probabilistic setting, and we seek a solution that has maximum likelihood, using an EM algorithm. The model allows errors and rare haplotypes, and the algorithm is particularly tailored to the practical situation in which the number of common haplotypes is very small. We applied our algorithm to two genotype data sets: on the data set of Daly et al. [4] our algorithm performed better than HaploBlock [8] and Eskin et al. [5]. On genotype and phenotype data for the μ opioid receptor gene, due to Hoehe et al. [10], our algorithm revealed strong association for disease, by using blocks partitioning and resolving, and improved sharply over the original analysis in [10].

Unlike most former probabilistic approaches [6, 15, 21, 18], our algorithm reconstructs the block partitioning and resolves the haplotypes simultaneously, and assigns a likelihood value to the complete solution. Consequently, it is considerably faster and more accurate. While our approach has some resemblance to HaploBlock, there are also significant differences. First, our approach is not based on a Bayesian network, but rather computes the maximum likelihood directly. Second, our algorithm actually computes the likelihood function of each block, and thus the real maximum likelihood partitioning is optimized, while HaploBlock uses an MDL criterion for block partitioning. Third, once the model parameters are found, we solve the phasing problem directly to optimality, such that the likelihood function is maximized. In constrast, HaploBlock applies a heuristic to find the block partitioning, even though this partitioning is part of the model parameters. Fourth, our stochastic model allows a continuous spectrum of probabilities for each component in each common haplotype, while the HaploBlock software allows only two common probability values for all mutations. HaploBlock has the theoretical advantage of allowing a larger number of common haplotypes, but this is apparently less relevant in practice [7, 4]. HaploBlock's model also incorporates inter-block transitions, while we handle them separately after the main optimization process.

This paper is organized as follows: In Section 2 we present our stochastic model, in Section 3 we show how block partitioning and resolution of haplotypes are performed under our model. Section 4 contains our results on the two real data sets.

2. THE STOCHASTIC MODEL

Consider first the problem of resolving a single block. The input to the problem is presented by a $n \times m$ *genotype matrix* M, in which the rows correspond to samples (individuals genotyped), and columns correspond to SNP sites. Hence, the i^{th} row $M[i,*]$ describes the i^{th} genotype (the vector of readings for all the SNP sites), which is also denoted by $\boldsymbol{g_i}$. We assume that all sites are bi-allelic, and that the two alleles were renamed arbitrarily to 0 and 1. The genotype readings are denoted by $M[i,j] \in \{0,1,2\}$. 0 and 1 stand for the two homozygote types {0,0} and {1,1}, respectively, and 2 stands for a heterozygote. A $2n \times m$ binary matrix M' is an *expansion* of the genotype matrix M if each row $M[i,*]$ expands to two rows denoted by $M'[i,*]$ and $M'[i',*]$, with $i' = n + i$, satisfying the following: for every i, if $M[j,i] \in \{0,1\}$, then $M[i,j] = M'[i,j] = M'[i',j]$; if $M[i,j] = 2$, then $M'[i,j] \neq M'[i',j]$. M' is also called a *haplotype matrix* corresponding to M. Given a genotype matrix, the *phasing problem* is to find its best expansion, i.e., the best n pairs of haplotype vectors that could have generated the genotype vectors. "Best" must be defined with respect to the data model used.

We now describe our stochastic model for how the haplotype matrix of a single block is generated. The model aims to reflect the fact that only few distinct common haplotypes are usually observed in each block [7, 4], and the variability between observed haplotypes originating from the same common haplotype. The model assumes a set of common haplotypes that occur in the population with certain probabilities. Each genotype is created by selecting independently two of the common haplotypes according to their probabilities and forming their confluence. The two (possibly identical) common haplotypes are called the *creators* of that genotype. The key property of our model is the probabilistic formulation of the common haplotypes: Formally, a *probabilistic common haplotype* is a vector, whose

components are the probabilities of having the allele '1' in each site of a haplotype created by it. Hence, a vector of only zeroes and ones corresponds to a standard (consensus) common haplotype, and a vector with fractional values allows for deviations from the consensus with certain (small) probabilities, independently for each site. In this way, a common haplotype may appear in different genotypes in distinct forms. A similar model was used in [12] in the context of block partitioning of haplotype (phased) data.

A precise definition of the stochastic model is as follows. Assume that the genotype matrix M contains only one block. Let k be the number of common haplotypes in that block. Let $\{\boldsymbol{\theta_i}\}_{1\le i\le k}$ be the probability vectors of the common haplotypes, where $\boldsymbol{\theta_i} = (\theta_{i,1}, \ldots, \theta_{i,m})$ and $\theta_{i,j}$ is the probability to observe '1' in the j^{th} site of the i^{th} common haplotype. (Consequently, $1-\theta_{i,j}$ is the probability to observe '0' in that site.) Let $\alpha_i > 0$ be the probability of the i^{th} common haplotype in the population, with $\sum_{i=1}^{k}\alpha_i = 1$. Each row in the matrix M is generated as follows:

- Choose a number i between 1 and k according to the probability distribution $\{\alpha_1, \ldots, \alpha_k\}$. i is the index of the first common haplotype.
- The haplotype $(x_1, \ldots, x_m)$ is generated by setting, for each site j independently, $x_j = 1$ with probability $\theta_{i,j}$.
- Repeat the steps above for the second haplotype and form their confluence. The result is the genotype in that row.

For generating a matrix with several blocks, the process is repeated for each block independently. Our main task will be to show how to infer the parameters and the haplotypes from genotype data of a single block. This inference also gives a likelihood for the block. Given a multi-block matrix, a dynamic programming algorithm is used to find the maximum likelihood block partitioning.

3. INFERRING THE MODEL PARAMETERS

For a single genotype $\boldsymbol{g_j}$, assuming its creators $\boldsymbol{\theta_a}$ and $\boldsymbol{\theta_b}$ are known, the probability of obtaining $\boldsymbol{g_j}$ is:

$$f(\boldsymbol{g_j};\boldsymbol{\theta_a},\boldsymbol{\theta_b}) = \prod_{i=1}^{m}\begin{cases}(1-\theta_{a,i})(1-\theta_{b,i}) & g_{j,i}=0\\ \theta_{a,i}\theta_{b,i} & g_{j,i}=1\\ \theta_{a,i}(1-\theta_{b,i})+\theta_{b,i}(1-\theta_{a,i}) & g_{j,i}=2\end{cases}.$$

We denote by I_i and J_i the index of the first and second creator of genotype g_i, respectively. The complete likelihood of all genotypes is:

$$L(M) = \prod_{i=1}^{n}\alpha_{I_i}\alpha_{J_i}f(\boldsymbol{g_i};\boldsymbol{\theta_{I_i}},\boldsymbol{\theta_{J_i}}).$$

Let the random variable $A_j^{(i)}$ be the number of times the vector $\boldsymbol{\theta_j}$ appears as a creator of genotype $\boldsymbol{g_i}$. Clearly, $A_j^{(i)}$ can be 0, 1, or 2. The log likelihood can be written as:

$$\begin{aligned}l(M) &= \sum_{i=1}^{n}[\log\alpha_{I_i}+\log\alpha_{J_i}+\log f(\boldsymbol{g_i};\boldsymbol{\theta_{I_i}},\boldsymbol{\theta_{J_i}})]\\ &= \sum_{i=1}^{n}[\sum_{a=1}^{k}A_a^{(i)}\log\alpha_a + \sum_{1\le a<b\le k}A_a^{(i)}A_b^{(i)}\log f(\boldsymbol{g_i};\boldsymbol{\theta_a},\boldsymbol{\theta_b})\\ &\quad + \sum_{a:\ A_a^{(i)}=2}\log f(\boldsymbol{g_i};\boldsymbol{\theta_a},\boldsymbol{\theta_a})].\end{aligned}$$

Let $I_{\{A_a^{(i)}=2\}}$ be an indicator random variable for the event $A_a^{(i)} = 2$. Then we can replace the last sum in $l(M)$ by $\sum_{a=1}^{k} I_{\{A_a^{(i)}=2\}}\log f(\boldsymbol{g_i};\boldsymbol{\theta_a},\boldsymbol{\theta_a})$. Since I_i and J_i, for $1\le i\le n$, are unknown, we use the EM approach (see, e.g., [17]). We denote the set of parameters by $\vartheta \equiv \{\alpha_i, \boldsymbol{\theta_i}\colon 1\le i\le k\}$. Given an initial set of parameters ϑ_0, we want to find another set of parameters ϑ of higher likelihood. This can be done by maximizing the conditional expectation:

$$\begin{aligned}Q_{M,\vartheta_0}(\vartheta) = \mathbb{E}_{\vartheta_0}[l|M] &= \sum_{i=1}^{n}[\sum_{a=1}^{k}\mathbb{E}_{\vartheta_0}[A_a^{(i)}|\boldsymbol{g_i}]\log\alpha_a\\ &\quad + \sum_{1\le a<b\le k}\mathbb{E}_{\vartheta_0}[A_a^{(i)}A_b^{(i)}|\boldsymbol{g_i}]\log f(\boldsymbol{g_i};\boldsymbol{\theta_a},\boldsymbol{\theta_b})\\ &\quad + \sum_{a=1}^{k}\mathbb{E}_{\vartheta_0}[I_{\{A_a^{(i)}=2\}}|\boldsymbol{g_i}]\log f(\boldsymbol{g_i};\boldsymbol{\theta_a},\boldsymbol{\theta_a})].\end{aligned}$$

In order to find $\arg\max_\vartheta Q_{M,\vartheta_0}(\vartheta)$, we need that $\forall i,j:\ 1\le i\le k;\ 1\le j\le m$, $\frac{\partial Q}{\partial\alpha_i}=0$ and $\frac{\partial Q}{\partial\theta_{i,j}}=0$.

Expectation:
The first step is to find $\boldsymbol{\alpha}$, such that Q is maximized. The conditional probabilities are:

$$\begin{aligned}P_{\vartheta_0}[A_j^{(i)}=1|\boldsymbol{g_i}] &= \frac{\sum_{1\le x\le k,\ x\ne j}2\alpha_x\alpha_j f(\boldsymbol{g_i};\boldsymbol{\theta_x},\boldsymbol{\theta_j})}{\sum_{x=1}^{k}\sum_{y=1}^{k}\alpha_x\alpha_y f(\boldsymbol{g_i};\boldsymbol{\theta_x},\boldsymbol{\theta_y})},\\ P_{\vartheta_0}[A_j^{(i)}=2|\boldsymbol{g_i}] &= \frac{\alpha_j\alpha_j f(\boldsymbol{g_i};\boldsymbol{\theta_j},\boldsymbol{\theta_j})}{\sum_{x=1}^{k}\sum_{y=1}^{k}\alpha_x\alpha_y f(\boldsymbol{g_i};\boldsymbol{\theta_x},\boldsymbol{\theta_y})}.\end{aligned} \quad (1)$$

We use Equations (1) to calculate the conditional expectation:

$$\mathbb{E}_{\vartheta_0}[A_j^{(i)}|\boldsymbol{g_i}] = P_{\vartheta_0}[A_j^{(i)}=1|\boldsymbol{g_i}] + 2P_{\vartheta_0}[A_j^{(i)}=2|\boldsymbol{g_i}].$$

The requested α_j can then be written as follows:

$$\alpha_j = \frac{1}{2n}\sum_{i=1}^{n}\mathbb{E}_{\vartheta_0}[A_j^{(i)}|\boldsymbol{g_i}].$$

In order to calculate the vectors $\boldsymbol{\theta_i}$ for $1\le i\le k$, we first need to get the conditional expectations:

$$\begin{aligned}\mathbb{E}_{\vartheta_0}[A_a^{(i)}A_b^{(i)}|\boldsymbol{g_i}] &= P_{\vartheta_0}[A_a^{(i)}=1, A_b^{(i)}=1|\boldsymbol{g_i}]\\ &= \frac{2\alpha_a\alpha_b f(\boldsymbol{g_i};\boldsymbol{\theta_a},\boldsymbol{\theta_b})}{\sum_{x=1}^{k}\sum_{y=1}^{k}\alpha_x\alpha_y f(\boldsymbol{g_i};\boldsymbol{\theta_x},\boldsymbol{\theta_y})},\\ \mathbb{E}_{\vartheta_0}[I_{\{A_a^{(i)}=2\}}|\boldsymbol{g_i}] &= P_{\vartheta_0}[A_a^{(i)}=2|\boldsymbol{g_i}].\end{aligned} \quad (2)$$

Maximization:
Now $\frac{\partial Q}{\partial\theta_{i,j}}$ can be calculated, using Equations (2):

$$\begin{aligned}&\frac{\partial Q}{\partial\theta_{i,j}} =\\ &\sum_{s=1}^{n}\Bigg[\sum_{1\le a\le k,\ a\ne i}\mathbb{E}_{\vartheta_0}[A_a^{(s)}A_i^{(s)}|\boldsymbol{g_s}]\cdot\begin{cases}\frac{1}{\theta_{i,j}-1} & g_{s,j}=0\\ \frac{1}{\theta_{i,j}} & g_{s,j}=1\\ \frac{1-2\theta_{a,j}}{\theta_{a,j}+\theta_{i,j}-2\theta_{a,j}\theta_{i,j}} & g_{s,j}=2\end{cases} +\\ &+\mathbb{E}_{\vartheta_0}[I_{\{A_a^{(s)}=2\}}|\boldsymbol{g_s}]\cdot\begin{cases}\frac{2}{\theta_{i,j}-1} & g_{s,j}=0\\ \frac{2}{\theta_{i,j}} & g_{s,j}=1\\ \frac{1-2\theta_{i,j}}{\theta_{i,j}-\theta_{i,j}^2} & g_{s,j}=2\end{cases}\Bigg].\end{aligned}$$

An inspection of the system of equations $\frac{\partial Q}{\partial\theta_{i,j}}=0$ for all $\theta_{i,j}$ reveals that for each j, the set of equations for $\{\theta_{i,j}: 1\le$

$i \leq k\}$ can be solved separately. In other words, for each j we have k polynomials with k variables: $\{\theta_{i,j} | 1 \leq i \leq k\}$. These equations can be solved numerically in practice, since k is assumed to be small.

Using this approach, we iteratively recalculate the parameters of the model, until convergence of the likelihood to a local maximum. Once the parameters are found, resolving is performed as follows: for each genotype $\boldsymbol{g_i}$, we find $P_\vartheta[A_a^{(i)} = 1, A_b^{(i)} = 1|\boldsymbol{g_i}]$ and $P_\vartheta[A_a^{(i)} = 2|\boldsymbol{g_i}]$, for each a and b. The indices of the creators of $\boldsymbol{g_i}$ are then determined by $\arg\max\{\max_{a \neq b} P_\vartheta[A_a^{(i)} = 1, A_b^{(i)} = 1|\boldsymbol{g_i}], \max_a P_\vartheta[A_a^{(i)} = 2|\boldsymbol{g_i}]\}$. Once the creators $\boldsymbol{\theta_a}$ and $\boldsymbol{\theta_b}$ of genotype $\boldsymbol{g_i}$ are known, its alleles at each heterozygote read j are $h_{i,j}^a = 1$, $h_{i,j}^b = 0$ if $\theta_{a,j} > \theta_{b,j}$, and $h_{i,j}^a = 0$, $h_{i,j}^b = 1$, otherwise.

Each of the common haplotypes is represented by a vector of probabilities $\boldsymbol{\theta_i}$. The corresponding binary common haplotype vector $\widehat{\boldsymbol{\theta}}_{\boldsymbol{i}}$ is obtained by rounding: $\widehat{\theta}_{i,j} = 0$ if $\theta_{i,j} \leq 0.5$ and $\widehat{\theta}_{i,j} = 1$ otherwise.

3.1 Finding the Number of Common Haplotypes in Each Block

The calculations of the maximum likelihood solution assume that k is known. In real biological data, we know that k is small, but its value is unknown. To overcome this obstacle, we calculate the likelihood $L(M, k)$ of a block with k common haplotypes, for $k = 1, \ldots, u$, where u is a small number (usually 5). It is easy to see that $L(M, i)$ is monotone non-decreasing in i. Let $\Delta(M, k) := L(M, k + 1) - L(M, k)$. In practice, when k exceeds the correct number of common haplotypes, $\Delta(M, k)$ becomes small. Thus, we choose the first k such that $\Delta(M, k) \leq \epsilon$, where ϵ is a parameter of the algorithm.

3.2 Finding the Blocks

To find the optimal block partition, we seek one that maximizes the overall likelihood of the data. The procedure is straightforward dynamic programming as in [27]. We first calculate for each j and for each $i > j$ the value l_{ji}, the log likelihood of the best solution forming a single block spanning columns i through j, as described above. Let T_i be the maximum log likelihood of a multi-block solution on the submatrix of M induced on the columns $1, \ldots, i$, where $T_0 = 0$. Then the following recursive formula is used to compute T_i:

$$T_i = \max_{0 \leq j \leq i-1} \{T_j + l_{ji}\} .$$

T_m gives the total log likelihood of the complete multi-block solution.

3.3 Matching Pairs of Blocks

So far, we have shown how to find the haplotypes of each individual within each block. This determines which alleles within the block appear together on the same chromosome. Our next challenge is to perform a similar task on the inter-block level, i.e., to determine for each individual which of the two haplotypes in each block occur on the same chromosome, and in this way to determine its complete chromosome pair. We call this problem *matching pairs of blocks*. If there are b blocks and the two haplotypes within each of them are distinct, then there are 2^{b-1} possible matchings. We seek a simultaneous solution for all individuals which will be "best" in a precise sense. This problem was presented in [5], where a combinatorial algorithm was proposed for solving it.

Our solution for the problem will be based on the observation that common haplotypes tend to pair unevenly across block boundaries [4]. Specifically, a common haplotype in one block may tend to appear on the same chromosome with another common haplotype in the next one, forming stretches that join together common haplotypes in several blocks.

The problem is solved in the following way: Let t and $t + 1$ be the indices of two consecutive blocks. Let $\{a_i^t, b_i^t\}$ and $\{a_i^{t+1}, b_i^{t+1}\}$ be the common haplotypes in blocks t and $t + 1$ respectively, for genotype $\boldsymbol{g_i}$. These can be matched as $\{(a_i^t, a_i^{t+1}), (b_i^t, b_i^{t+1})\}$ or $\{(a_i^t, b_i^{t+1}), (b_i^t, a_i^{t+1})\}$. In block t there are up to k different common haplotypes, denoted by $\{a_s^t\}_{1 \leq s \leq k}$. Recall, that the probability that the common haplotype a appears in a genotype is α_a. Let $P_{trans}(a, b)$ be the transition probability from common haplotype a in block t to common haplotype b in block $t+1$, i.e., the probability that if common haplotype a appears in block t, then common haplotype b appears in block $t + 1$ on the same chromosome. Denote by A_i^t an indicator random variable that has value 1 iff the matching for the i^{th} genotype is $\{(a_i^t, a_i^{t+1}), (b_i^t, b_i^{t+1})\}$, and let $\bar{A}_i^t = 1 - A_i^t$. Over all, with respect to blocks t and $t + 1$, the log likelihood function is:

$$l = \sum_{i=1}^{n} [\ln \alpha_{a_i^t} + \ln \alpha_{b_i^t} + A_i^t \ln(P_{trans}(a_i^t, a_i^{t+1}) P_{trans}(b_i^t, b_i^{t+1})) + \bar{A}_i^t \ln(P_{trans}(a_i^t, b_i^{t+1}) P_{trans}(b_i^t, a_i^{t+1}))].$$

Here too we find the parameters $\{P_{trans}(a_i, b_j)\}$ using maximum likelihood estimation by an EM approach: The transition probabilities can be obtained from $\{\mathbb{E}[A_i^t]\}_{1 \leq i \leq n}$ and vise versa by closed formulas, so calculating $\{\mathbb{E}[A_i^t]\}_{1 \leq i \leq n}$ and the transition probabilities can be performed iteratively, until convergence of the likelihood. Once the transition probabilities are known, the decision which of the two possible pairs matching to choose can be done for each $1 \leq i \leq n$, according to:

$$\arg\max\{P_{trans}(a_i^t, a_i^{t+1}) P_{trans}(b_i^t, b_i^{t+1}), P_{trans}(a_i^t, b_i^{t+1}) P_{trans}(b_i^t, a_i^{t+1})\}.$$

If in some block the two common haplotypes of a certain genotype originate from the same common haplotype, then the two possible matchings are identical. In that case, we preform the procedure on the haplotypes in the closest flanking blocks that have distinct common haplotypes (using all the haplotypes in theses blocks). This heuristic procedure aims to reveal longer range dependency between blocks.

4. RESULTS

Our algorithm was implemented in a C++ program called GERBIL (GEnotype Resolution and Block Identification using Likelihood). In the implementation, all initial parameters are chosen at random, the complete procedure is repeated 100 times and the maximal likelihood solution is selected. Running times on 2 GHz Pentium PC were less than 1 minute for resolving one block of 20 SNPs with 150 genotypes. Partitioning into blocks and phasing for a few hundred SNPs took several hours. GERBIL will be available in the future at www.cs.tau.ac.il\~rshamir.

We applied GERBIL to two published data sets, and compared the results to prior analysis of the same data. We describe how we dealt with missing data entries and outline

the methods that we used to evaluate the results, and then present the results on each data set.

Missing entries in the genotype matrix were completed in the original data, before the algorithm is performed, by the following heuristic. For each missing entry, we look at the window that spans 15 sites before and 15 sites after this site, and seek the closest other genotype within this window, where closeness is measured by the number of matching entries. The missing entry is then completed as the site value in that closest genotype. For an alternative approach to complete missing entries see [5].

4.1 Measures for Comparing Solutions

The data set of Daly et al. [4], on which we tested GERBIL, could be resolved to a large extent using pedigrees. The pedigree-based solution was assumed to be correct, and we used three methods for comparing different phasing solutions to it:

1. **Block Error Rate** - This test measures the error rate in a solution of a specific single block, w.r.t. the true solution. Let the two true haplotypes for genotype $\boldsymbol{g_i}$ be $\boldsymbol{t_1^i}, \boldsymbol{t_2^i}$ and let the two inferred haplotypes be $\boldsymbol{h_1^i}, \boldsymbol{h_2^i}$. Define the number of errors in genotype g_i as $e_i = \frac{1}{2}\min\{[d(\boldsymbol{t_1^i}, \boldsymbol{h_1^i}) + d(\boldsymbol{t_2^i}, \boldsymbol{h_2^i})], [d(\boldsymbol{t_1^i}, \boldsymbol{h_2^i}) + d(\boldsymbol{t_2^i}, \boldsymbol{h_1^i})]$, where d is the Hamming distance. If the number of heterozygote sites in genotype $\boldsymbol{g_i}$ is r_i, then the error rate is $\frac{\sum_{i=1}^{n} e_i}{\sum_{i=1}^{n} r_i}$.

2. **Average Block Error Rate** - This test measures the error rate in a multi-block solution with respect to the true solution. Let e^j be the total number of errors in block j (the numerator in the expression above), and let r^j be the total number of heterozygote sites in the the genotypes in block j (the denominator in the expression above), and let B be the number of blocks in the matrix. The measure is $\frac{\sum_{j=1}^{B} e^j}{\sum_{j=1}^{B} r^j}$.

3. **Switch Test** [14] - This test assumes matching of block pairs has been performed. It compares two solutions, $h = (h_1, h_2), t = (t_1, t_2)$ each of which is a pair of complete haplotype rows of sister chromosomes. Define the number of switches between h and t as the minimum number of times one has to 'jump' from one haplotype in h to the other in order to obtain t, when scanning the haplotypes from end to end. An example of switch test is shown in Table 1. This test is arguably more adequate than just counting the number of errors as above, since a whole group of errors can be corrected by changing the single decision to switch the group with that on the other haplotype. The total number of switches divided by the total number of heterozygote sites is called the *switch rate*.

4.2 Chromosome 5p31 Genotypes

The data set of Daly et al. [4] contains 129 pedigrees of father, mother and child, each genotyped at 103 SNP sites in chromosome 5p31. The original children data contain 13287 typed sites, of which 3873 (29%) are heterozygote alleles and 1334 (10%) are missing. After pedigree resolving, only 4315 (16%) of the 26574 single SNPs remained unknown (unresolved or missing data). Following [5], we used only the genotypes of the children and compared our solution to the pedigree-based solution from [4].

t	h
1: 11110001111	1: 00000000000
2: 00001110000	2: 11111111111

Table 1: Example of switch test. The number of switches that has to be done on h in order to obtain t is 2. Viewed as a single block, the minimum number of errors between the solutions is 3.

As a first step we applied GERBIL separately on each of the original blocks reported by Daly et al. The differences between the common haplotypes calculated by us and the true ones (which were constructed using the pedigrees) are minor: only in 4 common haplotypes there is a difference. In total, 10 bases out of 344 (2.9%) differed.

The results of GERBIL for resolving and block partitioning are presented in Table 2. We identified 8 blocks with total log likelihood of -4112.45, compared with log likelihood of -4647.36 of the solution of Daly et al., using the optimal model parameters for that solution. In each block 4-5 common haplotypes were found. The total number of switches in the haplotype matrix was 115 (3%). The average block error rate was 0.7%.

We compared the performance of our algorithm to two previously published phasing algorithms: the algorithm of Eskin et al. [5], which uses the perfect phylogeny criterion (see also [1]), and HaploBlock of Greenspan and Geiger [8], which resolves the genotypes by constructing a Bayesian network. The solution of [5] was taken from [25], and the solution of [8] was obtained by running HaploBlock [26] on the raw data. Table 3 compares the results of the three algorithms using the different error measures. In the switch test criterion, GERBIL made 29% less errors than Eskin et al., and 8% fewer errors than HaploBlock. With respect to average block error, GERBIL made 43% less errors than Eskin et al., and 62% less errors than HaploBlock.

Since HaploBlock partitioned the data into four blocks only, one could argue that the better results that we obtained were due to the increased number of blocks. To test it, we ran GERBIL on the blocks of HaploBlock, applying only the resolving procedure on the given partition, and the number of common haplotypes was assigned to be $k = 5$. The results are presented in Table 4. Our average block error rate was 30% less than HaploBlock.

4.3 OPRM1 Genotypes and Phenotypes

The data set of Hoehe et al. [10] consists of 172 genotypes and 25 SNPs. The SNPs are from the human μ opioid receptor gene (OPRM1) on chromosome 6, which is known to be related to morphine tolerance and dependence [16]. For each individual, its disease phenotype to substance (heroine and cocaine) dependence is available (case / control). No pedigree information is available and thus the true haplotypes are not known. Instead, we ran GERBIL on the data, with and without block partitioning, and tried to find association with the disease phenotype using the resolved haplotypes.

We first used GERBIL to resolve the data as a single block (Table 5). In all GERBIL runs we allowed four common haplotypes. In order to check for association between the resolved haplotypes and disease phenotype, we calculated

SNPs	Common Haplotypes of GERBIL	α_i	Number of Errors	Number of Heterozygotes	Error Rate
1 - 14	GGACAACCGTTACG	0.83	0	450	0
	AATTCGTGGCCCAA	0.13			
	AATTCGTGGTTACG	0.02			
	GGACAACCGCCCAA	0.01			
15 - 28	CCGGAGACGACGCG	0.55	6	583	0.0103
	TGACTGGTCGCTGC	0.24			
	CCGCAGACGACTGC	0.19			
	TGGCAGGTCGCTGC	0.02			
29 - 38	CCCGGATCCA	0.72	1	393	0.0025
	TATAACCGCG	0.17			
	CCCAACCCCA	0.06			
	CCCAACCCAA	0.05			
39 - 44	GCCCGA	0.54	4	210	0.0190
	CCCTGA	0.19			
	CTCTGA	0.14			
	CCATAC	0.13			
45 - 72	TCCCTGCTTACGGTGCAGTGGCACGTAT	0.7	2	942	0.0021
	CTCCCATCCATCATGGTCGAATGCGTAC	0.24			
	CCATCACTCCCCAGACTGTGATGTTAGT	0.05			
73 - 91	TGCACCGTTTAGCACAACA	0.59	9	711	0.0127
	ATTAGTGTTTGACGCGGTG	0.16			
	ATCAGTGATTAGCACGGTG	0.13			
	ATCAGTGATTAGCACGGTG	0.07			
	ATCTCTAATTGGCGTGACG	0.05			
92 - 98	GTTCTGA	0.57	4	294	0.0136
	TGTGTAA	0.28			
	TGTGCGG	0.15			
99 - 103	CGGCG	0.45	0	290	0
	TATAG	0.42			
	TATCA	0.14			
total			26	3873	0.0067

Table 2: Results of GERBIL in phasing and block partitioning on the data of Daly et al.

χ^2 scores, for each common haplotypes vs. the rest, and also for all the haplotypes together. The results of the association tests are summarized in Table 7. For all the common haplotypes together, the p-value was 0.02378; for the third common haplotype, the p-value was 0.0234.

Next, we ran GERBIL with blocks partitioning. We discovered two blocks (Table 6). We checked disease association in the same fashion. The first block was clearly associated to the disease with p-value of 0.0031. In the second block, only the second haplotype was associated with p-value of 0.0385. It is quite clear that association is much more prominent in the first block.

Hoehe et al. resolved the genotypes using the MULTIHAP [24] software, which is based on [6]. Then, the haplotypes were hierarchically clustered into a tree using an agglomerative nearest neighbor approach. The p-values of comparisons of haplotype frequencies and of cases and controls were calculated between the clusters calculated at each level of the hierarchical clustering. The lowest p-value which was achieved was 0.017.

In order to compare the significane of the two solutions, one has to correct for multiple testing. Since we performed eight different tests for two blocks (four tests in each block), after Bonferroni correction our p-value, for common haplotype number 4 in block number 1 was 0.0360. Hoehe et al. performed n different tests, where n is the number of haplotypes. To correct Hoehe et al. score for multiple testing, we multiplied their score by the number of distinct *groups* of haplotypes in their dendrogram, which was 5. Notably, this correction is much less strict than ours. Thus, after multiple testing correction, Hoehe et al. p-value was 0.0850, which is 2.3-times larger than ours. Hence, our solution achieves a much better statistical significance.

5. CONCLUDING REMARKS

We have introduced a new stochastic model for genotype generation, based on the biological finding that genotypes can be partitioned into blocks, and in each block, a small number of common haplotypes is found. Our model defined the notion of a probabilistic common haplotype, which might have different forms in different genotypes, thereby accommodating errors and rare mutations. We were able to define a likelihood function for this model. Finding the optimal parameters of the model was achieved using an EM algorithm, according to the maximum likelihood approach.

In tests on real data, our algorithm gave more accurate results than two recently published phasing algorithms [5, 8]. The haplotypes and blocks identified by the algorithm on case/control genotype data of the OPRM1 gene [10] led to finding more significant association with substance abuse phenotype.

Although our model finds a block partitioning that maximizes the overall likelihood, it performs resolving and block partitioning first, and then matches pairs of blocks along the chromosome as a postprocessing step. We plan to unite those two steps into a complete, single model. An additional open problem is to treat the missing data as a part of the model. We believe that solving these problems will lead to additional improvement in performance. Finally, the block patterns are sometimes unclear, and it has been argued that less restrictive models of haplotypes generation are needed (e.g., [23, 2]). We intend to generalize our approach in this spirit.

Acknowledgments

Ron Shamir was supported by a grant from the Israel Science Foundation (grant 309/02). We wish to thank Margret

Hoehe for providing us with the OPRM1 data and for useful comments. We thank Gideon Greenspan, Dan Geiger, Yoav Benjamini, Elazar Eskin and Koby Lindzen for helpful discussions and comments.

6. REFERENCES

[1] V. Bafna, D. Gusfield, G. Lancia, and S. Yooseph. Haplotyping as perfect phylogeny: A direct approach. Technical Report UCDavis CSE-2002-21, 2002.

[2] V. Bafna, B. V. Halldorsson, R. Schwartz, A. Clark, and S. Istrail. Haplotyles and informative SNP selection algorithms: Don't block out information. In *Proceedings of The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 19–27, 2003.

[3] A. Clark. Inference of haplotypes from PCR-amplified samples of diploid populations. *Molecular Biology and Evolution*, 7(2):111–22, 1990.

[4] M. J. Daly et al. High-resolution haplotype structure in the human genome. *Nature Genetics*, 29(2):229–232, 2001.

[5] E. Eskin, E. Halperin, and R. M. Karp. Large scale reconstruction of haplotypes from genotype data. In *Proceedings of The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 104–113, 2003.

[6] L. Excoffier and M. Slatkin. Maximum-likelihood estimation of mollecular haplotype frequencies in a diploid population. *Molecular Biology and Evolution*, 12(5):912–7, 1995.

[7] S. B. Gabriel et al. The structure of haplotype blocks in the human genome. *Science*, 296:2225–2229, 2002.

[8] G. Greenspan and D. Geiger. Model-based inference of haplotype block variation. In *Proceedings of The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 131–137, 2003.

[9] D. Gusfield. Haplotyping as perfect phylogeny: conceptual framework and efficient solutions. In *Proceedings of The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 166–175, 2002.

[10] M. R. Hoehe et al. Sequence variability and candidate gene analysis in complex disease: association of μ opioid receptor gene variation with substance dependence. *Human Molecular Genetics*, 9:2895–2908, 2000.

[11] G. Kimmel, R. Sharan, and R. Shamir. Identifying blocks and sub-populations in noisy SNP data. In *proceedings of the Third Workshop on Algorithms in Bioinformatics (WABI)*, pages 303–319, 2003.

[12] M. Koivisto et al. An MDL method for finding haplotype blocks and for estimating the strength of haplotype block boundaries. In *Proceedings of the Pacific Symposium on Biocomputing (PSB)*, volume 8, pages 502–513, 2003.

[13] L. Kruglyak and D. A. Nickerson. Variation is the spice of life. *Nature Genetics*, 27:234–236, 2001.

[14] S. Lin, D. Cutler, M. Zwick, and A. Chakravarti. Haplotype inference in random population samples. *American Journal of Human Genetics*, 71:1129–1137, 2002.

[15] J. Long et al. An EM algorithm and testing strategy for multiple-locus haplotypes. *American Journal of Human Genetics*, 56(3):799–810, 1995.

[16] H. W. Matthes et al. Loss of morphine-induced analgesia, reward effect and withdrawal symptoms in mice lacking the mu-opioid-receptor gene. *Nature*, 383:819–823, 1996.

[17] G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. John Wiley and Sons, inc., 1997.

[18] T. Niu et al. Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms. *American Journal of Human Genetics*, 70(1):157–69, 2002.

[19] N. Patil et al. Blocks of limited haplotype diversity revealed by high-resolution scanning of human chromosome 21. *Science*, 294:1719–1723, 2001.

[20] R. Sachidanandam et al. A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms. *Nature*, 291:1298–2302, 2001.

[21] M. Stephens et al. A new statistical method for haplotype reconstruction from population data. *American Journal of Human Genetics*, 68(4):978–89, 2001.

[22] C. Venter et al. The sequence of the human genome. *Science*, 291:1304–51, 2001.

[23] J. Wall et al. Assessing the performance of the haplotype block model of linkage disequilibrium. *American Journal of Human Genetics*, 73(3):502–515, 2003.

[24] `http://mahe.bioinf.mdc-berlin.de`.

[25] `http://www1.cs.columbia.edu/compbio/hap/`.

[26] `http://www.cs.technion.ac.il/Labs/cbl`.

[27] K. Zhang, M. Deng, T. Chen, M. Waterman, and F. Sun. A dynamic programming algorithm for haplotype block partitioning. *Proc. Natl. Acad. Sci. USA*, 99(11):7335–9, 2002.

[28] K. Zhang et al. Dynamic programming algorithms for haplotype block partitioning: applications to human chromosome 21 haplotype data. In *Proceedings of The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 332–340, 2003.

Algorithm	GERBIL	Eskin et al.	HaploBlock
Total Number of Errors	26	46	69
Average Block Error Rate	0.0067	0.0119	0.0178
Switch Test	115	163	125
Switch Rate	0.0297	0.0421	0.0323

Table 3: Performance of three algorithms for phasing and block partitioning. The error measures evaluate the solutions by GERBIL, by Eskin et al. and by Greenspan and Geiger, on the data of Daly et al., compared to the true solution.

Algorithm	Block:	1-24	25-36	37-91	92-103	total
	Number of Heterozygotes	846	537	1906	584	3873
GERBIL	Number of Errors	13	0	29	6	48
	Block Error Rate	0.0154	0	0.0152	0.0103	0.0124
HaploBlock	Number of Errors	28	3	31	7	69
	Block Error Rate	0.0331	0.0056	0.0163	0.0120	0.0178

Table 4: Resolving performance: Comparison of GERBIL and HaploBlock on the data of Daly et al. Here GERBIL used the blocks produced by HaploBlock and performed genotypes resolving only.

Common Haplotype Number	Common Haplotype	α_i
1	000000000000000000000100	0.817
2	101010000000101000000100	0.077
3	101011000000101000001100	0.054
4	000000000000010000010101	0.053

A

Haplotype Number	1	2	3	4	Total
Cases	218	24	19	13	274
Controls	62	2	0	6	70
Total	280	26	19	19	344

B

Table 5: GERBIL results on the OPRM1 data, without block partitioning. A: the common haplotypes identified. B: frequencies of cases and controls for the resolved common haplotypes.

	Block 1: SNPs 1-17		Block 2: SNPs 18-25	
Common Haplotype Number	Common Haplotype	α_i	Common Haplotype	α_i
1	00000000000000000	0.50	00000100	0.85
2	00000000000000100	0.21	10000000	0.097
3	00000000001000000	0.15	00010101	0.035
4	10101000000010100	0.13	00010001	0.017

A

Haplotype Number	1	2	3	4	Total
Cases	137	54	40	43	274
Controls	30	24	14	2	70
Total	167	78	54	45	344

B

Haplotype Number	1	2	3	4	Total
cases	239	21	10	4	274
controls	55	11	2	2	70
total	294	32	12	6	344

C

Table 6: Results of GERBIL on the OPRM1 data, with block partitioning. A: common haplotypes identified by GERBIL. B, C: frequencies of common haplotypes in the resolved data in cases and controls. B: first block; C: second block.

Haplotype Checked	One Block	Two Blocks: First Block	Two Blocks: Second Block
1 vs. rest	0.0839	0.2859	0.0667
2 vs. rest	0.0955	0.0093	0.0385
3 vs. rest	0.0234	0.2676	0.747
4 vs. rest	0.211	0.0045	0.4255
all vs. all	0.02378	0.0031	0.1648

Table 7: Association test results on the OPRM1 data: P-value χ^2 test results on the haplotypes resolved by GERBIL, with and without block partitioning.

GERBIL: Genotype resolution and block identification using likelihood

Gad Kimmel* and Ron Shamir

School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel

Edited by Richard M. Karp, International Computer Science Institute, Berkeley, CA, and approved November 11, 2004 (received for review July 1, 2004)

The abundance of genotype data generated by individual and international efforts carries the promise of revolutionizing disease studies and the association of phenotypes with individual polymorphisms. A key challenge is providing an accurate resolution (phasing) of the genotypes into haplotypes. We present here results on a method for genotype phasing in the presence of recombination. Our analysis is based on a stochastic model for recombination-poor regions (''blocks''), in which haplotypes are generated from a small number of core haplotypes, allowing for mutations, rare recombinations, and errors. We formulate genotype resolution and block partitioning as a maximum-likelihood problem and solve it by an expectation-maximization algorithm. The algorithm was implemented in a software package called GERBIL (genotype resolution and block identification using likelihood), which is efficient and simple to use. We tested GERBIL on four large-scale sets of genotypes. It outperformed two state-of-the-art phasing algorithms. The PHASE algorithm was slightly more accurate than GERBIL when allowed to run with default parameters, but required two orders of magnitude more time. When using comparable running times, GERBIL was consistently more accurate. For data sets with hundreds of genotypes, the time required by PHASE becomes prohibitive. We conclude that GERBIL has a clear advantage for studies that include many hundreds of genotypes and, in particular, for large-scale disease studies.

haplotype | algorithm | phasing | single-nucleotide polymorphism | expectation maximization

Most variations in the DNA sequence among individuals are at single-base sites, in which more than one nucleic acid can be observed across the population. Such differences and their sites are called SNPs (1, 2). In most cases only two alternative bases (alleles) occur at a SNP site. The total number of common human SNPs is estimated to be ≈10 million (3). The sequence of alleles in contiguous SNP sites along a chromosomal region is called a haplotype. Identification of haplotypes is a central challenge of the International HapMap Project (www.hapmap.org), because of their expected importance in disease associations (4, 5).

Recent evidence suggests that haplotypes tend to be preserved along relatively long genomic stretches, with recombination occurring mostly in narrow regions (1, 2). The stretch of SNP sites between two neighboring recombination regions is called a block. The number of different haplotypes within each block that are observed in a population is very small: typically, some 70–90% of the haplotypes within a block are identical (or almost identical) to very few (two to five) distinct common haplotypes (1).

Several studies have concentrated on the problem of block identification and partitioning in a given data set of haplotypes: Zhang *et al.* (6, 7) defined a block to be an interval of SNPs that minimizes the number of tag SNPs. Koivisto *et al.* (8) used a minimum description length criterion for block definition. Kimmel *et al.* (9) minimized the total number of common haplotypes when errors and missing data are allowed. The same dynamic programming approach (6) was used in all of these studies, and the main difference is in the optimization criterion used within the dynamic programming computation.

Diploid organisms, like human, have two near-identical copies of each chromosome. Most experimental techniques for determining SNPs do not provide the haplotype information separately for each of the two chromosomes. Instead, they generate for each site an unordered pair of allele readings, one from each copy of the chromosome (10). The sequence of these pairs is called a genotype. A homozygous site in the genotype of an individual has two identical bases, whereas a heterozygous site has different bases on the two chromosomal copies at that site. The process of inferring the haplotypes from the genotypes is called phasing or resolving.

In the absence of additional information, each genotype in a population can be resolved in 2^{h-1} different ways, where h is the number of heterozygous sites in this genotype. Thus, one must use some model of how the haplotypes are generated to perform genotype resolving. Clark (11) proposed the first approach to haplotype resolution, which was parsimony-based. Likelihood-based expectation-maximization (EM) algorithms (12, 13) gave more accurate results. Stephens and coworkers (14, 15) and Niu *et al.* (16) proposed Markov chain Monte Carlo-based methods. A combinatorial model based on the perfect phylogeny tree assumption was suggested by Gusfield (17). By using this model, Eskin *et al.* (18) showed good performance on real genotypes with low error rates. Recently, Greenspan and Geiger (19) proposed an algorithm that performs resolution while taking into account the block structure. The method is based on a Bayesian network model.

In this study we provide an algorithm that solves block partitioning and phasing simultaneously. Our algorithm is based on a model for genotype generation. The model and preliminary analysis on its performance were reported in ref. 20. The model is based on a haplotype generation model, parts of which were suggested by Koivisto *et al.* (8). Within each block, we redefine common haplotypes in a probabilistic setting and seek a solution that has maximum likelihood, by using an EM algorithm. The model accounts for errors and rare haplotypes.

The algorithm was implemented in a software package called GERBIL (genotype resolution and block identification using likelihood). We applied GERBIL to three genotype data sets: on a data set from chromosome 5 (21) it outperformed HAPLOBLOCK (19) and HAP (18) and gave similar results to PHASE (14), with much shorter run times. On the data set of Gabriel *et al.* (2) and on data from the International HapMap Project (www.hapmap.org), the PHASE algorithm was slightly more accurate than GERBIL when allowed to run with default parameters, but required two orders of magnitude more time. We also simulated data with larger numbers of genotypes (500 to 1,000) based on real haplotypes. In such a scenario, when the number of geno-

This paper was submitted directly (Track II) to the PNAS office.

Abbreviations: GERBIL, genotype resolution and block identification using likelihood; EM, expectation-maximization.

*To whom correspondence should be addressed. E-mail: kgad@tau.ac.il.

www.pnas.org/cgi/doi/10.1073/pnas.0404730102

types increased, GERBIL had a clear advantage over PHASE, because the latter required a prohibitively long time (and was, in fact, unable to solve the larger data sets). The GERBIL software can be downloaded at www.cs.tau.ac.il/~rshamir/gerbil.

Unlike most former probabilistic approaches (12–14, 16), our algorithm reconstructs block partitioning and resolves the haplotypes simultaneously. As in refs. 14 and 19 haplotype similarity is taken into account. Although our approach has some resemblance to HAPLOBLOCK, there are significant differences. First, our approach computes the maximum likelihood directly and is not based on a Bayesian network. Second, once the model parameters are found, we solve the phasing problem directly to optimality, so that the likelihood function is maximized. In contrast, HAPLOBLOCK applies a heuristic to find the block partitioning, even though this partitioning is part of the model parameters. Third, our stochastic model allows a continuous spectrum of probabilities for each component in each common haplotype, whereas the HAPLOBLOCK software allows only two common probability values for all mutations. HAPLOBLOCK's model has the advantage of incorporating interblock transitions, whereas we handle them separately after the main optimization process.

Methods

We first concentrate on modeling and analysis of a single block. The input to the problem is n genotypes $g_1, \ldots, g_n$, each of which is an m-long vector of readings $g_i(1), \ldots, g_i(m)$ corresponding to the SNP sites. We assume that all sites have at most two different alleles and rename the two alleles arbitrarily as 0 and 1. The genotype readings are denoted by $g_i(j) \in \{0, 1, 2\}$. 0 and 1 stand for the two homozygous types $\{0, 0\}$ and $\{1, 1\}$, respectively, and 2 stands for a heterozygous type. A resolution of g_i is two m-long binary vectors g_i^1, g_i^2 satisfying $g_i^1(j) = g_i^2(j) = g_i(j)$ if $g_i(j) = 0$ or 1, and $g_i^1(j) \neq g_i^2(j)$ if $g_i(j) = 2$.

The Probabilistic Model. Our stochastic model for the genotypes generation is based on the observation that within each block the variability of haplotypes is limited (2, 21). The model assumes a set of common haplotypes that occur in the population with certain probabilities. The generation of a genotype is as follows: First, two of the common haplotypes are chosen. Second, the alleles at each site of the haplotypes are determined. Third, their confluence is formed. In our model, these common haplotypes are not deterministic. Instead, we use the notion of probabilistic common haplotype that has specific allele probabilities at each site. Such a haplotype is a vector, whose components are the probabilities of having the allele 1 in each site of a realization of that haplotype. Hence, a vector of only zeroes and ones corresponds to a standard (consensus) common haplotype, and a vector with fractional values allows for deviations from the consensus with certain (small) probabilities, independently for each site. In this way, a common haplotype may appear differently in different genotypes. A similar model was used in ref. 8 in the block partitioning of phased data. Note that the model makes the Hardy–Weinberg (22) assumption that mating is random. An illustration of the model appears in Fig. 1.

A more formal definition of the stochastic model is as follows. Assume that the genotype data contain only one block. Let k be the number of common haplotypes in that block. Let $\theta_{i1 \le i \le k}$ be the probability vectors of the common haplotypes, where $\theta_i = (\theta_{i,1}, \ldots, \theta_{i,m})$ and $\theta_{i,j}$ is the probability to observe 1 in the jth site of the ith common haplotype. (Consequently, $1 - \theta_{i,j}$ is the probability to observe 0 in that site.) Let $\alpha_i > 0$ be the probability of the ith common haplotype in the population, with $\sum_{i=1}^{k} \alpha_i = 1$. Each genotype g_t is generated as follows:

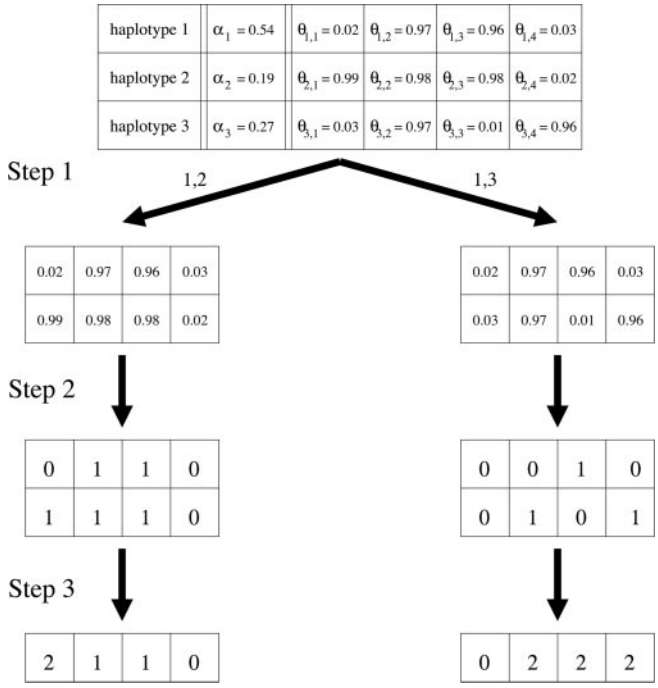


Fig. 1. An illustration of the probabilistic model. This model has three common haplotypes covering four SNPs. In the first step, pairs of the common haplotypes are chosen according to their probabilities α_i. In this example 1,2 and 1,3 are chosen. In the second step, the alleles at each site of the haplotypes are determined according to the probabilities $\theta_{i,j}$. In the third step, each genotype is formed by a confluence of two haplotypes created at the former step.

- Choose a number i between 1 and k according to the probability distribution $\{\alpha_1, \ldots, \alpha_k\}$. i is the index of the first common haplotype.
- The haplotype $(x_1, \ldots, x_m)$ is generated by setting, for each site j independently, $x_j = 1$ with probability $\theta_{i,j}$.
- Repeat the steps above for the second haplotype and form their confluence. The result is the genotype g_t.

For generating genotypes with several blocks, the process is repeated for each block independently.

EM. The two common haplotypes that were used to create a genotype are called its creators (the two may be identical). For a single genotype g_j, assuming its creators θ_a and θ_b are known, the probability of obtaining g_j is

$$f(\mathbf{g}_j;\ \boldsymbol{\theta}_a,\ \boldsymbol{\theta}_b) = \prod_{i=1}^{m} \begin{cases} (1 - \theta_{a,i})(1 - \theta_{b,i}) & g_{j,i} = 0 \\ \theta_{a,i}\theta_{b,i} & g_{j,i} = 1 \\ \theta_{a,i}(1 - \theta_{b,i}) + \theta_{b,i}(1 - \theta_{a,i}) & g_{j,i} = 2 \end{cases}.$$

We denote by I_i and J_i the index of the first and second creator of genotype g_i, respectively. The complete likelihood of all genotypes is

$$L(M) = \prod_{i=1}^{n} \alpha_{I_i}\, \alpha_{J_i} f(\mathbf{g}_i;\ \boldsymbol{\theta}_{I_i},\ \boldsymbol{\theta}_{J_i}).$$

Because I_i and J_i, for $1 \le i \le n$, are unknown, we use the EM approach (see, e.g., ref. 23) for iteratively increasing the likelihood. We get closed-form equations for the updating of α_i in each iteration, and we use numerical methods for updating the θ_i vectors. Thus, we iteratively recalculate the parameters of the model, until convergence of the likelihood to a local maximum.

For a detailed mathematical development of the solution for the optimization problem see ref. 20.

Block Partitioning and Finding the Parameter *k*. A simple approach that assumes independence between blocks would be to multiply the block likelihoods. However, as the parameter k is unknown, we use a minimal description length approach for finding the block partitioning and finding the parameter k for each of the blocks, in a similar fashion to ref. 8. For each pair of SNPs i,j, and for each possible k, we solve the problem as described above on the sites i through j, assuming they span a single block that contains k common haplotypes, and obtain the log likelihood score $L_{i,j}^{(k)}$. We also compute the description length $D_{i,j}^{(k)}$ of the model parameters. Note that when k is larger, the likelihood increases, but so does the description length of the model. The minimum description length of such a block is $M_{i,j}^{(k)} = D_{i,j}^{(k)} - L_{i,j}^{(k)}$. Let $k(i,j) = \arg\min_k M_{i,j}^{(k)}$. A partition P of the SNPs into b blocks is defined by $1 = i_1 < i_2 < \cdots < i_b \leq n$, where the tth block is $[i_t, i_{t+1} - 1]$. The score of such a partition according to the minimum description length criterion is $\Sigma_{s=1}^{b} M_{i_s,i_{s+1}-1}^{k(i_s,i_{s+1}-1)}$. Finding the optimal partition is solved by dynamic programming (cf. refs. 6 and 8).

Speedup. Instead of checking all possible k values, we first resolve the genotype data into a preliminary haplotype matrix H^P, by using a procedure that is based on our single block resolving algorithm (see *Appendix*). Now, for each candidate block $[i,j]$ the number of distinct haplotypes that appears in H^P more than once is used as an approximation for $k(i,j)$. Also, to save time, only candidate blocks of up to 30 SNPs are considered.

Pairing Haplotypes Across Block Boundaries. To construct a full chromosome sequence, one has to determine which alleles within the block appear together with alleles in the consecutive block, on the same haplotype. We call this problem matching pairs of blocks (cf. refs. 15 and 18). Our solution is based on the fact that specific common haplotypes in neighboring blocks tend to appear together on the same chromosome (21). For each pair of neighboring blocks and for each genotype, we simply choose the pairing that occurred more often in H^P.

Initialization. When applied to a block, the EM provides only a local optimum, and starting from good initial parameter values is critical both for the solution quality and the speed of the procedure. We generate such an initial solution as follows. We randomly permute the order of the genotypes and use Clark's inference algorithm (11) to resolve as many of the genotypes as possible. In case there is still some unresolved genotype, (either because of heterozygous sites or missing data entries), we resolve that genotype arbitrarily and reapply Clark's algorithm. This procedure ends when all genotypes are resolved. The next stage is to cluster the haplotypes around k common haplotypes (where k was already determined as described above). This requires finding a set C of k haplotypes such that $\Sigma_{i=1}^{n} \min_{h' \in C}(d(h_i, h'))$ is minimized, where $d(\ldots)$ denotes the Hamming distance. This subproblem is already hard (9), and we use the following heuristic procedure to solve it: We repeatedly select a random subset C' of k haplotypes $h_1, \ldots, h_k$, and each time calculate $\Sigma_{i=1}^{n} \min_{h' \in C'}(d(h_i, h'))$. This is repeated T times and the subset C that attains the minimum score is chosen. ($T = 100$ was used in practice.) We use the set C to construct the initial probabilistic common haplotypes for the EM procedure, in the following way: if h_i has value 1 in SNP j, we set $\theta_{i,j} = 0.999$, otherwise $\theta_{i,j} = 0.001$. The α_i value is proportional to the size of the cluster containing h_i. We also use H^P as an additional potential starting point.

Implementation. Our algorithm was implemented in a C++ program called GERBIL. Running time on a 2-GHz Pentium computer with 500 MB of memory is ≈1 min for resolving data with 100 SNPs and 150 genotypes. GERBIL is available for academic use at www.cs.tau.ac.il/~rshamir/gerbil.

Evaluating and Comparing Solutions. Let the true haplotypes for genotype g_i be $t = (t_1, t_2)$, and let the inferred haplotypes be $h = (h_1, h_2)$. Comparison of t and h can be done only for sites that are heterozygous and are resolved in t (e.g., because of pedigree data). Suppose we restrict t and h to these sites only, and obtain l-long vectors. We use the switch test (15, 24) to evaluate the accuracy of h. The test counts the number of times η_i we have to switch from reading h_1 to h_2 or back to obtain t_1, and divides the result by $d_i = l - 1$ (the maximum possible of switches for this genotype). The switch test value for a set of genotypes is $\Sigma_i \eta_i / \Sigma_i d_i$.

Results

Chromosome 5p31. The data set of Daly *et al.* (21) contains 129 pedigrees of father, mother, and child, each genotyped at 103 SNP sites in chromosome 5p31. The original children data contain 13,287 typed sites, of which 3,873 (29%) are heterozygous alleles and 1,334 (10%) are missing. After pedigree resolving, only 4,315 (16%) of the 26,574 SNPs remained unknown (unresolved or missing data). When applied to the children data GERBIL generated 18 blocks [compared with 11 blocks in the solution of Daly *et al.* (21)], and k ranged from 3 to 15 in the different blocks. The switch error rate was 3.3%.

We compared the performance of GERBIL to three previously published phasing algorithms: HAP (18), which uses the perfect phylogeny criterion (see also ref. 25) gave a switch error rate of 4.2%. HAPLOBLOCK (19), which uses a Bayesian network, gave a switch error rate of 3.6%. PHASE (14) (version 2.0.2 for Linux), which uses the coalescent as a Bayesian prior, gave a switch error rate of 3.1%. The run time of GERBIL was 1 min, whereas PHASE needed 4.1 h with its default parameters. When letting PHASE run a comparable time to GERBIL (2 min), PHASE achieved an error rate of 5.4%.

Yoruba Genotypes. Our second test focused on the Yoruba population genotypes from ref. 2. For this population there were parental genotypes that could be used to infer the true solution. We used 29 test genotypes (we removed one trio that had a high rate of missing entries). There are 52 different samples of size 13–114 SNPs from different regions of the human genome. GERBIL's average switch error rate was 15% with a total run time of 8 min over all samples. PHASE (with default parameters) gave more accurate results, averaging 12%, with a run time of 10.1 h. The relatively high values of switch error rate compared with the results above are caused by the combination of small sample size and high missing data rate (8%). The accuracy and speed on individual samples are displayed in Figs. 2 and 3, respectively. GERBIL was consistently 10–100 times faster.

HapMap Project Genotypes. Our third test used genotype data for 30 trios from the International HapMap Project (www.hapmap.org). As before, we used the parental information only to infer the true solution and applied the phasing algorithms to the children only. The missing entries rate in this data set was 1%, much smaller than in the former data sets.

We extracted four data sets of 70 SNPs from the beginning of each of the human chromosomes 1–22. The first data set in each chromosome contained SNPs 1–70, the second contained SNPs 71–140, etc. We applied both GERBIL and PHASE on all 88 data sets. The average switch error rate was 11% for GERBIL and 10% for PHASE. Overall run time was 31 min for GERBIL and 22.5 h for PHASE.

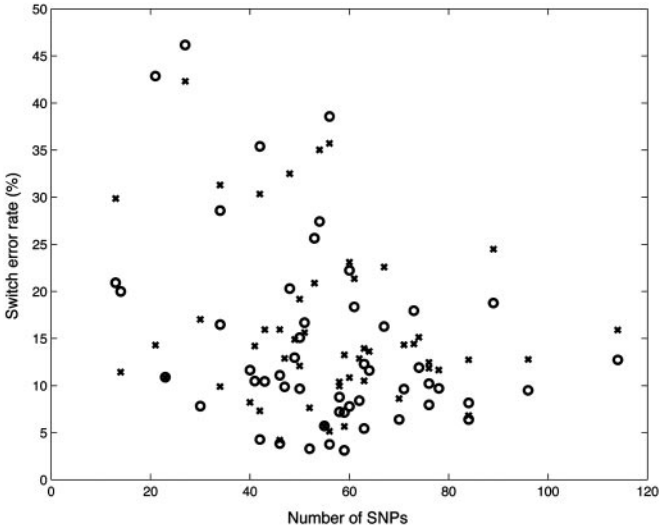


Fig. 2. Phasing accuracy on the Yoruba genotypes. The *x* axis shows the number of SNPs in each of the 52 data sets. The *y* axis shows the switch error rate of GERBIL (x) and PHASE (○) on each data set.

Large Simulated Data Sets. To assess our software on data sets with a larger number of genotypes, we simulated genotypes based on the known haplotypes of chromosome 5p31 (see above). We generated six different data sets by random sampling and pairing of haplotypes from the 258 known ones. These data sets contained 500, 600, 700, 800, 900, and 1,000 genotypes. The results of GERBIL and PHASE are presented in Table 1. On the larger data sets of 800, 900, and 1,000 genotypes, PHASE aborted after 12 min, because of memory allocation overload. Attempts to run PHASE on a larger memory machine (Pentium 3 with 2 gigabytes of memory) also were aborted. On the smaller data sets of 500, 600, and 700 genotypes, when giving PHASE the same amount of run time, GERBIL outperforms PHASE in accuracy. When using the default parameters of PHASE, the program provides more accurate results (1% vs. 3%), but requires considerably longer run times (≈3 days vs. <1 h).

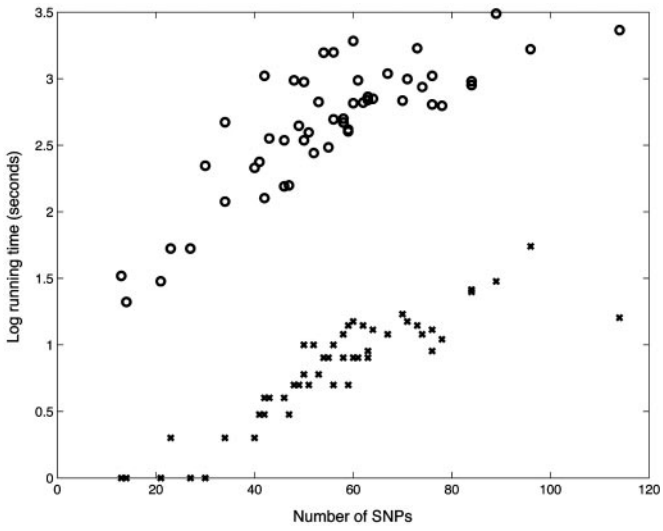


Fig. 3. Running times on the Yoruba genotypes. The *x* axis shows the number of SNPs in each data set. The *y* axis shows a logarithm (base 10) of running times (in seconds) of GERBIL (x) and PHASE (○) on each data set.

Table 1. Performance of GERBIL and PHASE on large data sets

	GERBIL		PHASE	
No. of genotypes	Switch error rate, %	Run time, min	Switch error rate, %	Run time, min
500	3	15	7	37
			1	4,711
600	3	20	6	47
			1	4,100
700	3	28	5	80
			1	4,525
800	3	36	No solution	
900	3	42	No solution	
1,000	3	59	No solution	

Results are shown for different numbers of simulated genotypes generated from true chromosome 5p31 haplotypes. All runs were on a Pentium 4 computer with 500 megabytes of memory. For PHASE, two runs were performed, one with the default parameters and one with a short running time comparable to GERBIL's. The PHASE processes that gave no solutions were terminated because of memory allocation overload. They also failed on a 2-gigabyte-memory machine.

Discussion

We have introduced an algorithm for haplotype resolution and block partitioning. The algorithm uses a stochastic model for genotype generation, based on the biological finding that genotypes can be partitioned into blocks of low recombination rate, and in each block, a small number of common haplotypes is found. Our model uses the notion of a probabilistic common haplotype, which can have different forms in different genotypes, thereby accommodating errors, rare recombination events, and mutations. We were able to define a likelihood function for this model. Finding the maximum-likelihood solution for genotype data under the model is achieved by using an EM algorithm. The algorithm was implemented in the GERBIL program.

In tests on real data, our algorithm gave more accurate results than two recently published phasing algorithms (18, 19). Most of our comparisons concentrated on PHASE (15), currently the leading algorithm for haplotyping. There are two performance criteria that should be considered in such a comparison. The first and foremost is accuracy, which we measured by using the switch test (15, 24). However, when a program becomes impractically slow as one attempts to use it on larger and larger problems, one should apply the criterion of speed and test the tradeoff between accuracy and speed. Hence, we ran PHASE in two modes: one that used similar running times to GERBIL, and another (default PHASE) that was run with the default parameters and required much longer run times. The tests covered 141 real data sets (2, 21) (www.hapmap.org), ranging in size between 29 and 129 genotypes and from 13 to 114 SNPs. When allowed similar run times, GERBIL was consistently more accurate than PHASE. Default PHASE was slightly more accurate than GERBIL but required two orders of magnitude more time (Fig. 3). The difference became more apparent on larger data sets containing 500 or more genotypes. On such data sets default PHASE required several days of computing time, and on 800 genotypes or more it completely failed to provide a solution (Table 1).

The next few years carry the promise of very large association studies that will use haplotypes extensively (26). Studies with 400–800 genotypes already have been reported (27), and studies with thousands of genotypes are envisioned (27). High-throughput genotyping methods are progressing rapidly (28). The number of SNPs typed is also likely to increase with technology improvements: DNA chips that can type >10,000 SNPs have been in use for over a year (29), and chips with

100,000 SNPs are already commercially available. Although clearly not all such SNPs need to be phased simultaneously, typing density would call for phasing hundreds of SNPs in up to a few thousands of genotypes. We believe that GERBIL has the potential to make a unique and important contribution to the analysis of such data.

Appendix: Constructing Preliminary Haplotype Matrix

The preliminary haplotype matrix H^P is constructed in the following way: for each genotype g and for each pair of sites i, j that are heterozygous in it, we want to score the two possible phasing solutions of the two sites. This is done by applying the EM algorithm (with $k = 4$) on the block $[i, i + 1, \ldots, j]$ from which other sites that are heterozygous in g are removed. The maximum-likelihood solution gives probabilities to the four possible haplotypes that could have generated g. For example, if the genotype is $g = 201102$, then the four possible haplotypes are $h_{00} = 001100$, $h_{11} = 101101$, $h_{01} = 001101$, and $h_{10} = 101100$. Denote the probabilities found by the algorithm for the corresponding common haplotypes by P_{00}, P_{11}, P_{01}, and P_{10}, respectively. (Note that these are probabilities of full haplotypes and not only of the pair of sites i and j.) We use these probabilities to calculate a score, which represents the certainty level in the more probable phasing solution. Specifically, we define $w(i, j) = n(P_{00}P_{11} - P_{01}P_{10})^2/(P_{00} + P_{01})(P_{10} + P_{11})(P_{00} + P_{10})(P_{01} + P_{11})$, where n is the number of haplotypes in the resolved block that match one of the possible haplotypes of g. [This measure is closely related to R^2 (e.g., ref. 30).] For each genotype, we build a complete weighted graph $G = (V, E)$, where each vertex corresponds to a heterozygous site, and the weight of the edge (v_1, v_2) is $w(v_1, v_2)$. Observe that any phasing solution is obtained according to some spanning tree of G. Hence, we find a maximum spanning tree of G, and it provides a most probable phasing solution for that genotype. In practice, we set $w(i, j) = 0$ if $|i - j| > 30$.

We thank Gideon Greenspan, Dan Geiger, Yoav Benjamini, Elazar Eskin, and Koby Lindzen for helpful discussions and comments. R.S. was supported by Israel Science Foundation Grant 309/02.

1. Patil, N., Berno, A. J., Hinds, D. A., Barrett, W. A., Doshi, J. M., Hacker, C. R., Kautzer, C. R., Lee, D. H., Marjoribanks, C., McDonough, D. P., *et al*. (2001) *Science* **294,** 1719–1723.
2. Gabriel, S. B., Schaffner, S. F., Nguyen, H., Moore, J. M., Roy, J., Blumenstiel, B., Higgins, J., DeFelice, M., Lochner, A., Faggart, M., *et al*. (2002) *Science* **296,** 2225–2229.
3. Kruglyak, L. & Nickerson, D. A. (2001) *Nat. Genet.* **27,** 234–236.
4. Martin, E. R., Lai, E. H., Gilbert, J. R., Rogala, A. R., Afshari, A. J., Riley, J., Finch, K. L., Stevens, J. F., Livak, K. J., Slotterbeck, B. D., *et al*. (2000) *Am. J. Hum. Genet.* **67,** 383–394.
5. Morris, R. W. & Kaplan, N. L. (2002) *Genet. Epidemiol.* **23,** 221–233.
6. Zhang, K., Deng, M., Chen, T., Waterman, M. S. & Sun, F. (2002) *Proc. Natl. Acad. Sci. USA* **99,** 7335–7339.
7. Zhang, K., Sun, F., Waterman, M. S. & Chen, T. (2003) in *Proceedings of The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)*, eds. Miller, W., Vingron, M., Istrail, S., Pevzner, P. & Waterman, M. (Association for Computing Machinery, New York), pp. 332–340.
8. Koivisto, M., Perola, M., Varilo, T., Hennah, W., Ekelund, J., Lukk, M., Peltonen, L., Ukkonen, E. & Mannila, H. (2003) in *Proceedings of the Pacific Symposium on Biocomputing (PSB 03)*, eds. Altman, R. B., Dunker, A. K., Hunter, L., Jung, T. A. & Kline, T. E. (World Scientific, Teaneck, NJ), Vol. 8, pp. 502–513.
9. Kimmel, G., Sharan, R. & Shamir, R. (2004) *INFORMS J. Comput.* **16,** 360–370.
10. Sachidanandam, R., Weissman, D., Schmidt, S. C., Kakol, J. M., Stein, L. D., Marth, G., Sherry, S., Mullikin, J. C., Mortimore, B. J., Willey, D. L., *et al*. (2001) *Nature* **291,** 1298–2302.
11. Clark, A. (1990) *Mol. Biol. Evol.* **7,** 111–122.
12. Excoffier, L. & Slatkin, M. (1995) *Mol. Biol. Evol.* **12,** 912–917.
13. Long, J., Williams, R. C. & Urbanek, M. (1995) *Am. J. Hum. Genet.* **56,** 799–810.
14. Stephens, M., Smith, N. J. & Donnelly, P. (2001) *Am. J. Hum. Genet.* **68,** 978–989.
15. Stephens, M. & Donnelly, P. (2003) *Am. J. Hum. Genet.* **73,** 1162–1169.
16. Niu, T., Qin, Z. S., Xu, X. & Liu, J. S. (2002) *Am. J. Hum. Genet.* **70,** 157–169.
17. Gusfield, D. (2002) in *Proceedings of the Sixth Annual International Conference on Research in Computational Molecular Biology (RECOMB 02)*, eds. Myers, G., Hannenhalli, S., Istrail, S., Pevzner, P. & Waterman, M. (Association for Computing Machinery, New York), pp. 166–175.
18. Eskin, E., Halperin, E. & Karp, R. M. (2003) in *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)*, eds. Miller, W., Vingron, M., Istrail, S., Pevzner, P. & Waterman, M. (Association for Computing Machinery, New York), pp. 104–113.
19. Greenspan, G. & Geiger, D. (2003) in *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)*, eds. Miller, W., Vingron, M., Istrail, S., Pevzner, P. & Waterman, M. (Association for Computing Machinery, New York), pp. 131–137.
20. Kimmel, G. & Shamir, R. (2004) in *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 04)*, eds. Gusfield, D., Bourne, P., Istrail, S., Pevzner, P. & Waterman, M. (Association for Computing Machinery, New York), pp. 2–9.
21. Daly, M. J., Rioux, J. D., Schaffner, S. F., Hudson, T. J. & Lander, E. S. (2001) *Nat. Genet.* **29,** 229–232.
22. Hardy, G. H. (1908) *Science* **18,** 49–50.
23. McLachlan, G. J. & Krishnan, T. (1997) *The EM Algorithm and Extensions* (Wiley, New York).
24. Lin, S., Cutler, D., Zwick, M. & Chakravarti, A. (2002) *Am. J. Hum. Genet.* **71,** 1129–1137.
25. Bafna, V., Gusfield, D., Lancia, G. & Yooseph, S. (2003) *J. Comput. Biol.* **10,** 323–340.
26. Shi, M. M. (2001) *Clin. Chem.* **47,** 164–172.
27. Nickerson, D. A., Taylor, S. L., Fullerton, S. M., Weiss, K. M., Clark, A. G., Stengard, J. H., Salomaa, V., Boerwinkle, E. & Sing, C. F. (2000) *Genome Res.* **10,** 1532–1545.
28. Tsuchihashi, Z. & Dracopoli, N. C. (2002) *Pharmacogenomics* **2,** 103–110.
29. Kennedy, G. C., Matsuzaki, H., Dong, S., Liu, W. M. Huang, J., Liu, G., Su, X., Cao, M., Chen, W., Zhang, J., *et al*. (2003) *Nat. Biotechnol.* **21,** 1233–1237.
30. Devlin, B. & Risch, N. (1995) *Genomics* **29,** 311–322.

JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 12, Number 10, 2005

Pp. 1243–1260

A Block-Free Hidden Markov Model for Genotypes and Its Application to Disease Association

GAD KIMMEL and RON SHAMIR

ABSTRACT

We present a new stochastic model for genotype generation. The model offers a compromise between rigid block structure and no structure altogether: It reflects a general blocky structure of haplotypes, but also allows for "exchange" of haplotypes at nonboundary SNP sites; it also accommodates rare haplotypes and mutations. We use a hidden Markov model and infer its parameters by an expectation-maximization algorithm. The algorithm was implemented in a software package called HINT (haplotype inference tool) and tested on 58 datasets of genotypes. To evaluate the utility of the model in association studies, we used biological human data to create a simple disease association search scenario. When comparing HINT to three other models, HINT predicted association most accurately.

Key words: haplotype, algorithm, single-nucleotide polymorphism, hidden Markov model, genotype, expectation maximization.

1. INTRODUCTION

MOST VARIATION IN THE DNA SEQUENCE among individuals is at specific positions, where more than one nucleic acid can be observed across the population. These positions are called *single nucleotide polymorphisms* (SNPs) (Patil *et al.*, 2001; Gabriel *et al.*, 2002). In almost all cases only two alternative bases (*alleles*) occur at a SNP site. The total number of common human SNPs is estimated to be about 10 million (Kruglyak and Nickerson, 2001; Botstein and Risch, 2003). The sequence of alleles in contiguous SNP sites along a chromosomal region is called a *haplotype*. Identification of haplotypes is a central challenge of the HapMap project (*www.hapmap.org*), due to their expected importance in disease associations (Martin *et al.*, 2000; Morris and Kaplan, 2002).

Diploid organisms, like human, have two homologous (nearly identical) copies of each chromosome (except the sex chromosome). Most experimental techniques for determining SNPs do not provide the haplotype information separately for each of the two chromosomes. Instead, they generate for each site an unordered pair of allele readings, one from each copy of the chromosome (cf. Sachidanandam *et al.* [2001]). The sequence of these pairs is called a *genotype*. A *homozygous* site in the genotype of an individual has two identical bases, while a *heterozygous* site has different bases on the two chromosomal copies at that site. The process of inferring the haplotypes from the genotypes is called *phasing* or *resolving*. Without additional information, each genotype with h heterozygous sites in a population can be resolved in 2^{h-1} different ways. Thus, one must use some model of how the haplotypes are generated in order to perform genotype resolving. Since Clark's introduction of the problem in 1990 (Clark, 1990), many studies

School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel.

attempted to solve it (Excoffier and Slatkin, 1995; Long *et al.*, 1995; Stephens *et al.*, 2001; Stephens and Donnelly, 2003; Niu *et al.*, 2002; Gusfield, 2002; Eskin *et al.*, 2003; Greenspan and Geiger, 2003; Kimmel and Shamir, 2004). These studies suggested a variety of models and proposed methods to simultaneously phase a set of genotypes under the model's assumptions.

Several researchers observed that haplotypes tend to be preserved along relatively long genomic stretches, with recombination occurring mostly in narrow regions (Gabriel *et al.*, 2002; Patil *et al.*, 2001). The stretch of SNP sites between two neighboring recombination regions is called a *block*. The number of different haplotypes within each block that are observed in a population is very small: typically, some 70–90% of the haplotypes within a block are identical (or almost identical) to very few (2–5) distinct *common haplotypes* (Patil *et al.*, 2001). Several studies suggested algorithms for the block identification and partitioning in a given dataset of haplotypes (e.g., Zhang *et al.* [2002a, 2003]; Koivisto *et al.* [2003]; Kimmel *et al.* [2004]). Recently, several studies proposed algorithms for simultaneous block partitioning and phasing on genotype data (Greenspan and Geiger, 2003; Kimmel and Shamir, 2005). However, as block patterns are sometimes unclear, it has been argued that less restrictive models of haplotype generation are needed (e.g., Wall and Pritchard [2003]; Bafna *et al.* [2003]).

In this study, we provide a new stochastic model for genotypes generation. The model, which generalizes our previous rigid blocks model (Kimmel and Shamir, 2005), aims to reflect two complementary (and somewhat contradictory) features observed on real haplotypes datasets: On one hand, contiguous SNPs tend to form "blocks" in which the recombination rate is low and with few distinct haplotypes. On the other hand, that block structure is not always conserved: Recombination may occur outside block boundaries, and some haplotypes may not fit the general structure altogether. From a different viewpoint, it was observed that different positions have different probability of recombination, and the linkage disequilibrium (i.e., the correlation between allele occurrence at different sites) is higher in some block-like regions than in others (Wall and Pritchard, 2003). The model preserves a basic "blocky" structure of haplotypes, but also allows for "exchange" of haplotypes at every point, and not only at block boundaries. Thus, it is possible that a specific contiguous stretch of SNPs identified as part of a haplotype can start or end inside another block of SNPs. Additionally, in any interval, some of the haplotypes may not be part of the blocky structure. We show how to infer the parameters of the model by deriving an EM method to achieve a maximum likelihood phasing solution.

We implemented our algorithm in a program called HINT (haplotype inference tool), which includes also a simple procedure to predict disease association. We tested HINT on 58 human datasets from four sources: HapMap project (*www.hapmap.org*), ENCODE project (*www.hapmap.org*), Daly *et al.* (2001), and Gabriel *et al.* (2002). Our experiments show that our algorithm consistently outperforms a strict blocks model and is also significantly more accurate than using haplotypes or using the raw genotypes.

We would like to point out that haplotype phasing is not strictly necessary for association testing. One can test for association of unphased genotypes, and the utility of using haplotypes is under debate (Morris and Kaplan, 2002; Martin *et al.*, 2000; Zhao *et al.*, 2003; Zaykin *et al.*, 2002; Zhang *et al.*, 2004). As we shall show, in our tests more accurate results are achieved by using our model than by using the raw genotypes.

The paper is organized as follows: In Section 2, we present the stochastic model. In Section 3, we show how to infer the parameters of the model. In Section 4, we present the experimental results. Section 5 discusses our methodology and future directions.

2. THE STOCHASTIC MODEL

The input to our problem is presented by an $n \times m$ *genotype matrix* M in which the rows correspond to samples (individuals genotyped) and the columns correspond to SNP sites. Hence, the i-th row $M[i, *]$, also denoted by g_i, describes the i-th genotype (the vector of readings for all the SNP sites in the ith individual). We assume that all sites are bi-allelic and that the two alleles are called arbitrarily 0 and 1. The genotype readings are denoted by $M[i, j] \in \{0, 1, 2, ?\}$ where 0 and 1 stand for the two homozygous types {0,0} and {1,1}, respectively, 2 stands for a heterozygous type, and "?" stands for a missing entry. A *phasing* of genotype g_i is a pair of binary n-vectors h_i^1, h_i^2, such that $(h_i^1(k), h_i^2(k))$ equals $(0, 0)$ if $g_i(k) = 0$ and (1,1) if $g_i(k) = 1$. If $g_i(k) = 2$, then it equals $(0, 1)$ or $(1, 0)$, and if $g_i(k) =?$ then all four

0/1 combinations are possible. Vectors (h_i^1, h_i^2) are called *haplotypes* corresponding to the genotype g_i. We use $g_{i,j}$ to denote the jth component (0, 1, 2, or ?) of the vector g_i. Given a genotype matrix, the *phasing problem* is to determine the most likely n pairs of haplotype vectors that constitute phasing of the corresponding genotype vectors. "Most likely" must be defined with respect to an assumed data model.

We now describe our probabilistic model for how the genotypes are generated. The model relaxes the rigid block structure assumption and allows for recombination of haplotypes at every point, and not only at block boundaries. Thus, it is possible that a specific contiguous stretch of SNPs starts or ends inside another block of SNPs. Additionally, in any interval, some of the haplotypes may not be part of the blocky structure. In a nutshell, the model is a hidden Markov model with few possible states at each site, each with its own emission probability, allowing transition from any state to any state in the next site. For each site there are at most k alternative states. Each genotype is created as the confluence of two independent Markov paths, each creating a single haplotype. Figure 1 gives an example of the model when applied on chromosome 5 dataset (Daly *et al.*, 2001).

We now define the model formally. The model has a position for each SNP site, and we use the terms site and position interchangeably. At each position there are k states. We denote by $S_{i,j}$ the ith state in the jth position. Each state generates ("emits") a SNP value in its corresponding position. We denote by $\theta_{i,j}$ the probability to generate "1" in the j-th site of the i-th state. (Consequently, $1-\theta_{i,j}$ is the probability to generate "0" in that site.) Let $\alpha^{(q)}$ denote the transition probability matrix from states in position q to states in position $q+1$. Transition probabilities between nonconsecutive positions are zero. The components of $\alpha^{(q)}$ are denoted by $(\alpha^{(q)})_{i,j} = \alpha_{i,j}^{(q)}$. Thus, $\alpha_{i,j}^{(q)} = \Pr[S_{j,q+1}|S_{i,q}]$. The starting state is denoted by S_0, and its corresponding transition probabilities to $\{S_{i,1}\}_{1\le i\le k}$ are denoted by $\alpha_{1,i}^{(0)}$.

Each genotype in the matrix M is generated as follows: For each haplotype independently, start from state S_0 and choose a number i between 1 and k according to the probability distribution $\{\alpha_{1,1}^{(0)}, \ldots, \alpha_{1,k}^{(0)}\}$. Pass to state $S_{i,1}$ and choose the first SNP value to be 1 with probability $\theta_{i,1}$. When at state $S_{i,q}$, choose a number j between 1 and k according to the probability distribution $\{\alpha_{i,1}^{(q)}, \ldots, \alpha_{i,k}^{(q)}\}$. Pass to state $S_{j,q+1}$ and choose the $(q+1)$th SNP value to be 1 with probability $\theta_{j,q+1}$. Continue the path until reaching one of the states $\{S_{i,m}\}_{1\le i\le k}$ (all m SNP values have been generated). Repeat the steps above for the second haplotype and form the confluence of the two haplotypes . The result is the genotype.

We now develop the likelihood function under the model. Define $f(g_{i,j};\theta_a,\theta_b)$ to be the probability of the observed value of the jth SNP in the ith genotype, given that the probabilities to observe 1 in the two paths that created the genotype are θ_a, θ_b, respectively. Then

$$f(g_{i,j};\theta_a,\theta_b) = \begin{cases} (1-\theta_a)(1-\theta_b) & g_{i,j}=0 \\ \theta_a\theta_b & g_{i,j}=1. \\ \theta_a(1-\theta_b)+\theta_b(1-\theta_a) & g_{i,j}=2 \end{cases}$$

Let $I_i^{(j)}$ and $J_i^{(j)}$ denote the state number of the first and second paths, respectively, of the ith genotype in the jth position. Note that $I_i^{(0)} = 1$ and $J_i^{(0)} = 1$ for $1 \le i \le n$. We use θ and α to denote the sets $\{\theta_{i,j} : 1 \le i \le n, 1 \le j \le m\}$ and $\{\alpha^{(j)} : 0 \le j \le m-1\}$, respectively. The full set of parameters is denoted by $\vartheta := \theta \cup \alpha$. The complete likelihood function can be written as follows:

$$L(M) = \prod_{i=1}^{n}\prod_{j=0}^{m-1} \alpha^{(j)}_{I_i^{(j)},I_i^{(j+1)}} \alpha^{(j)}_{J_i^{(j)},J_i^{(j+1)}} f(g_{i,j+1};\theta_{I_i^{(j+1)},j+1},\theta_{J_i^{(j+1)},j+1}).$$

The complete log likelihood function is

$$l(M) = \sum_{i=1}^{n}\sum_{j=0}^{m-1}\left[\log\alpha^{(j)}_{I_i^{(j)},I_i^{(j+1)}} + \log\alpha^{(j)}_{J_i^{(j)},J_i^{(j+1)}} + \log f(g_{i,j+1};\theta_{I_i^{(j+1)},j+1},\theta_{J_i^{(j+1)},j+1})\right]. \quad (1)$$

Dealing with missing entries is done as follows: given a missing entry, we sum over all possible values of the entry (0, 1, and 2), since these events are mutually exclusive. Inspection of Equation (1) reveals that this is equivalent to using a probability of 1 for the missing entry. In this way, missing entries are treated as part of the model and need not be handled separately from the optimization procedure.

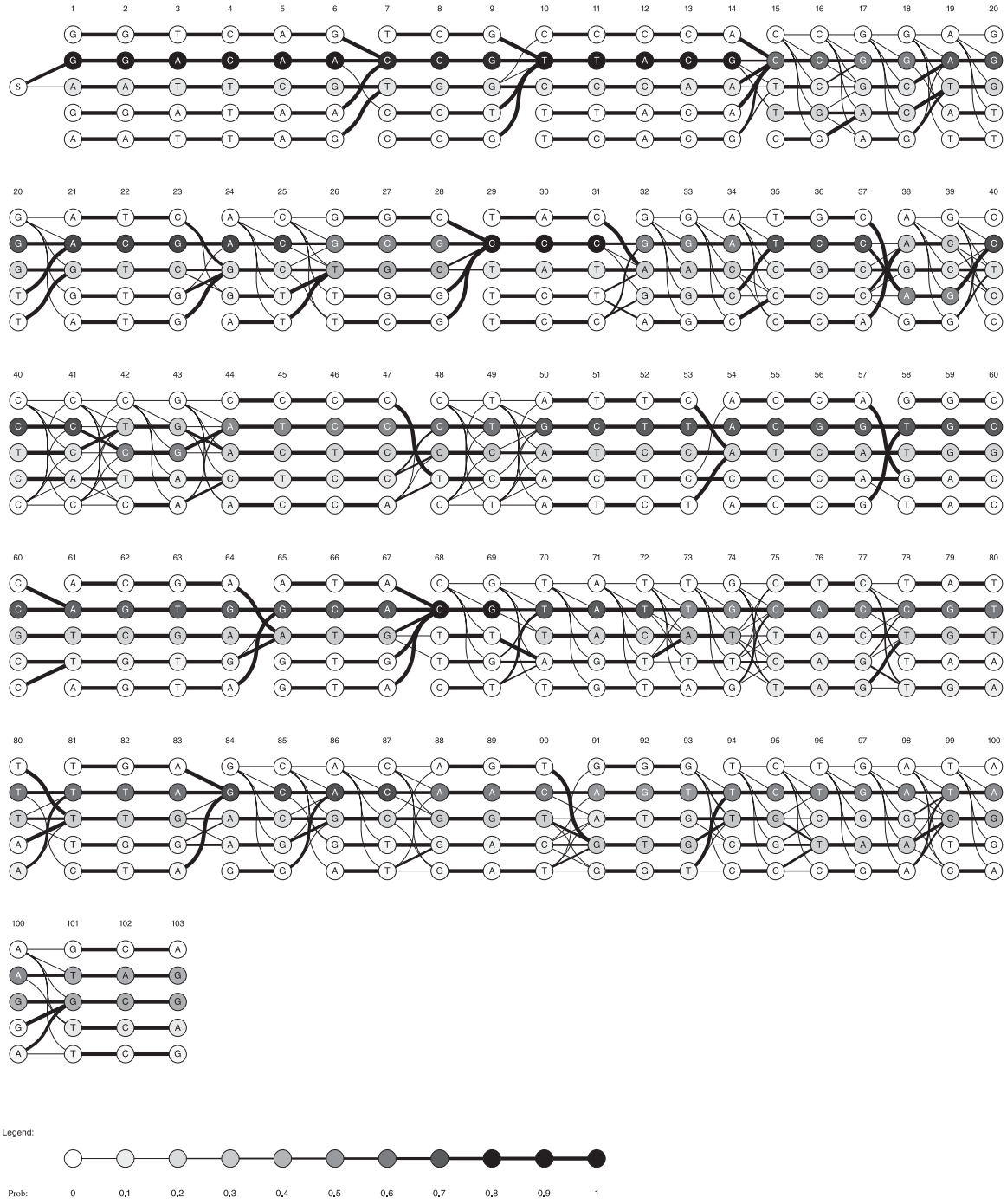


FIG. 1. A graphical illustration of the probabilistic model, obtained on the data of Daly *et al.* (2001) data. Each node is labeled by the name of the nucleotide that is more likely to be emitted at the corresponding state (i.e., the nucleotide in state i of position j is that corresponding to 1 if and only if $\theta_{i,j} > 0.5$). Transitions between states in consecutive positions are represented by edges with edge thickness proportional to probability of transition. For readability, only edges with probability above 0.05 are shown. Node probabilities are calculated from the Markov chain model and color coded according to the legend. The top line contains the SNP numbers. The picture was generated using the software Graphviz (*www.research.att.com/sw/tools/graphviz*). Observe that certain stretches of sites manifest a block-like structure, with little or no transition between lines of states (e.g., 1–6, 7–9, 10–14, 29–31, 61–64). Other stretches (e.g., 15–20, 38–44) contain many crossings and show little block structure.

3. AN ALGORITHM FOR INFERRING THE MODEL PARAMETERS

Using the maximum likelihood approach, the goal is to find a set of parameters $\widehat{\vartheta}$ that maximizes (1). Since $\{I_i^{(j)}, J_i^{(j)} : \forall i, j\}$ are unknown, we use EM approach (e.g., McLachlan and Krishnan [1997]). We define the random variable $A_{a,b,i}^{(s)}$ to be the number of transitions that are performed from state $S_{a,s}$ to the state $S_{b,s+1}$ in the process generating the ith genotype. Hence, $A_{a,b,i}^{(s)}$ can be 0, 1, or 2. We also define the random indicator variable $I_{\{A_{a,b,i}^{(j)}=2\}}$ to be 1 if $A_{a,b,i}^{(j)} = 2$, and 0 otherwise. Now, Equation (1) can be rewritten as:

$$l(M) = \sum_{i=1}^{n}\sum_{j=0}^{m-1}\left[\sum_{\substack{1\le a\le k\\1\le b\le k}} A_{a,b,i}^{(j)}\log\alpha_{a,b}^{(j)} + \frac{1}{2}\sum_{\substack{1\le a\le k\\1\le b\le k}}\sum_{\substack{1\le c\le k\\1\le d\le k\\(c,d)\neq(a,b)}} A_{a,b,i}^{(j)}A_{c,d,i}^{(j)}\log f(g_{i,j+1};\theta_{b,j+1},\theta_{d,j+1})\right.$$
$$\left. + \sum_{\substack{1\le a\le k\\1\le b\le k}} I_{\{A_{a,b,i}^{(j)}=2\}}\log f(g_{i,j+1};\theta_{b,j+1},\theta_{b,j+1})\right].$$

Taking the expectation with respect to the probability measure under the parameter set ϑ_0, we get

$$Q_{M,\vartheta_0}(\vartheta) = \sum_{i=1}^{n}\sum_{j=0}^{m-1}\left[\sum_{\substack{1\le a\le k\\1\le b\le k}} \mathbb{E}[A_{a,b,i}^{(j)}]\log\alpha_{a,b}^{(j)} + \frac{1}{2}\sum_{\substack{1\le a\le k\\1\le b\le k}}\sum_{\substack{1\le c\le k\\1\le d\le k\\(c,d)\neq(a,b)}}\right.$$
$$\times\ \mathbb{E}[A_{a,b,i}^{(j)}A_{c,d,i}^{(j)}]\log f(g_{i,j+1};\theta_{b,j+1},\theta_{d,j+1})$$
$$\left. + \sum_{\substack{1\le a\le k\\1\le b\le k}} \mathbb{E}[I_{\{A_{a,b,i}^{(j)}=2\}}]\log f(g_{i,j+1};\theta_{b,j+1},\theta_{b,j+1})\right].$$

For compactness of writing, we use matrices to describe the above equations. Define the matrices $W_i^{(s)}$, $Y_i^{(s)}$, $\alpha_i^{(s)}$, $G_i^{(s)}$ as follows.

$$(W_i^{(s)})_{a,b} := \mathbb{E}[A_{a,b,i}^{(s)}]$$
$$(Y_i^{(s)})_{b,d} := \begin{cases} \frac{1}{2}\sum_{\substack{1\le a\le k\\1\le c\le k}} \mathbb{E}[A_{a,b,i}^{(s)}A_{c,d,i}^{(s)}] & b\neq d \\ \frac{1}{2}\sum_{\substack{1\le a\le k\\1\le c\le k\\a\neq c}} \mathbb{E}[A_{a,b,i}^{(s)}A_{c,d,i}^{(s)}] + \sum_{\substack{1\le a\le k\\1\le c\le k\\a=c}} \mathbb{E}[I_{A_{a,b,i}^{(s)}=2}] & b=d \end{cases} \quad (2)$$
$$(\alpha^{(s)})_{a,b} := \alpha_{a,b}^{(s)}$$
$$(G_i^{(s)})_{a,b} := \begin{cases} (1-\theta_{a,s})(1-\theta_{b,s}) & g_{i,s}=0 \\ \theta_{a,s}\theta_{b,s} & g_{i,s}=1 \\ \theta_{a,s}(1-\theta_{b,s})+\theta_{b,s}(1-\theta_{a,s}) & g_{i,s}=2 \end{cases}$$

If U, V are $n \times m$ real matrices, let $U \bullet V$ denote the inner product of the two matrices: $U \bullet V = \text{trace}(U^T V)$ (equivalently, it can also be expressed as $\sum_{\substack{1 \le i \le n \\ 1 \le j \le m}} U_{i,j} V_{i,j}$). We use $U_{a,\cdot}$ to denote the ath row of U, and $U_{\cdot,b}$ to denote the bth column of U. Vector $\boldsymbol{e_n}$ is defined to be the vector $(1, 1, \ldots, 1)^T$ of size n. Let $\mathbb{S}^n$ denote the n-dimensional open simplex, i.e., $\mathbb{S}^n = \{x \in \mathbb{R}^n \mid \forall i : \ 0 < x_i < 1\}$.

Using the above definitions, $Q_{M,\vartheta_0}(\vartheta)$ can be rewritten, and we get the following optimization problem:

$$\begin{aligned} &\text{max: } Q_{M,\vartheta_0}(\vartheta) = \sum_{i=1}^{n} \sum_{j=0}^{m-1} \left[W_i^{(j)} \bullet \log(\alpha^{(j)}) + Y_i^{(j)} \bullet \log(G_i^{(j+1)}) \right], \\ &\text{subject to: } \vartheta \in \mathbb{S}^{|\vartheta|}, \\ &\forall a, s : \ \sum_{b=1}^{k} \alpha_{a,b}^{(s)} = 1. \end{aligned} \tag{3}$$

We use the interior domain of the simplex, excluding the surface, since we want the objective function to be twice continuously differentiable. For practical purposes, this assumption is not significant. We use W, Y, α, G to denote the sets $\{W_i^{(j)}\}, \{Y_i^{(j)}\}, \{\alpha^{(j)}\}, \{G_i^{(j+1)}\}$, respectively. Note that W, Y are constants and α, G are determined directly by the parameters set ϑ. The constants W, Y are obtained using ϑ_0, as will be presented below. The optimization problem (3) is called "the genotypes optimization subproblem." Clearly, a solution $\widehat{\vartheta}$ to the genotypes optimization subproblem increases the complete likelihood score of (1).

We now describe the expectation and maximization steps of the algorithm. We show how to calculate W and Y given ϑ and how to calculate ϑ when W and Y are known. We also explain how we resolve the genotypes, once the model's parameters are found.

3.1. Expectation

Since each genotype is generated by two passes through the hidden Markov model, we can formally double the model for the representation below. In this doubled model, for each position there are $\binom{k}{2}$ states, which correspond to all possible pairs of states in the original model. We denote them by $\{D_{\{a,b\},j} : 1 \le a \le b \le k, 1 \le j \le m\}$, where $D_{\{a,b\},j}$ corresponds to the pair of states $S_{a,j}, S_{b,j}$. Let $\Pr[e|D_{\{a,b\},j}]$ denote the probability to observe $e \in \{0, 1, 2\}$, given the state $D_{\{a,b\},j}$. The probability to obtain 0, 1, and 2 in each of these states can be calculated as follows:

$$\begin{aligned} \Pr[0|D_{\{a,b\},j}] &= (1 - \theta_{a,j})(1 - \theta_{b,j}), \\ \Pr[1|D_{\{a,b\},j}] &= \theta_{a,j}\theta_{b,j}, \\ \Pr[2|D_{\{a,b\},j}] &= \theta_{a,j}(1 - \theta_{b,j}) + \theta_{b,j}(1 - \theta_{a,j}). \end{aligned}$$

The transition probabilities in the doubled model can also be calculated directly:

$$\Pr[D_{\{c,d\},j'}|D_{\{a,b\},j}] = \begin{cases} \alpha_{a,c}^{(j)}\alpha_{b,d}^{(j)} + \alpha_{a,d}^{(j)}\alpha_{b,c}^{(j)} & c \ne d, \ j' = j + 1 \\ \alpha_{a,c}^{(j)}\alpha_{b,c}^{(j)} & c = d, \ j' = j + 1 \\ 0 & \text{otherwise} \end{cases}.$$

Using the Baum–Welch algorithm (Baum, 1972), we find $\Pr[D_{\{a,b\},j}, D_{\{c,d\},j+1}|g_i]$, and we can then obtain

$$\begin{aligned} \Pr[A_{a,b,i}^{(j)} = 1|g_i] = &\sum_{\substack{a_1=a_2=a \\ b_1=b, b_2 \ne b}} \Pr[D_{(a_1,a_2),j}, D_{(b_1,b_2),j+1}|g_i] \\ &+ \sum_{\substack{a_1=a, a_2 \ne a \\ b_1=b, b_2 \ne b}} \left[\Pr[D_{(a_1,a_2),j}, D_{(b_1,b_2),j+1}|g_i] \frac{\alpha_{a_1,b_1}^{(j)}\alpha_{a_2,b_2}^{(j)}}{\alpha_{a_1,b_1}^{(j)}\alpha_{a_2,b_2}^{(j)} + \alpha_{a_1,b_2}^{(j)}\alpha_{a_2,b_1}^{(j)}} \right], \end{aligned}$$

$$\Pr[A^{(j)}_{a,b,i}=2|g_i]=\Pr[D_{(a,a),j},D_{(b,b),j+1}|g_i],$$

$$\Pr[A^{(j)}_{a_1,b_1,i}=1,A^{(j)}_{a_2,b_2,i}=1|g_i]=\begin{cases}\Pr[D_{(a_1,a_2),j},D_{(b_1,b_2),j+1}|g_i]\dfrac{\alpha^{(j)}_{a_1,b_1}\alpha^{(j)}_{a_2,b_2}}{\alpha^{(j)}_{a_1,b_1}\alpha^{(j)}_{a_2,b_2}+\alpha^{(j)}_{a_1,b_2}\alpha^{(j)}_{a_2,b_1}} & a_1\neq a_2\\ \Pr[D_{(a_1,a_2),j},D_{(b_1,b_2),j+1}|g_i] & a_1=a_2\end{cases}.$$

We are now ready to calculate the expectation:

$$\begin{aligned}\mathbb{E}[A^{(j)}_{a,b,i}|g_i]&=\Pr[A^{(j)}_{a,b,i}=1|g_i]+2\Pr[A^{(j)}_{a,b,i}=2|g_i],\\ \mathbb{E}[A^{(j)}_{a_1,b_1,i}A^{(j)}_{a_2,b_2,i}|g_i]&=\Pr[A^{(j)}_{a_1,b_1,i}=1,A^{(j)}_{a_2,b_2,i}=1|g_i],\ \text{if } (a_1,b_1)\neq(a_2,b_2),\\ \mathbb{E}[I_{\{A^{(j)}_{a,b,i}=2\}}]&=\Pr[A^{(j)}_{a,b,i}=2|g_i].\end{aligned}\tag{4}$$

Now, matrices $W^{(j)}_i$, $Y^{(j)}_i$ can be obtained by substituting (4) into (2).

3.2. Maximization

The following lemma implies that solving the genotypes optimization subproblem, even when W and Y are given, is not trivial, as the optimization function is neither convex nor concave.

Lemma 1. *The genotypes optimization subproblem is neither convex nor concave.*

Proof. It suffices to give an example where convexity and concavity do not hold. Let $k=2$ and $m=2$. The function $Q_{M,\vartheta_0}(\vartheta)$ is assumed to be twice continuously differentiable, and thus we use $H(\vartheta)$ to denote the Hessian matrix of $Q_{M,\vartheta_0}(\vartheta)$ at point ϑ. We note that $Q_{M,\vartheta_0}(\vartheta)$ is the sum of functions of $\alpha^{(0)}$, $\alpha^{(1)}$, $\theta_{\cdot,1}$, and $\theta_{\cdot,2}$, and hence it is enough to show that $Q_{M,\vartheta_0}(\vartheta)$ is nonconvex and nonconcave in $\theta_{\cdot,2}$. Denote by $H(\theta_{\cdot,2})$ the submatrix of $H(\vartheta)$, which is induced only by $\theta_{\cdot,2}$; i.e., $(H(\theta_{\cdot,2}))_{i,j}:=\frac{\partial^2 Q_{M,\vartheta_0}}{\partial\theta_{i,2}\partial\theta_{j,2}}$. We can write $Q_{M,\vartheta_0}(\vartheta)$ as follows:

$$\begin{aligned}Q_{M,\vartheta_0}(\vartheta)&=t_1(\alpha)+t_2(\theta_{\cdot,1})\\ &\quad+C_{11}\log(\theta^2_{1,2})+C_{12}\log(\theta_{1,2}\theta_{2,2})+C_{22}\log(\theta^2_{2,2})\\ &\quad+D_{11}\log(\theta_{1,2}(1-\theta_{1,2}))+D_{22}\log(\theta_{2,2}(1-\theta_{2,2}))\\ &\quad+E_{11}\log((1-\theta_{1,2})^2)+E_{12}\log((1-\theta_{1,2})(1-\theta_{2,2}))+E_{22}\log((1-\theta_{2,2})^2)\\ &\quad+R\log[\theta_{1,2}(1-\theta_{2,2})+\theta_{2,2}(1-\theta_{1,2})],\end{aligned}$$

where $t_1(\alpha)$ is a function of α, $t_2(\theta_{\cdot,1})$ is a function of $\theta_{\cdot,1}$, and $C_{\cdot\cdot}$, $D_{\cdot\cdot}$, $E_{\cdot\cdot}$, R are positive constants that depend on $Y^{(1)}_i$ and on the genotypes data. In this way, the Hessian matrix is

$$H(\theta_{\cdot,2})=\begin{pmatrix}-\left[P_{11}\frac{1}{\theta^2_{1,2}}+S_{11}\frac{1}{(1-\theta_{1,2})^2}+R\frac{(1-2\theta_{2,2})^2}{(\theta_{1,2}+\theta_{2,2}-2\theta_{1,2}\theta_{2,2})^2}\right] & -R\frac{1}{(\theta_{1,2}+\theta_{2,2}-2\theta_{1,2}\theta_{2,2})^2}\\ -R\frac{1}{(\theta_{1,2}+\theta_{2,2}-2\theta_{1,2}\theta_{2,2})^2} & -\left[P_{22}\frac{1}{\theta^2_{2,2}}+S_{22}\frac{1}{(1-\theta_{2,2})^2}+R\frac{(1-2\theta_{1,2})^2}{(\theta_{1,2}+\theta_{2,2}-2\theta_{1,2}\theta_{2,2})^2}\right]\end{pmatrix},$$

where

$$P_{11} := 2C_{11} + C_{12} + D_{12} + D_{11},$$

$$S_{11} := 2E_{11} + E_{12} + D_{21} + D_{11},$$

$$P_{22} := 2C_{22} + C_{12} + D_{21} + D_{22},$$

$$S_{22} := 2E_{22} + E_{12} + D_{12} + D_{22}.$$

Choosing $\theta_{\cdot,2} = \widehat{\theta_{\cdot,2}} = (0.5 \;\; 0.5)^T$, we get

$$H(\widehat{\theta_{\cdot,2}}) = \begin{pmatrix} -[4P_{11} + 4S_{11}] & -4R \\ -4R & -[4P_{22} + 4S_{22}] \end{pmatrix}. \tag{5}$$

We can set R to be larger enough than $P_{11}, S_{11}, P_{22}, S_{22}$, such that $\det[H(\widehat{\theta_{\cdot,2}})] < 0$. This is possible since the constants $P_{11}, S_{11}, P_{22}, S_{22}$ depend only on the number of homozygous SNPs (0 or 1) and R depends only on the number the heterozygous SNPs (2). According to the Frobenius Theorem, since all the components of (5) are negative, it has at least one negative eigenvalue, and since $\det[H(\widehat{\theta_{\cdot,2}})] < 0$, the other eigenvalue must be positive. If the signs of the two eigenvalues of $H(\widehat{\theta_{\cdot,2}})$ are different, then $H(\widehat{\theta_{\cdot,2}})$ is neither positive semi-definite nor negative semi-definite, so $\widehat{\theta_{\cdot,2}}$ is a feasible point, where the function $Q_{M,\vartheta_0}(\vartheta)$ is nonconvex and nonconcave. ■

Notice that if the haplotypes are given in a resolved form, and the problem is only to find the parameters, then we have a simple hidden Markov chain model. In this case, the optimization subproblem is convex and has an analytical solution using the Baum–Welch algorithm (Baum, 1972). In general, the whole EM process converges to a local optimum that is not necessarily the global optimum, but as we shall show, overall performance is usually good and robust on the real data that we used.

A solution to the genotype optimization subproblem. Although, as was shown in Lemma 1, the genotype optimization subproblem is nonconvex and nonconcave, we cope with it in the following way: As can be observed in (3) the function $Q_{M,\vartheta_0}(\vartheta)$ is a linear combination of $\log(\alpha^{(j)})$ and $\log(G_i^{(j+1)})$, for all $0 \le j \le m-1$. Thus, we can perform the optimization process for each j separately. For a given j, we want to solve

$$\begin{aligned} &\text{max: } Q_{M,\vartheta_0}(\vartheta_j) = \sum_{i=1}^{n} \left[W_i^{(j)} \bullet \log(\alpha^{(j)}) + Y_i^{(j)} \bullet \log(G_i^{(j+1)}) \right], \\ &\vartheta_j \in \mathbb{S}^{|\vartheta_j|}, \\ &\forall a : \; \sum_{b=1}^{k} \alpha_{a,b}^{(j)} = 1. \end{aligned} \tag{6}$$

Under the domain $\vartheta_j \in \mathbb{S}^{|\vartheta_j|}$, the Lagrangian is

$$L_{M,\vartheta_0}(\alpha^{(j)}, \theta_j, \lambda) = \sum_{i=1}^{n} \left[W_i^{(j)} \bullet \log(\alpha^{(j)}) + Y_i^{(j)} \bullet \log(G_i^{(j+1)}) \right] + \sum_{a=1}^{k} \lambda_a \left(\sum_{b=1}^{k} \alpha_{a,b}^{(j)} - 1 \right), \tag{7}$$

where $\lambda = (\lambda_1, \ldots, \lambda_k)^T \in \mathbb{R}^k$. We find an analytical solution for $\alpha^{(j)}$ by solving $\frac{\partial L_{M,\vartheta_0}(\alpha^{(j)}, \theta_j, \lambda)}{\partial \alpha^{(j)}} = 0$. We obtain

$$\alpha_{a,b}^{(s)} = \frac{\sum_{i=1}^{n} \mathbb{E}[A_{a,b,i}^{(s)}]}{\sum_{i=1}^{n} \sum_{b=1}^{k} \mathbb{E}[A_{a,b,i}^{(s)}]}.$$

Finding θ_j is done numerically by solving

$$\max_{\theta_j \in \mathbb{S}^{|\theta_j|}} Y_i^{(j-1)} \bullet \log(G_i^{(j)}), \tag{8}$$

which is neither convex nor concave, by Lemma 1. However, we note that there are only k variables in this optimization problem, and k is assumed to be small, so solving it numerically is possible in practice. Moreover, when working with biological genotypes, it is reasonable to restrict the possible values of θ_j to be close to 0 or close to 1. Hence, we solve the same optimization problem described in (8), but we use the domain $\mathbb{S}_\epsilon^n = \{x \in \mathbb{R}^n | \ \forall i : \ 0 < x_i < \epsilon \text{ or } 1 - \epsilon < x_i < 1\}$ instead of $\mathbb{S}^n$, where ϵ is some small constant.

3.3. Resolving the genotypes

Once the parameters set $\widehat{\vartheta} = \{\widehat{\alpha}, \widehat{\theta}\}$ is found, we use Viterbi's algorithm (Viterbi, 1967) to find the optimal path for each genotype in the double Markov chain model. The algorithm provides a pair $D_{\{a_1,a_2\},j}$ for each position j, and one still has to determine the two paths corresponding to the two haplotypes, in terms of the original Markov chain. If for a genotype g_i, in positions j, $j+1$, the two state pairs found by the algorithm are $D_{(a_1,a_2),j}$ and $D_{(b_1,b_2),j+1}$, respectively, then we have two possibilities for the two paths in the original Markov chain:

1. $S_{a_1,j} \to S_{b_1,j+1}$ and $S_{a_2,j} \to S_{b_2,j+1}$,
2. $S_{a_1,j} \to S_{b_2,j+1}$ and $S_{a_2,j} \to S_{b_1,j+1}$.

For each, the probability can be calculated as follows:

$$\Pr[S_{a_1,j} \to S_{b_1,j+1}, S_{a_2,j} \to S_{b_2,j+1} | D_{(a_1,a_2),j}, D_{(b_1,b_2),j+1}] = \frac{\alpha^{(j)}_{a_1,b_1}\alpha^{(j)}_{a_2,b_2}}{\alpha^{(j)}_{a_1,b_1}\alpha^{(j)}_{a_2,b_2} + \alpha^{(j)}_{a_1,b_2}\alpha^{(j)}_{a_2,b_1}},$$

$$\Pr[S_{a_1,j} \to S_{b_2,j+1}, S_{a_2,j} \to S_{b_1,j+1} | D_{(a_1,a_2),j}, D_{(b_1,b_2),j+1}] = \frac{\alpha^{(j)}_{a_1,b_2}\alpha^{(j)}_{a_2,b_1}}{\alpha^{(j)}_{a_1,b_1}\alpha^{(j)}_{a_2,b_2} + \alpha^{(j)}_{a_1,b_2}\alpha^{(j)}_{a_2,b_1}}.$$

We choose the possibility with larger probability, and thus the resulting two paths maximize the likelihood as required.

Once we have a separate path for each haplotype, we resolve each SNP in each haplotype, according to the larger probability in each site. If for haplotype $h_{i,p}$ (where $p \in \{1, 2\}$) in position j the corresponding state is $S_{a,j}$, then $h_{i,p,j}$ is determined to be 0 if $\theta_{a,j} < 0.5$, and 1 otherwise.

4. RESULTS ON BIOLOGICAL DATASETS

We implemented our algorithm in a software package call HINT (haplotype inference tool). HINT was implemented in C++. Running time on a 2 GHz Pentium 4 for 100 genotypes with 100 SNPs is approximately two minutes.

4.1. Description of the datasets

We tested HINT extensively using four datasets encompassing 58 different genomic regions.

- A dataset due to Daly *et al.* (2001). In this study, genotypes for 103 SNPs from a 500 KB region of chromosome 5q31 were collected from 129 mother, father, and child trios from a European derived population in an attempt to identify a genetic risk for Crohn's disease. We used only the child population in this dataset.

- Population D from the study of Gabriel *et al.* (2002). The data consist of 51 sets of genotypes from various genomic regions, where the number of SNPs per region ranges from 13 to 114.
- Regions ENm013, ENr112, and ENr113 of the ENCODE project (*www.hapmap.org*). These are 500 KB regions of chromosomes $7q21.13$, $2p16.3$, and $4q26$, respectively, which were collected from 30 trios. The numbers of SNPs genotyped in each region are 361, 412, and 515 respectively (thus, the density of the sample is 3–5 times greater than the density of that of Daly *et al.* [2001]). For convenience, we divided each of these regions into four or five datasets that contain approximately 100 SNPs each.
- Genotypes from the HapMap project (*www.hapmap.org*). We used three sets of SNPs spanning the three genes PP2R4, STEAP, and TRPM8. For each of these genes, we took the HapMap SNPs that are spanned by the gene plus 10 KB upstream and downstream. The resulting sets contain 39, 23, and 102 SNPs.

The last three sets contained 30 mother, father, and child trios from Yoruba's population. In each case we used only the 60 genotypes of the parents.

4.2. Initialization and predefined constants

When using the model to predict diseases, we chose to use $k = 5$ on *all datasets* since our tests show that this value obtains the most accurate results on a large number of different datasets (Fig. 2). Here, we used *mean prediction rate* as a measure for accuracy. This measure reflects the accuracy in predicting a missing causative SNP and will be described in detail in Section 4.4. Notably, the changes are minor when different values of k are used.

We used the GERBIL software (Kimmel and Shamir, 2004) for finding initial parameter values for HINT. GERBIL phases the data and creates a block partition. In each block, the number of common haploypte in GERBIL is determined using a minimum description length criterion (see Section 4.3). We use only the k most common haplotypes in each block, where k is a predefined parameter of HINT. A probability matrix is computed for the transitions between common haploypte in consecutive blocks. Accordingly, if the neighboring SNPs j and $j+1$ are in the same block, then the initialized values of $\alpha^{(j)}$ are set to be $\alpha^{(j)}_{q,q} = 1$ and $\alpha^{(j)}_{q_1,q_2} = 0$, for $q_1 \neq q_2$. If the neighboring SNPs j and $j+1$ are in different blocks, then

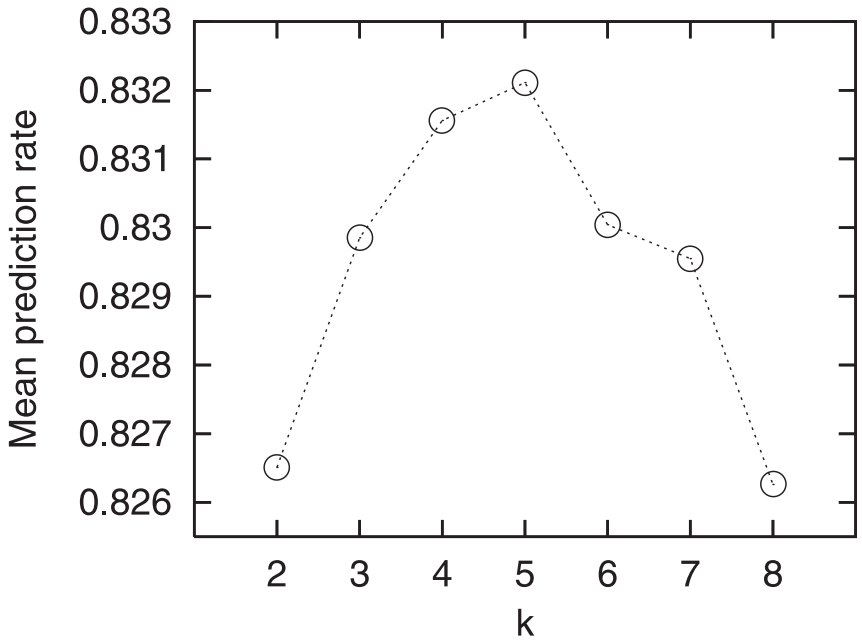


FIG. 2. Mean prediction accuracy versus different values of k parameter, on 51 different datasets of Gabriel *et al.* (2002).

$\alpha^{(j)}$ is set according GERBIL's transition probability matrix between common haplotypes. Each parameter θ is initialized to its corresponding value in GERBIL.

4.3. MDL comparison

In order to assess the possible advantage of our new block-free model over a rigid block model, we chose to compare it to GERBIL. This choice was made due to two reasons: first, GERBIL was shown to be relatively accurate (Kimmel and Shamir, 2004), and second, the main difference between the two models is the strict block structure of GERBIL. Since the HINT model is more complex, direct comparison of likelihood is meaningless. Instead, we calculated the minimum description length (MDL) score in both solutions, as suggested by Greenspan and Geiger (2003) and by Koivisto *et al.* (2003) using the formula: Length(Model, Data) = Length(Model) + Length(Data | Model), where Length$(\cdot)$ measures the description length in bits. We used accuracy of $(\log n)/2$ to describe the numbers, based on (Rissanen, 1987). Hence, Length(Model) is the length of the parameters in bits, and Length(Data | Model) = $-\log(\Pr[\text{Data} \mid \text{Model}]) = -l(M)$, which is derived in Equation (1).

Our experiments show that using $k = 2$ achieves the minimal MDL score, so we used this value for the MDL evaluation. The results on all datasets are presented in Fig. 3. For GERBIL's model, the parameter k and the blocks partition are chosen to minimize the MDL score. HINT's MDL score was significantly lower than GERBIL's with paired t-test p-value of 0.0074. The mean of the difference between HINT's MDL score and GERBIL's score was 695.58. We would like to comment that using $k = 2$ in HINT is not the optimal parameter for disease prediction (see Section 4.2), although it is the optimized parameter for the MDL score.

4.4. Disease Prediction

We wanted to assess the model's utility in another test, which is paramount for medical applications: the veracity of predicting genotypes at unobserved SNPs. This is a first step towards finding disease alleles. For that purpose, we used real genotypes to simulate case-control data, as follows: Suppose a single SNP

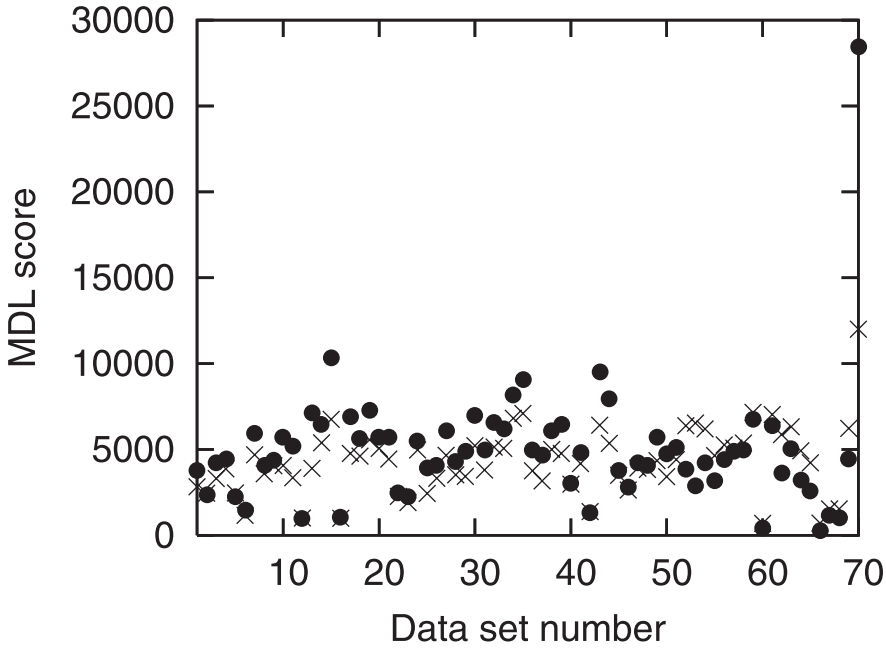


FIG. 3. A comparison of MDL scores on all biological datasets: blue X—HINT model, red circles—strict block model (GERBIL). Dataset number 70 of Daly *et al.* (2001) contains more genotypes than the datasets 1–69 (129 versus 60), which explains its higher MDL score.

causes the disease with 100% penetrance. Assume that the SNP value (0, 1, or 2) is the phenotype data. Our test is done in a leave-one-out manner: We select one test genotype and use the rest as training data. In all genotypes, the causative SNP is removed. In the training data, its value is used as the phenotype of each genotype. The goal is to determine the phenotype of the test genotype, based on the training information and on the other SNPs in the test genotype. The process is applied repeatedly by selecting all possible combinations of test genotype and causative SNPs.

Formally, for a genotype matrix M, let s be the test genotype number, and let t be the missing SNP number. For specific s, t, we build the $n-1 \times m-1$ induced matrix $M^{s,t}$, which equals M without the tth line and the sth column:

$$M^{s,t}_{i,j} = \begin{cases} M_{i,j} & i < s \text{ and } j < t \\ M_{i+1,j} & i \geq s \text{ and } j < t \\ M_{i,j+1} & i < s \text{ and } j \geq t \\ M_{i+1,j+1} & i \geq s \text{ and } j \geq t \end{cases}.$$

Let $d^{s,t} = (M_{i,t} : i \neq s)^T$ be the missing SNP vector in the training set genotypes, and let $g^{s,t} = (M_{s,j} : j \neq t)$ be the test genotype without that SNP. The goal is to predict the causative SNP based on the training data. Let Pred be an algorithm that predicts the phenotype in a test genotype given training data (genotypes and phenotypes). Let $\text{Pred}(M^{s,t}, d^{s,t}, g^{s,t})$ be the value predicted by Pred for the tth SNP in the sth genotype. Note that the input for the learning algorithm does not include the position of the missing SNP (t). The *prediction accuracy* μ is the probability to be correct in predicting a specific SNP in some genotype and is evaluated by (see Halperin *et al.* [2005]):

$$\mu = \frac{1}{nm} |\{\text{Pred}(M^{i,j}, d^{i,j}, g^{i,j}) = M_{i,j} \mid 1 \leq i \leq n,\ 1 \leq j \leq m\}|.$$

We checked two different scenarios:

1. The phenotype value is given as the original form of the missing SNP (0, 1, or 2). The goal here is predict that value.
2. A dominant disease model is assumed, where values of 1 or 2 in the causative SNP represent cases and 0 represents controls. The phenotype information is represented only in the form of case/control (both as an input for the prediction algorithm and as the predicted value).

In both models, we assume 100% penetrance, which is unrealistic. The reason is that we wish to measure the disease prediction accuracy of several models. Adding randomness by using lower penetrance may reflect the biological reality better, but would have obfuscated this measured value.

We now present our prediction algorithm. The algorithm, a variation of the prediction algorithm of Halperin *et al.* (2005), is based on the observation, made by several biological studies, that the correlation between SNPs tends to decay as the physical distance increases (see, e.g., Gabriel *et al.* [2002]; Bafna *et al.* [2003]; Daly *et al.* [2001]; Kimmel and Shamir [2004]; Halperin *et al.* [2005]). We assume that given the genotypes values of two SNPs, the probabilities of the values at any intermediate SNPs do not change by knowing the values of additional more distant ones. This assumption, although not valid in all cases, was shown to lead to accurate results in predicting SNPs correctly (Halperin *et al.*, 2005). Here, we apply some variation of the algorithm described by Halperin *et al.* (2005) to predict the missing SNP in two steps: (1) finding two consecutive SNPs $j, j+1$ in the training set that predict the phenotype most accurately and (2) predicting the missing SNPs using SNPs $j, j+1$. The prediction algorithm is presented in Fig. 4. For simplicity, we describe scenario (1), where the SNP value itself (0, 1, or 2) is the phenotype.

We compared the prediction accuracy of HINT to that of three different models: resolved haplotypes, genotypes, and a strict blocks model. For each of the four methods, we first constructed an *axillary matrix A*

Input: $M^{s,t}$, $d^{s,t}$ and $g^{s,t}$.

Output: An integer $q \in \{0, 1, 2\}$ which is a predicted value of the t-th SNP in the s-th genotype.

For every pair of adjacent SNPs j and $j + 1$,

Apply a leave one out procedure on $M^{s,t}$, as follows:

For $i = 1$ to $n - 1$,

Let l_i be the test genotype, and $M_i^{s,t}$ be the matrix $M^{s,t}$ without genotype l_i,

Predict the i-th component of $d^{s,t}$ using majority vote based on the

training set $M_i^{s,t}$ and columns $j, j + 1$.

If the predicted value equals the real value then set Score(j)=Score(j)+1.

Use the columns argmax $\{\text{Score}(j)\}$ and argmax $\{\text{Score}(j)\}$ +1 of $M^{s,t}$ to predict the SNP in position t

in the s-th genotype using majority vote applied on $g^{s,t}$.

FIG. 4. The prediction algorithm.

using the specific model. Then, using $M = A$, the same prediction algorithm described above (Fig. 4) was employed in the four cases to predict the disease status or the missing SNP of the test genotype. The matrix A reflects the data and the additional information obtained by the model. It is calculated as follows:

1. For the HINT model, A is a $2n \times m$ matrix that contains the states' indices of the resolved haplotypes of the data. Hence, if the SNP $g_i(j)$ was found using HINT to be generated by states $(S_{q_1,j}, S_{q_2,j})$, then $(A)_{2i-1,j} = q_1$ and $(A)_{2i,j} = q_2$. (Hence, $A_{i,j} \in \{1, 2, \ldots, k\}$.)
2. For the haplotypes model, A is a $2n \times m$ matrix that equals the resolved haplotype matrix of the genotype matrix M. (Hence, $A_{i,j} \in \{0, 1\}$.) Here the true haplotypes information available from pedigree data was used.
3. For the simple SNPs model, A is a $n \times m$ matrix that equals the genotype matrix M. (Hence, $A_{i,j} \in \{0, 1, 2\}$.)
4. For a strict blocks model, we use GERBIL (Kimmel and Shamir, 2004). Here, A is a $2n \times b$ matrix, where b is the number of blocks obtained by the algorithm. Indices $(A)_{2i-1,j}$ and $(A)_{2i,j}$ are defined to be the common haplotype indices of the jth block of the ith genotype. (Hence, $A_{i,j} \in \{1, 2, \ldots, k_b\}$, where k_b is the maximal number of common haplotypes allowed in a block.)

Figures 5 and 6 present the results of the SNP prediction model and the heterozygous disease model. The *prediction rate* is the fraction of correct predictions made by the model. Means and standard deviations are summarized in Tables 1 and 2. HINT shows a consistent advantage over the other models on most data sets. Notably, the simple haplotypes model was the second most accurate. The differences between the different models are statistically significant. For example, in the SNP prediction model, the difference between the HINT model and the haplotypes model is 7.4 STDs, which corresponds to a t-test p-value of $6.15 \cdot 10^{-12}$. A less significant difference is seen in the heterozygous disease model, possibly due to loss of information (p-value $= 0.039$). Interestingly, in both SNP and disease prediction scenarios, the blocks structure model was the least accurate.

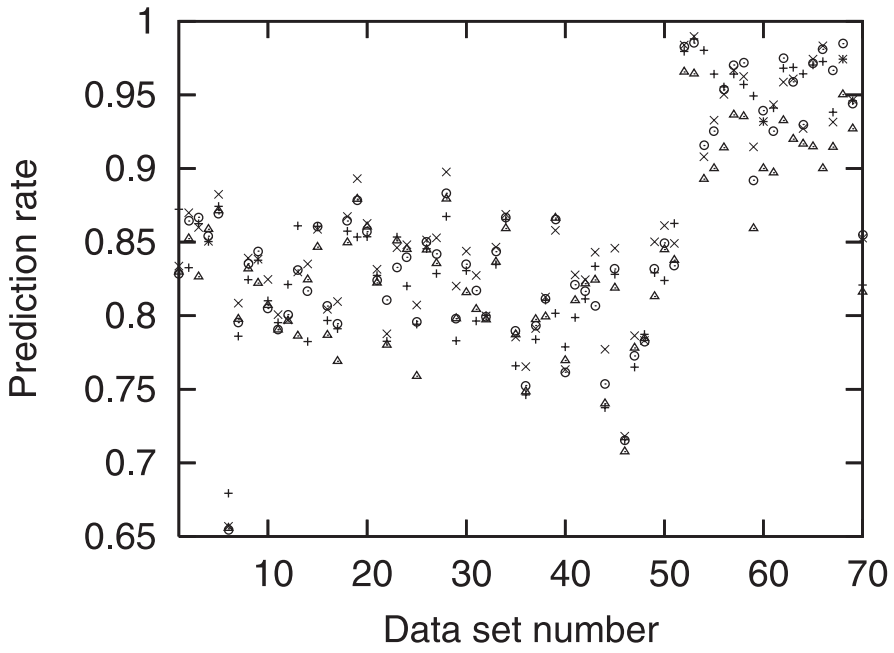


FIG. 5. SNP value prediction rates for all the biological datasets: black X—HINT, red circles—haplotypes, blue +—genotypes, green triangles—strict blocks (GERBIL). Datasets 52–69 from the ENCONDE project have high density SNPs, which explains the better prediction rate by all the algorithms.

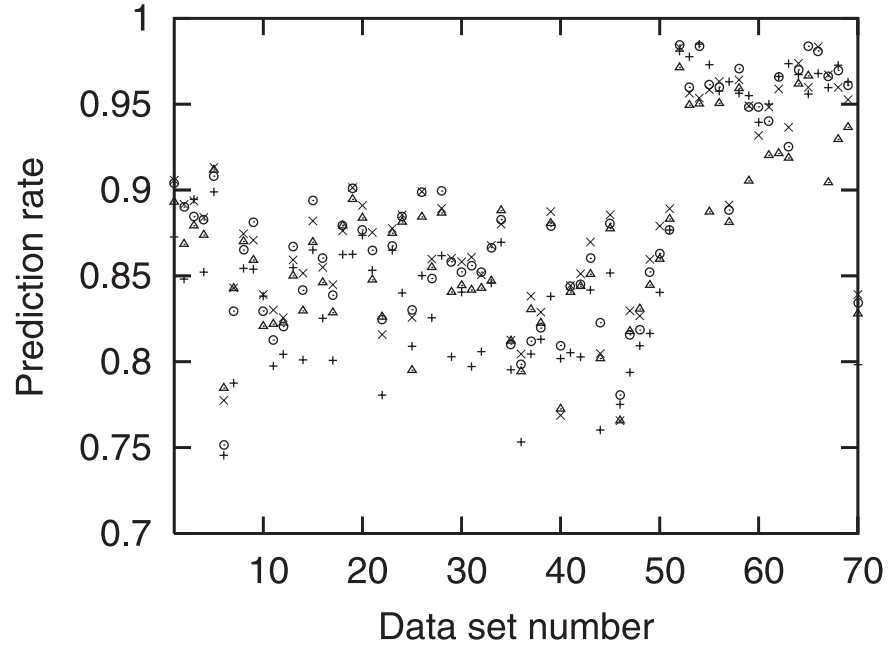


FIG. 6. Disease phenotype prediction rates for all the biological datasets: black X—HINT, red circles—haplotypes, blue +—genotypes, green triangles—strict blocks (GERBIL).

TABLE 1. SNP VALUE PREDICTION RATES[a]

Model	*HINT*	*Haplotypes*	*Genotypes*	*Blocks*
Mean prediction rate	0.86962	0.86496	0.86307	0.84481
Standard deviation	0.00063	0.00064	0.00064	0.00068

[a]The average and standard deviations are calculated on all the biological datasets, using the four different prediction models.

TABLE 2. DISEASE PHENOTYPE PREDICTION RATES ON ALL THE BIOLOGICAL DATASETS, USING THE FOUR DIFFERENT PREDICTION MODELS

Model	*HINT*	*Haplotypes*	*Genotypes*	*Blocks*
Mean prediction ratio	0.88776	0.88671	0.87102	0.87064
Standard deviation	0.00059	0.00059	0.00062	0.00063

5. DISCUSSION

In this paper, we have defined a novel model for genotypes generation. The model aims to reflect the somewhat blocky structure of haplotypes, but allows deviations, i.e., intrablock transitions. A first-order Markov model is kept, without the need to maintain a strict block structure. We have shown how to resolve the model parameters using an EM algorithm.

Our model was examined on a broad spectrum of biological datasets. The prediction rate was used as a measure for the validity of the model. The goal in our experiments was to predict a missing causative SNP, given a training set of genotypes. We have shown that HINT gives more accurate results when compared to simpler models. The advantage is not very large, but is statistically significant. An additional interesting byproduct of our analysis is the conclusion that better predictions are made when using haplotypes compared to using genotypes. The strict blocky structure, on the other hand, seemed to cause loss of information and was less accurate in predicting diseases.

It has been argued that haplotype block structures can be helpful for association studies because each haplotype block can be treated as a single locus with several alleles (the block-specific haplotypes) (Daly *et al.*, 2001). It was shown that finding the blocks of SNPs is expected to contribute to association studies, by decreasing the number of SNPs needed to be genotyped, with minimal statistical power loss (Zhang *et al.*, 2002b). A major problem is that, currently, there are different ways of defining and identifying haplotype blocks (for example, Kimmel *et al.* [2003]; Zhang *et al.* [2002a]; Koivisto *et al.* [2003]). The advantage of blocks is in reducing the number of multiple tests one has to perform, when conducting association studies. Nevertheless, this approach has a drawback, the information loss. Here, we try to take some advantage of the blocks, and by relaxing the model to a “mosaic-like" structure, less information is lost. We plan to explore the power of HINT in disease association studies.

Another interesting question is whether using a higher-order Markov model improves the accuracy. Biologically, an improvement is expected as there is evidence for SNPs that are more linked to distal ones than to closer SNPs (Carlson *et al.*, 2004). On the other hand, adding more parameters to the model is a major disadvantage, as the extent of data we have is limited and the learning process may become infeasible.

A natural alternative to the hidden Markov model is a recombination-based approach, which attempts to reconstruct the past recombination events in the gene genealogy rather than seek ad hoc exchanges of extant haplotypes. The method presented here is much faster, while most investigators that have attempted the ancestral recombination graph (ARG) reconstruction approach have found it to be prohibitively slow (Griffiths and Marjoram, 1996; Song and Hein, 2005). The advantage of the ARG approach is of course that there is an underlying population genetic model, which HINT lacks. Comparing the performance of the two approaches is an interesting research direction.

We have focused our attention here on human genotypes. Haplotype inference and association testing is being applied to other organisms, where the levels of variability and of recombination may be quite different. While HINT can be applied to any organism, performance may not be as good on some organisms. In such cases, retuning of HINT may be necessary, and other approaches that exploit the properties of these organisms' haplotypes may be better.

Another natural extension that we intend to pursue is to include the pedigree information, when available (e.g., HapMap and ENCODE trios), to the HINT inference. Such extension would have a clear advantage by combining the population LD data as HINT does with the transmission information from the trios. (We have already implemented a similar extension for the GERBIL model.)

In our simulation for disease prediction, we assumed for simplicity that the disease trait has 100% penetrance and that it is caused by a single SNP. Obviously, under such assumptions, the trait would be a Mendelian factor and would be mapped by linkage much faster and more easily than by a population association test. None of the diseases that are being mapped by association testing today have such major SNPs, but instead entail multiple SNPs with weak penetrance. Extending the simulation to these more complex and more realistic scenarios is therefore desirable, but it would require substantially larger population sizes.

In this study, our focus was to build a model with improved performance in association studies. We did not aim to improve the phasing per se, and therefore we did not compare phasing quality to that of extant phasing programs. If one wishes to use HINT for phasing, a natural question is how to assign confidence to individual haplotypes or to a phased pair of haplotypes. Naturally, the HMM attributes a probability to each pair of paths it produces. To distinguish among near-optimal solutions, one should find several best-scoring path pairs for each genotype and compare their probabilities. Another criterion for phasing performance is a direct comparison to a known solution, using, e.g., the switch test (Stephens and Donnelly, 2003; Kimmel and Shamir, 2005) or error rate (Stephens and Donnelly, 2003; Kimmel and Shamir, 2004). Note that when phasing a large number of SNPs, the resulting pairs of full haplotypes are less meaningful for association, as most of the SNPs are relatively distant from each other and thus poorly associated. A better definition of phasing and haplotypes that emphasizes locality is needed in these cases.

ACKNOWLEDGMENTS

We thank Eran Halperin for helpful discussions and comments. R.S. was supported by a grant from the Israel Science Foundation (grant 309/02).

REFERENCES

Bafna, V., Halldorsson, B.V., Schwartz, R., Clark, A., and Istrail, S. 2003. Haplotypes and informative SNP selection algorithms: Don't block out information. *Proc. 7th Ann. Int. Conf. on Research in Computational Molecular Biology (RECOMB '03)*, 19–27.

Baum, L.E. 1972. An inequality and associated maximization technique occuring in the statistical analysis of probabilistic functions of Markov chains. *Inequalities* 3, 1–8.

Botstein, D., and Risch, N. 2003. Discovering genotypes underlying human phenotypes: Past successes for Mendelian disease, future approaches for complex disease. *Nature Genet.* 33, 228–237.

Carlson, C.S., Eberle, M.A., Rieder, M.J., Yi, Q., Kruglyak, L., and Nickerson, D.A. 2004. Selecting a maximally informative set of single-nucleotide polymorphisms for association analysis using linkage disequilibrium. *Am. J. Human Genet.* 74(1), 106–120.

Clark, A. 1990. Inference of haplotypes from PCR-amplified samples of diploid populations. *Mol. Biol. Evol.* 7(2), 111–122.

Daly, M.J., Rioux, J.D., Schaffner, S.F., Hudson, T.J., and Lander, E.S. 2001. High-resolution haplotype structure in the human genome. *Nature Genet.* 29(2), 229–232.

Eskin, E., Halperin, E., and Karp, R.M. 2003. Large scale reconstruction of haplotypes from genotype data. *Proc. 7th Ann. Int. Conf. on Research in Computational Molecular Biology (RECOMB '03)*, 104–113.

Excoffier, L., and Slatkin, M. 1995. Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Mol. Biol. Evol.* 12(5), 912–917.

Gabriel, S.B., Schaffner, S.F., Nguyen, H., Moore, J.M., Roy, J., Blumenstiel, B., Higgins, J., DeFelice, M., Lochner, A., Faggart, M., Liu-Cordero, S.N., Rotimi, C., Adeyemo, A., Cooper, R., Ward, R., Lander, E.S., Daly, M.J., and Altshuler, D. 2002. The structure of haplotype blocks in the human genome. *Science* 296, 2225–2229.

Greenspan, G., and Geiger, D. 2003. Model-based inference of haplotype block variation. *Proc. 7th Ann. Int. Conf. on Research in Computational Molecular Biology (RECOMB '03)*, 131–137.

Griffiths, R.C., and Marjoram, P. 1996. An ancestral recombination graph, *in IMA Volume on Mathematical Population Genetics and Its Application*, 257–270. Editors: Donnelly, P., Tavaré, S. Springer, New York.

Gusfield, D. 2002. Haplotyping as perfect phylogeny: conceptual framework and efficient solutions. *Proc. 6th Ann. Int. Conf. on Research in Computational Molecular Biology (RECOMB '02)*, 166–175.

Halperin, E., Kimmel, G., and Shamir, R. 2005. Tag SNP selection in genotype data for maximizing snp prediction accuracy. *Proc. Annual Meeting of the International Society for Computational Biology (ISMB)*, to appear.

Kimmel, G., and Shamir, R. 2004. Maximum likelihood resolution of multi-block genotypes. *Proc. 8th Ann. Int. Conf. on Research in Computational Molecular Biology (RECOMB '04)*, 2–9.

Kimmel, G., and Shamir, R. 2005. GERBIL: Genotype resolution and block identification using likelihood. *Proc. Natl. Acad. Sci. USA* 102, 158–162.

Kimmel, G., Sharan, R., and Shamir, R. 2003. Identifying blocks and sub-populations in noisy SNP data. *Proc. 3rd Workshop on Algorithms in Bioinformatics (WABI '03)*, 303–319.

Kimmel, G., Sharan, R., and Shamir, R. 2004. Computational problems in noisy SNP and haplotype analysis: Block scores, block identification and population stratification. *INFORMS Journal on Computing* 16(4), 360–370.

Koivisto, M., Perola, M., Varilo, T., Hennah, W., Ekelund, J., Lukk, M., Peltonen, L., Ukkonen, E., and Mannila, H. 2003. An MDL method for finding haplotype blocks and for estimating the strength of haplotype block boundaries. *Proc. Pac. Symp. on Biocomputing (PSB '03)* 8, 502–513.

Kruglyak, L., and Nickerson, D.A. 2001. Variation is the spice of life. *Nature Genet.* 27, 234–236.

Long, J., Williams, R.C., and Urbanek, M. 1995. An EM algorithm and testing strategy for multiple-locus haplotypes. *Am. J. Human Genet.* 56(3), 799–810.

Martin, E.R., Lai, E.H., Gilbert, J.R., Rogala, A.R., Afshari, A.J., Riley, J., Finch, K.L., Stevens, J.F., Livak, K.J., Slotterbeck, B.D., Slifer, S.H., Warren, L.L., Conneally, P.M., Schmechel, D.E., Purvis, I., Pericak-Vance, M.A., Roses, A.D., and Vance, J.M. 2000. SNPing away at complex diseases: Analysis of single-nucleotide polymorphisms around APOE in Alzheimer's disease. *Am. J. Human Genet.* 67, 383–394.

McLachlan, G.J., and Krishnan, T. 1997. *The EM Algorithm and Extensions*, John Wiley, New York.

Morris, R.W., and Kaplan, N.L. 2002. On the advantage of haplotype analysis in the presence of multiple disease susceptibility alleles. *Genet. Epidemiol.* 23, 221–233.

Niu, T., Qin, Z.S., Xu, X., and Liu, J.S. 2002. Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms. *Am. J. Human Genet.* 70(1), 157–169.

Patil, N., Berno, A.J., Hinds, D.A., Barrett, W.A., Doshi, J.M., Hacker, C.R., Kautzer, C.R., Lee, D.H., Marjoribanks, C., McDonough, D.P., Nguyen, B.T., Norris, M.C., Sheehan, J.B., Shen, N., Stern, D., Stokowski, R.P., Thomas, D.J., Trulson, M.O., Vyas, K.R., Frazer, K.A., Fodor, S.P., and Cox, D.R. 2001. Blocks of limited haplotype diversity revealed by high-resolution scanning of human chromosome 21. *Science* 294, 1719–1723.

Rissanen, J. 1987. Stochastic complexity. *J. Royal Statist. Soc. B* 49, 223–239.

Sachidanandam, R., Weissman, D., Schmidt, S.C., Kakol, J.M., Stein, L.D., Marth, G., Sherry, S., Mullikin, J.C., Mortimore, B.J., Willey, D.L., Hunt, S.E., Cole, C.G., Coggill, P.C., Rice, C.M., Ning, Z., Rogers, J., Bentley, D.R., Kwok, P.Y., Mardis, E.R., Yeh, R.T., Schultz, B., Cook, L., Davenport, R., Dante, M., Fulton, L., Hillier, L., Waterston, R.H., McPherson, J.D., Gilman, B., Schaffner, S., Van Etten, W.J., Reich, D., Higgins, J., Daly, M.J., Blumenstiel, B., Baldwin, J., Stange-Thomann, N., Zody, M.C., Linton, L., Lander, E.S., and Altshuler, D. 2001. A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms. *Nature* 291, 1298–1302.

Song, Y.S., and Hein, J. 2005. Constructing minimal ancestral recombination graphs. *J. Comp. Biol.* 12(2), 147–169.

Stephens, M., and Donnelly, P. 2003. A comparison of Bayesian methods for haplotype reconstruction from population genotype data. *Am. J. Human Genet.* 73(6), 1162–1169.

Stephens, M., Smith, N.J., and Donnelly, P. 2001. A new statistical method for haplotype reconstruction from population data. *Am. J. Human Genet.* 68(4), 978–989.

Viterbi, A.J. 1967. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Trans. Inform. Theory* IT-13, 260–269.

Wall, J.D., and Pritchard, J.K. 2003. Assessing the performance of the haplotype block model of linkage disequilibrium. *Am. J. Human Genet.* 73(3), 502–515.

Zaykin, D., Westfall, P.H., Young, S.S., *et al.* 2002. Testing association of statistically inferred haplotypes with discrete and continuous traits in samples of unrelated individuals. *J. Human Heredity* 53, 79–91.

Zhang, K., Deng, M., Chen, T., Waterman, M.S., and Sun, F. 2002a. A dynamic programming algorithm for haplotype block partitioning. *Proc. Natl. Acad. Sci. USA* 99(11), 7335–7339.

Zhang, K., Calabrese, P., Nordborg, M., and Sun, F. 2002b. Haplotype block structure and its applications to association studies power and study designs. *Am. J. Human Genet.* 71, 1386–1394.

Zhang, K., Qin, Z., Liu, J., Chen, T., Waterman, M.S., and Sun, F. 2004. Haplotype block partitioning and tag SNP selection using genotype data and their applications to association studies. *Genome Res.* 14(5), 908–916.

Zhang, K., Sun, F., Waterman, M.S., and Chen, T. 2003. Dynamic programming algorithms for haplotype block partitioning: Applications to human chromosome 21 haplotype data. *Proc. 7th Ann. Int. Conf. Research in Computational Molecular Biology (RECOMB '03)*, 332–340.

Zhao, K., Pfeiffer, R., and Gail, M.K. 2003. Haplotype analysis in population genetics and association studies. *Pharmacogenomics* 4(2), 171–178.

Address correspondence to:
Gad Kimmel
School of Computer Science
Tel-Aviv University
Tel-Aviv 69978, Israel

E-mail: kgad@tau.ac.il

BIOINFORMATICS

Vol. 21 Suppl. 1 2005, pages i195–i203
doi:10.1093/bioinformatics/bti1021

Tag SNP selection in genotype data for maximizing SNP prediction accuracy

Eran Halperin[1,†], *Gad Kimmel*[2,*,†] *and Ron Shamir*[2]

[1]*International Computer Science Institute, Berkeley, CA 94704, USA, and*
[2]*School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel*

Received on January 15, 2005; accepted on March 27, 2005

ABSTRACT

Motivation: The search for genetic regions associated with complex diseases, such as cancer or Alzheimer's disease, is an important challenge that may lead to better diagnosis and treatment. The existence of millions of DNA variations, primarily single nucleotide polymorphisms (SNPs), may allow the fine dissection of such associations. However, studies seeking disease association are limited by the cost of genotyping SNPs. Therefore, it is essential to find a small subset of informative SNPs (tag SNPs) that may be used as good representatives of the rest of the SNPs.

Results: We define a new natural measure for evaluating the prediction accuracy of a set of tag SNPs, and use it to develop a new method for tag SNPs selection. Our method is based on a novel algorithm that predicts the values of the rest of the SNPs given the tag SNPs. In contrast to most previous methods, our prediction algorithm uses the genotype information and not the haplotype information of the tag SNPs. Our method is very efficient, and it does not rely on having a block partition of the genomic region.

We compared our method with two state-of-the-art tag SNP selection algorithms on 58 different genotype datasets from four different sources. Our method consistently found tag SNPs with considerably better prediction ability than the other methods.

Availability: The software is available from the authors on request.

Contact: kgad@tau.ac.il

1 INTRODUCTION

Most of the genetic variation among different people can be characterized by single nucleotide polymorphisms (SNPs), which are mutations at single nucleotide positions that occurred during human history and were passed on through heredity. Most of these SNPs are biallelic, i.e. only two bases (alleles) are observed across the population at such sites. It has been estimated that there are about 7 million common SNPs (i.e. SNPs with minor allele frequency of at least 5%) in the human genome (Kruglyak and Nickerson, 2001; Botstein and Risch, 2003). Alleles of SNPs in close physical proximity to each other are often correlated, and the variation of the sequence of alleles in contiguous SNP sites along a chromosomal region (haplotype) is known to be of limited diversity. The identification and analysis of haplotypes, currently a major effort of the international community (http://www.hapmap.org/), is expected to play a key role in trait and disease association studies (Martin *et al*., 2000; Morris and Kaplan, 2002).

The objective of disease association studies is to find genetic factors correlated with complex disease. In these studies, the DNA of individuals from two populations (healthy individuals and carriers of the disease) is sampled. Then, discrepancies in the haplotype structure of the two populations are revealed by various statistical tests. These discrepancies serve as evidence for the correlation of the genomic region studied with the disease.

Clearly, the statistical significance of the study is directly affected by the number of individuals typed. The total cost of the study is also affected by the number of SNPs typed. Therefore, to save resources, one wishes to reduce the number of SNPs typed per individual. This is usually done by choosing an appropriate small subset of the SNPs, called tag SNPs, that could predict the rest of the SNPs with a small error. Thus, when performing a disease association study, the geneticist would experimentally test for association by considering only the tag SNPs, thereby considerably saving resources (or increasing the power of the statistical tests by increasing the number of individuals). Hence, a key problem is to find a set of tag SNPs of minimum size that would have a very good prediction ability. In this paper, we propose a new method, selection of tag SNPs to maximize prediction accuracy (STAMPA) that finds a set of tag SNPs given a genotype sample taken from a set of unrelated individuals.

Finding a high-quality set of tag SNPs is a challenging task for several reasons. One of the main challenges is that the haplotype information is usually not given, and instead we get the genotypes. As opposed to haplotypes, the genotypes give bases at each SNP in both copies of the chromosome,

*To whom correspondence should be addressed.

†The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint first Authors.

but lack the phase, i.e. information as to the chromosome on which each base appears. Owing to technology constraints, most sequencing techniques provide the genotypes and not the haplotypes. There are however, computational tools that use the correlations between neighboring SNPs in order to predict the phase information. Their accuracy depends on the proximity and correlation of the tagged SNPs. When a set of tag SNPs is chosen and then tagged, the rest of the SNPs are not measured and instead must be predicted from this information. The accuracy of such prediction is limited, since the correlation between the tag SNPs is not necessarily as strong as the correlation between SNPs that are in close proximity to each other. One of the advantages of our tag SNPs predictor is that it only uses the genotype information and does not require knowledge of the haplotypes of the tag SNPs. We use the phase information in a reference training set to select the tag SNPs, and subsequently predict the other SNP values in a test individual on the genotype of that individual for the tag SNPs only. To the best of our knowledge, all extant programs that aim to explicitly predict individual SNPs use the haplotypes of the tag SNPs.

Another issue that is crucial in the search for tag SNPs is the definition of an adequate measure of the prediction quality. Many of the current tag SNP selection methods partition the region into blocks of limited diversity (e.g. Zhang *et al*., 2002, 2003, 2004), and find a set of tag SNPs that aims to predict the common haplotypes of each block. There are various disadvantages to such methods, most apparent is the lack of cross-block information and the dependency of the tag SNPs choice on the block definition. We propose here a new natural measure, prediction accuracy, which directly evaluates the average SNP prediction quality.

There is a large body of research on finding a highly predictive set of tag SNPs (Zhang *et al*., 2002; Avi-Itzhak *et al*., 2003; Bafna *et al*., 2003; Carlson *et al*., 2004; Pe'er *et al*., 2004). In contrast to most previous methods, our method uses the genotype information for the tag SNP selection. Zhang *et al*. (2004) have also used genotypes information for tag SNP selection. However, their study selects the SNPs so as to maximize haplotype diversity, and given the genotypes of the tag SNPs in a tested individual it infers blocks and common haplotypes, but does not predict the individual SNPs. Another key difference between our method and previous ones is that we do not rely on any block partition.

We performed extensive tests of STAMPA on genotypes from a variety of sources. Our tests covered 58 datasets from four sources: HapMap project http://www.hapmap.org, ENCODE project http://www.hapmap.org, Daly *et al*. (2001), and Gabriel *et al*. (2002). We show that using STAMPA, very accurate results are achieved. For example, only 17 tag SNPs out of 103 SNPs (16.5%) suffice to attain prediction accuracy of 95% in the data of Daly *et al*. (2001). Our method is also very efficient: runs on a regular PC required seconds to several minutes on all datasets.

We compared our algorithm with two state-of-the-art tag SNP selection algorithms: ldSelect (Carlson *et al*., 2004) and HapBlock (Zhang *et al*., 2004). Our experiments show that STAMPA consistently outperforms both these methods. On the average ldSelect uses ten times more tag SNPs than STAMPA in order to achieve prediction accuracy of 90%. Our algorithm was also more accurate than HapBlock on each of the 58 datasets, sometimes by >15%. Moreover, the running time of STAMPA was much less than HapBlock. For example, on chromosome 5q31 dataset, STAMPA was faster by a factor of 97. Such advantage will be more prominent on future larger datasets.

2 PROBLEM FORMULATION

In order to present our method, we first formalize the problem of tag SNPs prediction. We first need to introduce some notations and definitions. Since we are only interested in biallelic SNPs, we assume that each haplotype is represented by a binary string. Thus, a haplotype of length m is a sequence over $\{0,1\}^m$. A genotype of length m is represented by a $\{0,1,2\}$ sequence, where 0 and 1 stand for the homozygous types $\{0,0\}$ and $\{1,1\}$, respectively, and 2 stands for a heterozygous type. We are given a set of n genotypes $g_1,\ldots,g_n$ of length m each. We use $g_{i,j}$ to denote the j-th component (0, 1 or 2) of the vector g_i. A phasing of a genotype g_i is a pair of haplotypes, $h_i^1, h_i^2 \in \{0,1\}^m$, such that $h_{i,k}^1 \neq h_{i,k}^2$ if $g_{i,k}=2$ and $h_{i,k}^1 = h_{i,k}^2 = g_{i,k}$ if $g_{i,k} \in \{0,1\}$. We also use the notation $g(j)$ to denote the j-th SNP in genotype g.

Consider a genomic region that spans a set of m SNPs. The frequencies of the genotypes in that region across the entire populations are given by some unknown distribution function $\Pr(g_i \in \mathcal{G})$, where $\mathcal{G}$ is the sample space of all genotypes in the population. A prediction algorithm is a function $f:\{0,1,2\}^t \rightarrow \{0,1,2\}^m$. Informally, the prediction algorithm uses the genotype values of the tag SNPs in order to predict the values of the rest of the SNPs. For a given vector $q \in \{0,1,2\}^t$ of tag SNPs values, let $f_j(q)$ denote the j-th component of that vector. Note that f_j refers to the components of the predicted vector of all m SNPs, given the tag genotypes q. Finally, let $z_T:\{0,1,2\}^m \rightarrow \{0,1,2\}^t$ be the restriction of the genotype to the tag SNPs position. For instance, for a set of tag SNPs $T=\{1,3,5,6\}$ the restriction of the genotype $g_i=0122010$ is $z_T(g_i)=0201$.

Our goal is to find a minimum size set of tag SNPs and a prediction algorithm, such that the prediction error is minimized. Formally, for a given t, our objective is to find a set of tag SNPs T of size t and a prediction function f, such that the following expression is minimized.

$$\eta = \sum_{j=1}^{m} \Pr[f_j(z_T(g)) \neq g(j)], \qquad (1)$$

where the probability is over the sample space given by $\Pr(g \in \mathcal{G})$. In other words, for a randomly picked individual

ALGORITHM Predict(i, j_1, j_2, a_1, a_2)

Input: i, j_1, $j_2 \in \{1, \ldots, m\}$, and $a_1, a_2 \in \{0, 1, 2\}$.
Output: An integer $v \in \{0, 1, 2\}$ which is a predicted value of a SNP in position i, given that in position j_1 and j_2 the values are a_1 and a_2 respectively.

1. For every $(x, y, z) \in \{0, 1\}^3$ we let $C(x, y, z) = \{(j, p) \mid h^p_{jj_1} = x, h^p_{jj_2} = y, h^p_{ji} = z\}$ be the set of haplotypes having the values x, y, z in positions j_1, i and j_2 respectively.
2. Let $A(x, y) = z \in \{0, 1\}$, where $|C(x, y, z)| \geq |C(x, y, 1 - z)|$ breaking ties arbitrarily.
3. Let $c(x, y) = |C(x, y, 0)| + |C(x, y, 1)|$.
4. We compute the values of two variables x, y using the following case analysis.
   - If $a_1 < 2$ and $a_2 < 2$, then we set $x = y = A(a_1, a_2)$.
   - If $a_1 = 2$, $a_2 = 2$ and $c(0, 0) \cdot c(1, 1) \geq c(0, 1) \cdot c(1, 0)$, then $x = A(0, 0)$ and $y = A(1, 1)$.
   - If $a_1 = 2$, $a_2 = 2$ and $c(0, 0) \cdot c(1, 1) < c(0, 1) \cdot c(1, 0)$, then $x = A(0, 1)$ and $y = A(1, 0)$.
   - If $a_1 = 1$, $a_2 = 2$ ($a_2 = 1$, $a_1 = 2$), then we set $x = A(1, 1)$ and $y = A(1, 0)$ ($y = A(0, 1)$).
   - If $a_1 = 0$, $a_2 = 2$ ($a_2 = 0$, $a_1 = 2$), then we set $x = A(0, 0)$ and $y = A(0, 1)$ ($y = A(1, 0)$).
5. If $x \neq y$ output 2, else output x.

Fig. 1. The procedure Predict. We implicitly assume that the training set and its phase are given. The variables x and y computed by the case analysis represent the majority votes for the two haplotypes induced by the values a_1 and a_2. Note that the output value is determined by simply counting the frequencies of different partial haplotypes in the training set that match a_1 and a_2 and taking the majority vote.

from the population, we want to minimize the expected number of prediction errors.

The main problem in achieving this goal is that the frequencies of the genotypes in the population are unknown. Therefore, we use a training dataset of genotypes, $g_1, \ldots, g_n$ in order to learn the distribution of genotypes in the data. For a given prediction algorithm $f : \{0, 1, 2\}^t \to \{0, 1, 2\}^m$, we are interested in finding a set of tag SNPs T of size t, such that expression (1) is minimized when the genotype is randomly picked from the training set. Formally, if $X_T = |\{(i, j) \mid g_{i,j} \neq f_j(z_T(g_i))\}|$, where $g_{i,j}$ is the j-th SNP of g_i, then we are looking for a set T of SNPs of size t such that X_T is minimized. The resulting prediction rate of the tag SNPs depends both on the prediction function f and on the choice of the tag SNPs.

3 THE PREDICTION ALGORITHM

In this section we present our prediction algorithm. The algorithm is based on the observation made by several biological studies, that the correlation between SNPs tends to decay as the physical distance increases (Gabriel *et al.*, 2002; Bafna *et al.*, 2003; Daly *et al.*, 2001; Kimmel and Shamir, 2004). We assume that given the genotypes values of two SNPs, the probabilities of the values at any intermediate SNPs do not change by knowing the values of additional distal ones. Formally, this assumption can be stated as:

$$\forall s\colon a < s < b,\ \forall q\colon q < a \text{ or } q > b,\ \forall v \in \{0, 1, 2\},\ \forall i\colon \\ \Pr[g_{i,s} = v | g_{i,a}, g_{i,b}] \approx \Pr[g_{i,s} = v | g_{i,a}, g_{i,b}, g_{i,q}]. \quad (2)$$

Thus, our prediction function predicts an SNP value using only the values of the two closest tag SNPs to this SNP. To be precise, $f_i(z_T(g)) = f(g_{j_1}, g_j, g_{j_2})$, where j_1 and j_2 are the closest tag SNPs to j, on both sides, if possible. Although many biological studies support this assumption, it clearly does not hold for all SNPs or in all datasets. However, the assumption is a rather faithful approximation of the reality in most cases. As we shall show in Section 5, using this assumption we achieve very high-prediction rates.

Given a set of tag SNPs $T = (s_1, \ldots, s_t)$, we use the procedure Predict given in Figure 1 to predict the value of SNP i given the value of the tag SNPs. We assume that we are given the training set of genotypes $g_1, \ldots, g_n$ together with their corresponding haplotypes $h^1_1, h^2_1, h^1_2, \ldots, h^2_n$, where $h^j_i = (h^j_{i1}, \ldots, h^j_{im}) \in \{1, 2\}^m$ for $j = 1, 2$. The haplotypes can be computed from the genotypes using a variety of available algorithms (Kimmel and Shamir, 2005; Eskin *et al.*, 2003; Stephens and Donnelly, 2003; Greenspan and Geiger, 2003).

Let j_1 and j_2, $j_1 < i < j_2$ be the positions of the tag SNPs closest to position i on both sides. If there is no tag SNP in position $j_2 > j$, then j_1 and j_2 are the last two SNPs, and if there is no tag SNP in position $j_1 < j$ then j_1 and j_2 are the first two SNPs. The procedure Predict(i, j_1, j_2, a_1, a_2) uses a majority vote in order to determine which value is more likely to appear in position i given that positions j_1 and j_2 have the values $a_1 \in \{0, 1, 2\}$ and $a_2 \in \{0, 1, 2\}$, respectively. In order to use the phased information given by the model, we use two majority votes to determine the two different alleles. For

instance, if $a_1 = 0$ and $a_2 = 2$, we find the most likely allele given that the alleles in positions j_1 and j_2 are both 0, and another allele given that the alleles in positions j_1 and j_2 are 0 and 1, respectively. Further details are given in Figure 1. Note that predicting SNP i using the procedure Predict makes no use of most of the tag SNPs—we simply ignore all the tag SNPs except for the ones closest to i.

4 ALGORITHMS FOR TAG SNP SELECTION

Recall that our goal is to find a set of tag SNPs T of size t, such that X_T is minimized, where $X_T = |\{(i,j) \mid g_{i,j} \neq \text{Predict}(j, j_1, j_2, g_{i,j_1}, g_{i,j_2})\}|$. We give two algorithms for selecting the tag SNPs. Both algorithms use the prediction algorithm as a subroutine. The first is a polynomial algorithm that guarantees an optimal solution. The second is a simpler and faster random sampling algorithm. We shall discuss their performance in Section 5.

4.1 An exact algorithm

We now describe an algorithm that solves this problem to optimality. The algorithm, STAMPA, uses dynamic programming.

Let $X_T^{i,j} = 1$ if $g_{i,j} \neq \text{Predict}(j, j_1, j_2, g_{i,j_1}, g_{i,j_2})$ and let $X_T^{i,j} = 0$ otherwise. Clearly, $X_T = \sum_{i,j} X_T^{i,j}$. For every pair of SNPs $m_1 < m_2$ we next define three auxiliary score functions, $\text{score}(m_1, m_2)$, $\text{score}_1(m_1, m_2)$ and $\text{score}_2(m_1, m_2)$, which will be used in the dynamic program recursion. These score functions evaluate the expected number of errors in a subregion (a contiguous set of SNPs), given a partial set of the tag SNPs. We assume that $m_1, m_2 \in T$ and that for each $m_1 \leq j \leq m_2$, $j \notin T$. Then, we define

$$\text{score}(m_1, m_2) = \sum_{i=1}^{n} \sum_{j=m_1}^{m_2 - 1} X_T^{i,j}.$$

Thus, $\text{score}(m_1, m_2)$ is the total number of prediction errors in SNPs $m_1, \ldots, m_2 - 1$, given that m_1 and m_2 are tag SNPs, and that there are no tag SNPs between m_1 and m_2. Since the procedure Predict infers an SNP value by considering only its neighboring tag SNPs, we can readily compute the score, while disregarding the information on all the other tag SNPs.

For $\text{score}_1(m_1, m_2)$, we assume that m_1 and m_2 are the last two tag SNPs. Then, the score is defined as

$$\text{score}_1(m_1, m_2) = \sum_{i=1}^{n} \sum_{j=m_1}^{m} X_T^{i,j}.$$

Thus, $\text{score}_1(m_1, m_2)$ is the the total number of prediction errors in SNPs $m_1, \ldots, m$ when the last two SNPs are in positions m_1, m_2. Again, since Predict only uses the closest tag SNPs in order to compute the SNP values, we can compute score_1 independently of the locations of the rest of the SNPs.

Similarly, for $\text{score}_2(m_1, m_2)$ we assume that m_1 and m_2 are the first two tag SNPs, and define

$$\text{score}_2(m_1, m_2) = \sum_{i=1}^{n} \sum_{j=1}^{m_2 - 1} X_T^{i,j}.$$

In this case, $\text{score}_2(m_1, m_2)$ is the total number of prediction errors in SNPs $1, \ldots, m_2 - 1$ when the first two SNPs are in positions m_1, m_2.

We next define the function f that will be used in the dynamic programming recursion. $f(m^*, l)$ is defined for $l \geq 2$ and $1 \leq m^* \leq m$. For $l < t$, the function $f(m^*, l)$ represents the minimum number of prediction errors in SNPs $1, 2, \ldots, m^*$, given that the l-th tag SNP is in position m^*. For $l = t$, the function $f(m^*, t)$ represents the minimum number of prediction errors in all SNPs $1, 2, \ldots, m$ given that the last tag SNP is in position m^*. Formally, we define $f(m^*, l)$ in the following way:

- For $l = t$, $f(m^*, t) = \sum_{i=1}^{n} \sum_{j=1}^{m} X_T^{i,j}$ when the last tag SNP is in position m^*.
- For $t > l \geq 2$, $f(m^*, l) = \sum_{i=1}^{n} \sum_{j=1}^{m^* - 1} X_T^{i,j}$ when the l-th tag SNP is in position m^*.

It is easy to verify by the definitions of f and of score, score_1 and score_2, that the following recurrence relation holds:

$$f(m^*, l) = \begin{cases} \min_{1 \leq m' < m^*} \text{score}_2(m', m^*), & l = 2, \\ \min_{l-1 \leq m' < m^*} \{f(m', l-1), & \\ \quad + \text{score}(m', m^*)\}, & 2 < l < t \\ \min_{t-1 \leq m' < m^*} \{f(m', t-1) & \\ \quad + \text{score}_1(m', m^*)\}, & l = t. \end{cases} \tag{3}$$

We now apply dynamic programming in order to find the value of $f(m^*, t)$ for every $t \leq m^* \leq m$, using the above recurrence relation. Since $f(m^*, t)$ is the total number of prediction errors given that the last tag SNP is in position m^*, it is clear that the minimum value of X_T over all possible sets of tag SNPs of size t is $\min_{\{m^* | t \leq m^* \leq m\}} f(m^*, t)$. Using back pointers in the process, one can also find a set of tag SNPs minimizing X_T.

4.1.1 Complexity analysis We first compute the three scores for all $\binom{m}{2}$ possible pairs of SNPs. For every pair the running time is $O(mn)$. Hence, the total running time for this stage is $O(m^3 n)$. We keep the scores in a matrix and we use that matrix in order to compute f. Given the computed scores, for every $m^* \leq m$, computing $f(m^*, 2)$ takes $O(m^*)$, so doing this for all m^* takes $O(m^2)$. Similarly, computing $f(m^*, i)$ for every $i < t, m^* < m$ takes $O(m^2 t)$. Finally, computing $f(m^*, t)$ for every $m^* \leq m$ takes $O(m^2)$. Since $t \leq m$ the total running time is $O(m^3 n)$.

If the number of SNPs is large (even in the hundreds), a running time of $O(m^3 n)$ is very expensive. However, in practice, the correlation between SNPs is decaying when the physical distance between the SNPs increases. Put differently, tag

SNPs tend to predict well other SNPs in the same or neighboring block, but not farther away. Thus, having a very large distance between neighboring tag SNPs yields poor prediction power. Hence, in most practical cases one can use a bound c on the maximal distance in SNPs between neighboring tag SNPs. c will depend on the SNP typing density and will typically not exceed 20 or 30. In such a case, computing score(m_1, m_2) take $O(mc^2n)$ and computing score$_1$ and score$_2$ take $O(c^3n)$. Computing $f(m^*, i)$ for each i takes $O(mtc)$. Thus, the total running time is $O(mtc + mc^2n) = O(mc(cn + t))$.

4.2 Random sampling

In some cases we are interested in finding quickly a very small number of tag SNPs that roughly predict the rest of the SNPs, i.e. we are willing to give up some of the prediction power if we can get a very small number of tag SNPs. In these cases, the assumption that the tag SNPs are close to each other cannot be made, since c is very large, and the exact algorithm may be too slow. We therefore suggest a very simple and much more efficient algorithm that does not guarantee optimal results.

The algorithm is as follows: We generate 100 sets of tag SNPs, $T_1, T_2, \ldots, T_{100}$, each generated by randomly picking t positions out of the m possible positions. We then compute X_{T_i} for $i = 1, 2, \ldots, 100$, and we choose the set of tag SNPs T_i that minimizes X_{T_i}. This algorithm is very naive, but we show that it gives reasonable results in practice.

5 RESULTS ON BIOLOGICAL DATASETS

5.1 Description of the datasets

We used four datasets encompassing 58 different genomic regions.

- A dataset from the works of Daly *et al*. (2001). In this study, genotypes for 103 SNPs, from a 500 kb region of chromosome 5q31, were collected from 129 mother, father and child trios from European derived population in an attempt to identify a genetic risk for Crohn's disease. We only used the population of children in this dataset.
- Population D from the study of Gabriel *et al*. (2002). The data consist of 51 sets of genotypes from various genomic regions, with number of SNPs per region ranging from 13 to 114. The sets contained 30 mother, father, child trios that were taken from a Yoruba's population, from which we only used the 60 genotypes of the parents.
- Regions ENm013, ENr112 and ENr113 of the ENCODE project (http://www.hapmap.org). These are 500 kb regions of chromosomes $7q21.13, 2p16.3$ and $4q26$, which were collected from 30 trios. The number of SNPs genotyped in each region is 361, 412 and 515, (thus, the density of the sample is 3–5 times greater than the density of Daly *et al*., 2001). We used the 60 genotypes corresponding to the parents from each dataset.
- Genotypes from the HapMap project (http://www.hapmap.org). We used three sets of SNPs spanning the three genes PP2R4, STEAP and TRPM8. For each of these genes we took the HapMap SNPs that are spanned by the gene plus 10 kb upstream and downstream. The resulting sets contain 39, 23 and 102 SNPs. In this dataset we used the genotypes of the parents.

5.2 Implementation

STAMPA was implemented in C. All reported runs used a Linux operating system on a 4 Ghz PC using 500 M cache. Running times are discussed below (Fig. 3 and Table 2).

The Predict procedure requires a phased training set. To obtain that solution when applying STAMPA, we used the GERBIL algorithm (Kimmel and Shamir, 2005). Running times for phasing using GERBIL were almost always <1 min. The dataset of Daly *et al*. (2001) required the most time, $\sim$2 min. These times are not included in the reporting below.

5.3 Exact solution versus random sampling algorithm

We first measured the prediction accuracy of the two algorithms in Section 4. For STAMPA, we used $c = 30$ as the upper bound of distance between tag SNPs. The experiments were performed in a leave-one-out manner: We repeatedly removed one of the genotypes from the set, used the remaining genotypes as the training set in order to find a set of tag SNPs, and used these tag SNPs in order to predict the other SNPs in the removed genotype.

The results show that STAMPA uses very few tag SNPs in order to predict the other SNPs with high confidence. For example, in chromosome 5q31 dataset (Daly *et al*., 2001), typing 2 SNPs suffices to predict all the 103 SNPs with 80% accuracy, 6 SNPs are needed to achieve 90% and only 17 SNPs need to be typed for 95%.

The results of the comparison of the two algorithms are summarized in Figure 2. As expected, in most cases, STAMPA was more accurate than the random sampling algorithm. However, when the number of tag SNPs is small, there is a clear advantage for the random sampling algorithm. For example, in Encode region ENr113, when less than 15 tag SNPs are required, the prediction accuracy of the random sampling algorithm was high. This gap can be explained by the fact that when the number of tag SNPs is small, the upper bound for the distance between tag SNPs is too restrictive for STAMPA. It is important to emphasize, that each of the two algorithms has a parameter, that can be increased to obtain more accurate results, but at the expense of larger running times. Such is the parameter c in STAMPA, and the number of samples in the random sampling algorithm. Although in our experiments we saw a clear advantage to STAMPA, in some situations we expect the opposite to be true,

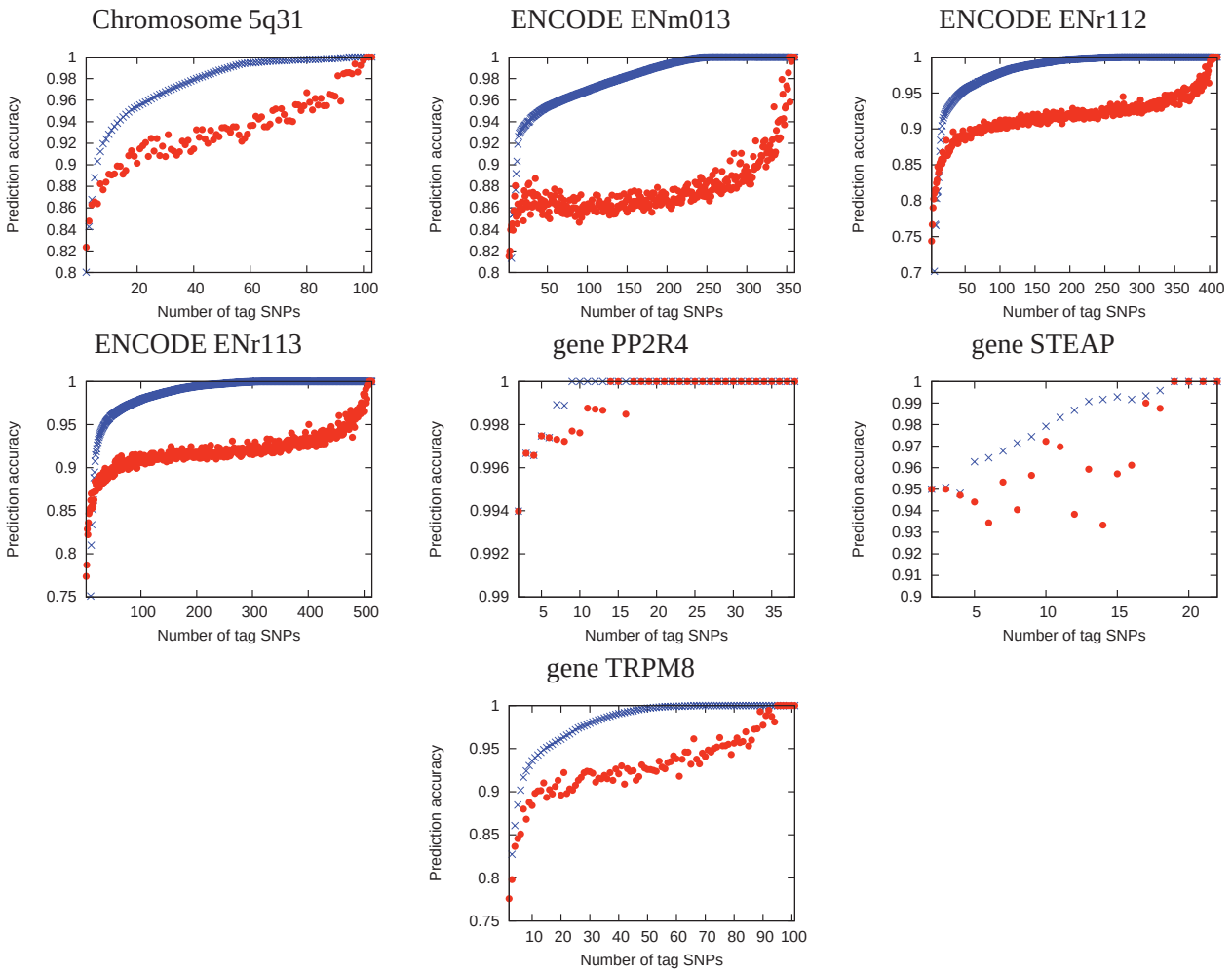


Fig. 2. Prediction accuracy as a function of the number of tag SNPs used in the two selection algorithms. Blue X, STAMPA, red circles, random sampling algorithm.

e.g. when SNPs are genotyped with high density in a very long region and the number of tag SNPs is required to be very small.

5.4 Comparisons to extant methods

We chose to compare our algorithm with two recent algorithms for tag SNP selection that are widely used: ldSelect, an algorithm suggested by Carlson *et al*. (2004), which uses a greedy approach, and HapBlock suggested by Zhang *et al*. (2004), which uses dynamic programming and a partition-ligation EM subroutine to phase subintervals in the recursion. Two additional tag SNP selection algorithms that were reported in the literature (Bafna *et al*., 2003; Pe'er *et al*., 2004) could not be included in the comparisons since their implementations were not available.

In order to evaluate the prediction accuracy of a tag SNP selection algorithm, one has to provide a prediction algorithm such as Predict. Unfortunately, ldSelect and HapBlock do not provide a prediction algorithm. Hence, in order to evaluate the prediction accuracy of these algorithms, we had to choose a prediction algorithm for each of them.

ldSelect requires phased genotypes as input. We used PHASE (Stephens and Donnelly, 2003) to obtain the phasing solution, since it is a widely used and highly accurate phasing program (Kimmel and Shamir, 2005). The output of the program is sets of SNPs and for each one a subset of its tag SNPs. SNPs in a set are not necessarily contiguous. We used a majority vote of the tag SNPs inside each set as the prediction method of SNPs in this set. (This rule is equivalent to that of Predict in the case of two tag SNPs, with the key difference that Predict assumes a specific order of the two tag SNPs and the predicted one.)

HapBlock gets as input a genotype matrix and outputs the tag SNPs. There are several input parameters for this software, such as the algorithm for block partitioning and the method of tag SNP selection. Additional numeric parameters

Table 1. Performance of STAMPA and ldSelect. The number of tag SNPs needed in order to reach accuracies of 80 and 90% by each algorithm is listed

Dataset	80% accuracy		90% accuracy		Total number of SNPs
	STAMPA	ldSelect	STAMPA	ldSelect	
5q31	2	64	6	91	103
Gabriel *et al.*	3.4 (1.8)	41.6 (14.8)	12.1 (6.3)	51 (17.8)	55.6 (20.2)
ENm013	5	84	12	189	360
ENr112	9	97	17	169	411
ENr113	11	83	18	325	514
PP2R4	2	6	2	6	38
STEAP	2	20	2	22	22
TRPM8	3	38	6	53	101

For the data of Gabriel *et al.* (2002) the first number is the average over all 51 datasets, followed by the standard deviation in parentheses. See Figure 3 for more detailed results on these datasets.

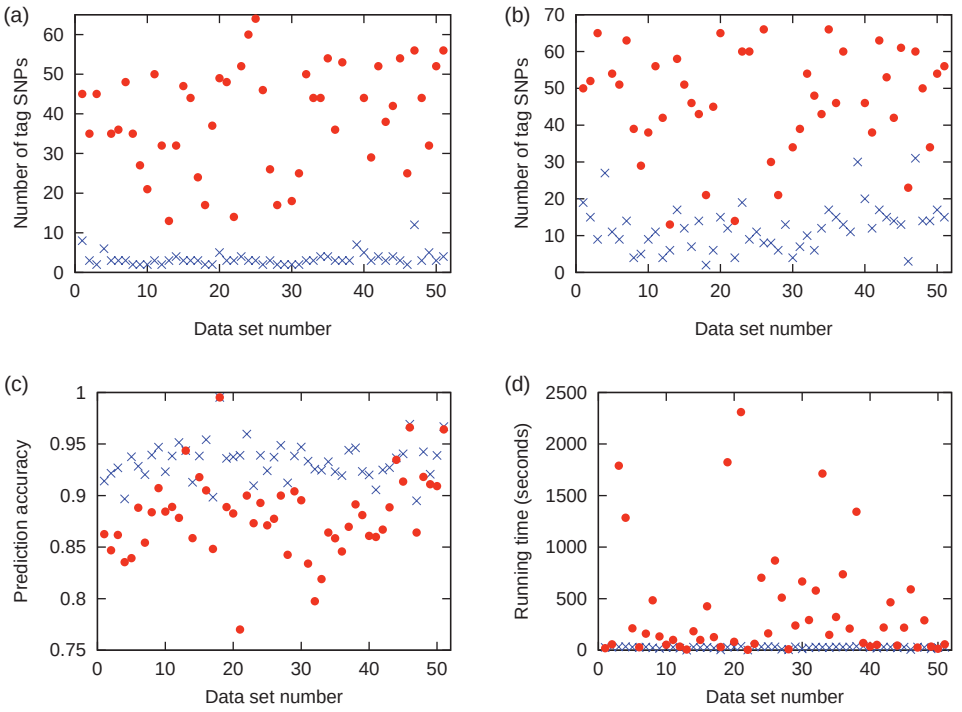


Fig. 3. Performance of STAMPA, ldSelect and HapBlock on each of the 51 genotyped regions in Gabriel *et al.* (2002). The x-axis is the 51 datasets in an arbitrary order; blue cross—STAMPA, red circles—the other algorithm. Comparison with ldSelect: the number of tag SNPs found by the algorithm to reach an accuracy of 80% (**a**) and 90% (**b**). Comparison with HapBlock: prediction accuracy (**c**) and running times (**d**) of the algorithms on each dataset.

are required, e.g. a threshold for common haplotypes. We used the default values presented in the software manual (http://www.cmb.csu.edu/~msms/HapBlock). Since the input to this program is unphased genotypes and no prediction algorithm was suggested, we used our own prediction algorithm (Section 3) to measure the accuracy of tag SNPs chosen by the algorithm.

In Table 1 and in Figure 3 we give a summary of the comparison of STAMPA with ldSelect. In each of the methods, we searched for the minimal number of tag SNPs needed in order to reach accuracies of at least 80 and 90%. Since the input format of ldSelect does not allow specifying the number of tag SNPs, but rather the Pearson correlation value between the tag SNPs and the predicted SNPs, we searched for the minimal Pearson correlation value needed in order to reach 80% (or 90%) accuracy. Reducing the value of the Pearson correlation results in a smaller number of tag SNPs. Our experiments show that STAMPA consistently outperforms ldSelect. On average, ldSelect uses 10 times more tag SNPs than STAMPA in order to reach an accuracy of 90%.

Table 2. Prediction accuracy and running times of STAMPA and HapBlock

Dataset	Number of tag SNPs	Prediction accuracy STAMPA	HapBlock	Running times (s) STAMPA	HapBlock
5q31	17	0.949	0.889	179	17 311
ENm013	15	0.929	0.759	78	8710
ENr112	33	0.939	0.822	87	3810
STEAP	3	0.951	0.763	3	5
TRPM8	12	0.942	0.811	34	140
Gabriel *et al*.	16.9 (6.5)	0.932 (0.019)	0.88 (0.04)	1282	20 131

The number of tag SNPs is determined according the output of HapBlock software, using its default parameters. No comparison could be performed on ENr113 since HapBlock gave no solution due to memory overload. The gene PP2R4 was dropped since HapBlock outputs only one tag SNP for that gene, so comparison was meaningless. For the data of Gabriel *et al*. (2002) the first number is the average over all 51 datasets, followed by the standard deviation in parentheses; running times are totals over all 51 datasets. See Figure 3 for more detailed results on these datasets.

In Table 2 and Figure 3 we give a summary of the comparison of STAMPA and HapBlock . We used the same number of tag SNPs generated by HapBlock to select tag SNPs with STAMPA. In all the 58 datasets STAMPA was more accurate. Moreover, the running time of STAMPA was much less than HapBlock. For example, on chromosome 5q31 dataset, STAMPA was faster by a factor of 97. Such advantage will be more prominent on future larger datasets.

6 DISCUSSION

In this paper, we have defined a novel measure for evaluating the quality of tag SNP selection. The measure we use, prediction accuracy, has a very simple and intuitive meaning: it aims to maximize the expected accuracy of predicting untyped SNPs, given the unphased (genotype) information of the tag SNPs. The prediction itself is done using a simple majority vote. By making an additional natural approximate assumption that SNP values can be determined best based on the values of their nearest tag SNPs on each side, the prediction becomes quite simple, and the optimal selection of tag SNPs can be done in polynomial time.

We presented a method for tag SNPs selection and for SNP prediction based on the genotype values of the tag SNPs. Our selection method, called STAMPA, is unique in its treatment of the prediction part. Most extant methods for tag SNP selection (Zhang *et al*., 2002; Avi-Itzhak *et al*., 2003; Bafna *et al*., 2003; Carlson *et al*., 2004; Pe'er *et al*., 2004) rely on haplotype information that is often not readily available in real life scenarios. One exception is the HapBlock algorithm (Zhang *et al*., 2004), which selects the tag SNPs based on the genotypes and not on the haplotypes. However, HapBlock selects the tag SNPs in order to maximize diversity of the common haplotypes in blocks, and it is not clear whether this method could be easily extended to an SNP prediction algorithm using genotype data for the tag SNPs.

Another difference between STAMPA and HapBlock is in the use of phasing: Although both methods employ the dynamic programming approach, HapBlock solves many phasing subproblems in the dynamic programming recursion, determines the blocks and selects the tag SNPs in each block. In contrast, STAMPA uses phased data for the training set and then employs only the much simpler and faster prediction algorithm in the recursion. This is the reason the latter algorithm is much faster.

We presented two tag SNP selection algorithms, one based on dynamic programming and the other based on random sampling. The dynamic programming algorithm guarantees an optimal solution in polynomial time, but may be prohibitively slow in practice when the number of tag SNPs is large. A practical compromise that we used is to limit the distance between neighboring tag SNPs. Under this restriction optimality is not guaranteed anymore, but our results using over 50 different genotype sets show that accuracy is very good in most cases even with a modest distance bound ($c = 30$). The distance-bounded dynamic programming approach usually provides better results than the random sampling approach. These findings are consistent with the report in Zhang *et al*. (2004), where a different criterion (power) was used to evaluate random sampling and HapBlock performance on simulated data. However, the random sampling algorithm is very efficient, and therefore we believe that it may be useful in some specific situations, e.g. on large datasets where a very sparse set of tag SNPs is sought.

In comparison with another tag SNP selection algorithm, ldSelect (Carlson *et al*., 2004), STAMPA consistently obtained higher accuracy. This is not surprising, since ldSelect uses a simple greedy approach. Interestingly, even the random sampling approach outperformed ldSelect (data not shown). ldSelect has the added flexibility to select tag SNPs for non-contiguous sets of SNPs, and thus may have an advantage over STAMPA in the cases where the LD does not decay with distance.

What is the best measure for selecting tag SNPs? The answer is still not clear, and also depends on the context. We propose here the expected prediction accuracy, and show that under reasonable assumptions it yields an efficient and

accurate method for selection. All the same, other criteria have been proposed. If the ultimate goal is to detect disease association, the power of a selection method may be evaluated using this criterion. We intend to explore the power of STAMPA in disease association in the future. Another objective may be to maximize the distinction between common haplotypes in blocks. STAMPA does not provide common haplotypes and does not assume any block structure, which simplifies the algorithmics but may be viewed as a disadvantage. Our work shows that if the expected number of errors is of interest, then our algorithms provide more accurate prediction compared with the existing algorithms.

ACKNOWLEDGEMENT

R.S. holds the Raymond and Beverly Sackler Chair in Bioinformatics at Tel Aviv University and was supported in part by the Israeli Science Foundation (grant 309/02).

REFERENCES

Avi-Itzhak,H.I., Su,X. and De La Vega,F.M. (2003) Selection of minimum subsets of single nucleotide polymorphisms to capture haplotype block diversity. *Proc. Pacific Symp. Biocomput. (PSB 03)*, **8**, 466–477.

Bafna,V., Halldorsson,B.V., Schwartz,R., Clark,A. and Istrail,S. (2003) Haplotypes and informative SNP selection algorithms: don't block out information. In *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)*. The Association for Computing Machinery, pp. 19–27.

Botstein,D. and Risch,N. (2003) Discovering genotypes underlying human phenotypes: past successes for Mendelian disease, future approaches for complex disease. *Nat. Genet.*, **33**, 228–237.

Carlson,C.S., Eberle,M.A., Rieder,M.J., Yi,Q., Kruglyak,L. and Nickerson,D.A. (2004) Selecting a maximally informative set of single-nucleotide polymorphisms for association analysis using linkage disequilibrium. *Am. J. Human Genet.*, **74**, 106–120.

Daly,M.J., Rioux,J.D., Schaffner,S.F., Hudson,T.J. and Lander,E.S. (2001) High-resolution haplotype structure in the human genome. *Nat. Genet.*, **29**, 229–232.

Eskin,E., Halperin,E. and Karp,R.M. (2003) Large scale reconstruction of haplotypes from genotype data. In *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)*. The Association for Computing Machinery, pp. 104–113.

Gabriel,S.B., Schaffner,S.F., Nguyen,H., Moore,J.M., Roy,J., Blumenstiel,B., Higgins,J., DeFelice,M., Lochner,A., Faggart,M., *et al*. (2002) The structure of haplotype blocks in the human genome. *Science*, **296**, 2225–2229.

Greenspan,G. and Geiger,D. (2003) Model-based inference of haplotype block variation. In *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)*. The Association for Computing Machinery, pp. 131–137.

Kimmel,G. and Shamir,R. (2004) Maximum likelihood resolution of multi-block genotypes. In *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 04)*. The Association for Computing Machinery, pp. 2–9.

Kimmel,G. and Shamir,R. (2005) GERBIL: genotype resolution and block identification using likelihood. *Proc. Natl Acad Sci US A*, **102**, 158–162.

Kruglyak,L. and Nickerson,D.A. (2001) Variation is the spice of life. *Nat. Genet.*, **27**, 234–236.

Martin,E.R., Lai,E.H., Gilbert,J.R., Rogala,A.R., Afshari,A.J., Riley,J., Finch,K.L., Stevens,J.F., Livak,K.J., Slotterbeck,B.D. *et al*. (2000) SNPing away at complex diseases: analysis of single-nucleotide polymorphisms around APOE in Alzheimer's disease. *Am. J. Human Genet.*, **67**, 383–394.

Morris,R.W. and Kaplan,N.L. (2002) On the advantage of haplotype analysis in the presence of multiple disease susceptibility alleles. *Genet. Epidemiol.*, **23**, 221–233.

Pe'er,I., Bakker,P., Barret,J.C., Altshuler,D. and Daly,M.J. (2004) A branch and bound algorithm for the chromosome tagging problem. In *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes*, pp. 89–98.

Stephens,M. and Donnelly,P. (2003) A comparison of Bayesian methods for haplotype reconstruction from population genotype data. *Am. J. Human Genet.*, **73**(6), 1162–1169.

Zhang,K., Deng,M., Chen,T., Waterman,M.S. and Sun,F. (2002) A dynamic programming algorithm for haplotype block partitioning. *Proc. Natl Acad. Sci. USA*, **99**, 7335–7339.

Zhang,K., Sun,F., Waterman,M.S. and Chen,T. (2003) Dynamic programming algorithms for haplotype block partitioning: applications to human chromosome 21 haplotype data. In *Proceedings of The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)*. The Association for Computing Machinery, pp. 332–340.

Zhang,K., Qin,Z., Liu,J., Chen,T., Waterman,M.S. and Sun,F. (2004) Haplotype block partitioning and tag SNP selection using genotype data and their applications to association studies. *Genome Res.*, **14**, 908–916.

A Fast Method for Computing High-Significance Disease Association in Large Population-Based Studies

Gad Kimmel and Ron Shamir

Because of rapid progress in genotyping techniques, many large-scale, genomewide disease-association studies are now under way. Typically, the disorders examined are multifactorial, and, therefore, researchers seeking association must consider interactions among loci and between loci and other factors. One of the challenges of large disease-association studies is obtaining accurate estimates of the significance of discovered associations. The linkage disequilibrium between SNPs makes the tests highly dependent, and dependency worsens when interactions are tested. The standard way of assigning significance (*P* value) is by a permutation test. Unfortunately, in large studies, it is prohibitively slow to compute low *P* values by this method. We present here a faster algorithm for accurately calculating low *P* values in case-control association studies. Unlike with several previous methods, we do not assume a specific distribution of the traits, given the genotypes. Our method is based on importance sampling and on accounting for the decay in linkage disequilibrium along the chromosome. The algorithm is dramatically faster than the standard permutation test. On data sets mimicking medium-to-large association studies, it speeds up computation by a factor of 5,000–100,000, sometimes reducing running times from years to minutes. Thus, our method significantly increases the problem-size range for which accurate, meaningful association results are attainable.

Linking genetic variation to personal health is one of the major challenges and opportunities facing scientists today. It was recently listed as 1 of the 125 "big questions" that face scientific inquiry over the next quarter century.[1] The accumulating information about human genetic variation has paved the way for large-scale, genomewide disease-association studies that can find gene factors correlated with complex disease. Preliminary studies have shown that the cumulative knowledge about genome variation is, indeed, highly instrumental in disease-association studies.[2–4]

The next few years hold the promise of very large association studies that will use SNPs extensively.[5] There are already reported studies with 400–800 genotypes,[6] and studies with thousands of genotypes are envisioned.[6] High-throughput genotyping methods are progressing rapidly.[7] The number of SNPs typed is also likely to increase with technological improvements: DNA chips with >100,000 SNPs are in use,[8] and chips with 500,000 SNPs are already commercially available (Affymetrix). Hence, it is essential to develop computational methods to handle such large data sets. Our focus here is on improving a key aspect in the mathematical analysis of population-based disease-association studies.

The test for association is usually based on the difference in allele frequency between case and control individuals. For a single SNP, a common test suggested by Olsen et al.[9] is based on building a contingency table of alleles compared with disease phenotypes (i.e., case/control) and then calculating a χ^2-distributed statistic. When multiple markers in a chromosomal region are to be tested, several studies suggested the use of generalized linear models.[10–12] Such methods must assume a specific distribution of the trait, given the SNPs, and this assumption does not always hold. Typically, a Bonferroni correction for the *P* value is employed to account for multiple testing. However, this correction does not take into account the dependence of strongly linked marker loci and may lead to overconservative conclusions. This problem worsens when the number of sites increases.

To cope with these difficulties, Zhang et al.[13] suggested a Monte Carlo procedure to evaluate the overall *P* value of the association between the SNP data and the disease: the χ^2 value of each marker is calculated, and the maximum value over all markers, denoted by CC_{max}, is used as the test statistic. The same statistic is calculated for many data sets with the same genotypes and with randomly permuted labels of the case and control individuals. The fraction of permutations for which this value exceeds the original CC_{max} is used as the *P* value. A clear advantage of this test is that no specific distribution function is assumed. Additionally, the test handles multiple testing directly and avoids correction bias. Consequently, it is widely used and, for instance, is implemented in the state-of-the-art software package, Haploview, developed in the HapMap project.

The permutation test can be readily generalized to handle association between haplotypes and the disease—for example, by adding artificial loci of block haplotypes,[14–16] with states corresponding to common haplotypes. Similarly, one can represent loci interactions as artificial loci whose states are the allele combinations.

From the School of Computer Science, Tel Aviv University, Tel Aviv

Received April 27, 2006; accepted June 27, 2006; electronically published July 24, 2006.

Address for correspondence and reprints: Dr. Gad Kimmel, School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. E-mail: kgad@post.tau.ac.il

Am. J. Hum. Genet. 2006;79:481–492. 0002-9297/2006/7903-0011$15.00

Running time is a major obstacle in performing permutation tests. The time complexity of the algorithm is $O(N_S nm)$, where N_S is the number of permutations, n is the number of samples, and m is the number of loci. To search for P values as low as p, at least $1/p$ permutations are needed (see appendix A for details). Therefore, the time complexity can be written as $O(\frac{1}{p}nm)$. For instance, to reach a P value of 10^{-6} in a study that contains 1,000 cases and 1,000 controls with 10,000 loci, $>10^{13}$ basic computer operations are required, with a running time of >30 d on a standard computer. Scaling up to larger studies with $\geqslant$100,000 loci is completely out of reach.

When complex diseases are being studied, SNP interactions should also be considered, and, then, time complexity is an even greater concern. Several statistical studies focus on modeling loci interactions that have little or no marginal effects at each locus.[17–19] Recently, Marchini et al.[20] addressed the issue of designing association studies, given the plausibility of interactions between genetic loci with nonnegligible marginal effects. In all of these studies, the multiple-testing cost of fitting interaction models is much larger than that of the single-locus analysis. Furthermore, the dependency among different tests is higher, so the disadvantage of the conservative Bonferroni correction is exacerbated. For example, when all possible pairwise loci interactions are tested, the number of tests grows quadratically with the number of loci, and applying Bonferroni correction would artificially decrease the test power. In this case, the permutation test is of even higher value. Unfortunately, the running time is linearly correlated with the number of tests, which causes this algorithm to become prohibitively slow, even with a few hundred SNPs.

In this study, we present a faster algorithm for computing accurate P values in disease-association studies. We apply a well-known statistical technique, importance sampling, to considerably decrease the number of sampled permutations. We also use the linkage disequilibrium (LD) decay property of SNPs, to further improve the running time. These two elements are incorporated in a new sampling algorithm called "RAT (Rapid Association Test)." Accounting for decay in LD has already been employed by several studies, for the development of more-efficient and more-accurate algorithms. For example, by using this property, Halperin et al.[21] reported a more accurate and faster method for tagging-SNP selection, and Stephens et al.[22] presented an algorithm that improves the phasing accuracy. To the best of our knowledge, LD decay has not yet been exploited in permutation tests.

In the standard permutation test (SPT), when γ permutations are performed and z successes are obtained, the P value is estimated as z/γ. However, when $z = 0$, we know only that $p \leqslant 1/\gamma$. Therefore, to obtain small P value bounds, one has to expend a lot of computational effort. In contrast, our method provides an estimate of the true P value, with a guaranteed narrow error distribution around it. The distribution gets narrower as the P value decreases, and, therefore, much less effort is needed to achieve accurate, very low P values.

Our method has a running time of $O(n\beta + N_R nc)$, where N_R is the number of permutations drawn by RAT, β is a predefined sampling constant, and c is the upper bound on the distance in SNPs between linked loci. Put differently, any two SNPs that have $\geqslant c$ typed SNPs between them along the chromosome are assumed to be independent. In appendix A, we analyze N_R in terms of the needed accuracy and the true P value.

We compared the performance of our algorithm with that of the regular permutation test, on simulated data generated under the coalescent model with recombination[23] (ms software) and on real human data. For both algorithms, we measured accuracy by the SD of the measured P value. We required an accuracy of 10^{-6} and compared the times to convergence in both algorithms. On realistic-sized data sets, RAT runs 3–5 orders of magnitude faster. For example, it would take ~30 d for the SPT to evaluate 10,000 SNPs in a study with 1,000 cases and 1,000 controls, whereas RAT needs ~2 min. When marker-trait association is tested in simulations with 3,000 SNPs from 1,000 cases and 1,000 controls, it is >5,000 times faster. With 10,000 SNPs from chromosome 1, a speed-up of >20,000 is achieved. With 30,556 simulated SNPs from 5,000 cases and 5,000 controls, it would take 4.62 years for the SPT to achieve the required accuracy, whereas RAT requires 24.3 min. Hence, our method significantly increases the problem-size range for which accurate, meaningful association results are attainable.

This article is organized as follows: in the "Methods" section, we formulate the problem and present the mathematical details of the algorithm. In the "Results" section, results for simulated and real data are presented. The "Discussion" section discusses the significance of the results and future plans. Some mathematical analysis and proofs are deferred to appendix A.

Methods

Problem Formulation

Let n be the number of individuals tested, and let m be the number of markers. The input to our problem is a pair $(M,\mathbf{d})$, where M is an $n \times m$–"markers matrix" and $\mathbf{d}$ is an n-dimensional "disease-status" vector. When haplotype data are used, the dimensions of the matrix may be $2n \times m$. The possible types (alleles) a marker may attain are denoted by $0,1,\ldots,s-1$. Hence, $M(i,j) = k$ if the ith individual has type k in the jth marker. Each component of the disease vector is either 0 (for an unaffected individual) or 1 (for an affected individual). An "association score" $S(\mathbf{d})$ between M and $\mathbf{d}$ is defined below. Let $\pi(\mathbf{d})$ be a permutation of the vector $\mathbf{d}$. The goal is to calculate the P value of $S(\mathbf{d})$—that is, the probability of obtaining an association score $\geqslant S(\mathbf{d})$ under the random model, in which all instances $(M, \pi(\mathbf{d}))$ are equiprobable.

Let $\xi = \sum_{a=1}^{n} \mathbf{d}(a)$ (i.e., ξ is the number of affected individuals). In this article, the set of all possible permutations of the binary vector $\mathbf{d}$ is defined by $\{v|v$ is a binary vector that contains exactly ξ 1s$\}$. In other words, two permutations cannot have the same coordinates set to 1. Following this definition, there are $\binom{n}{\xi}$ pos-

sible permutations of $\mathbf{d}$, instead of $n!$ possibilities by the standard definition of a permutation. Notice that, since all (standard) permutations are equiprobable, our definition for a permutation is equivalent to the standard one from a probabilistic view: for each of the $\binom{n}{\xi}$ permutations with the use of our definition, there are exactly $\xi!(n-\xi)!$ permutations with the use of the standard definition.

For two marker vectors $\mathbf{x}$ and $\mathbf{y}$ of size n, let T denote their contingency table. T is built as follows: $T_{i,j} = |\{k|\mathbf{x}(k) = i, \mathbf{y}(k) = j\}|$. Let T_E be its expected contingency table, assuming the vectors $\mathbf{x}$ and $\mathbf{y}$ are independent—that is, $T_{Ei,j} = \Sigma_a T_{i,a} \Sigma_a T_{a,j} / \Sigma_{a,b} T_{a,b}$. The Pearson χ^2 score of the table T is $S(T) = \Sigma_{i,j} (T_{i,j} - T_{Ei,j})^2 / T_{Ei,j}$. We also use $S(\mathbf{x},\mathbf{y})$ to denote $S(T)$.

The jth column of the matrix M is denoted by $M_{\cdot,j}$. We use the notation $S_j(\mathbf{x})$ for the score $S(M_{\cdot,j},\mathbf{x})$. Hence, $S_j(\mathbf{d})$ is the Pearson score of marker j and the disease vector $\mathbf{d}$. Under the random model described above, the asymptotic distribution of $S_j(\mathbf{d})$ is χ^2, with $s-1$ df.[24] For a vector $\mathbf{x}$, let $S(\mathbf{x}) = \max_j S_j(\mathbf{x})$—that is, the highest Pearson score of any marker in M with the disease vector $\mathbf{x}$. $S(\mathbf{d})$ is called the "association score" for $(M,\mathbf{d})$. We would like to calculate the probability that $S(\mathbf{x}) > S(\mathbf{d})$, where $\mathbf{x}$ is a random permutation of the vector $\mathbf{d}$.

Let $\mathcal{F}$ be the event space of all $\binom{n}{\xi}$ possible permutations of the vector $\mathbf{d}$. The probability measure of $\mathcal{F}$ is defined as $\forall a \in \mathcal{F}$: $\Pr_{\mathcal{F}}(a) = \frac{1}{|\mathcal{F}|}$. We use $f(\cdot)$ to denote $\Pr_{\mathcal{F}}(\cdot)$. Let $\mathcal{H}$ be the subset of $\mathcal{F}$, such that $\mathcal{H} = \{\mathbf{d}_i | \mathbf{d}_i \in \mathcal{F}, S(\mathbf{d}_i) \geqslant S(\mathbf{d})\}$. (Note that throughout we denote, by $\mathbf{d}_i$, the ith permutation and not the ith component of the vector $\mathbf{d}$.) Then, $p = \frac{|\mathcal{H}|}{|\mathcal{F}|}$ is the desired P value. Zhang et al.[13] proposed a Monte Carlo sampling scheme of the space $\mathcal{F}$. This test will be referred to as the "SPT." The running time of this algorithm is $O(nmN_s)$, where N_s is the number of permutations of the standard algorithm. We use p_s to denote the calculated P value of SPT and p_R to denote the calculated P value of our algorithm.

Importance Sampling

We now describe our sampling method. We use the methodology of importance sampling.[25] Informally, in SPT, sampling is done from all possible permutations of the labels of the case and control individuals. This is very computationally intensive, since the number of all possible permutations can be very large. For example, the number of all possible permutations for 1,000 cases and 1,000 controls is $\binom{2,000}{1,000} \approx 10^{600}$. In our method, instead of sampling from this huge space, sampling is done from the space of all "important permutations"—namely, all possible permutations that give larger association scores than the original one. To achieve this goal, we first define this probability space (i.e., define a probability measure for each of these permutations) and then show how to correctly sample from it. This sampling is done in three steps: (1) a column (or a SNP) is sampled, (2) a contingency table is sampled for that column from the set of all possible contingency tables that are induced by this column and whose association score is at least as large as the original one, and (3) an important permutation that is induced by this contingency table is sampled.

We construct an event space $\mathcal{G}$, which contains the same events as $\mathcal{H}$ but with a different probability measure that will be defined below. $\mathcal{G}$ has three important properties: (1) one can efficiently sample from $\mathcal{G}$, (2) the probability of each event in $\mathcal{G}$ can be readily calculated, and, (3) for each $\mathbf{d}_i \in \mathcal{H}$, $\Pr_{\mathcal{G}}(\mathbf{d}_i) > 0$. The probability function over $\mathcal{G}$ is denoted by $g(\cdot)$.

We use N_R to denote the number of permutations drawn by the RAT algorithm. With the use of property 3, if N_R samples are drawn from $\mathcal{G}$ instead of from $\mathcal{F}$, then

$$p = \lim_{N_R \to \infty} \frac{1}{N_R} \sum_{i=1}^{N_R} \frac{f(\mathbf{d}_i)}{g(\mathbf{d}_i)} \ . \tag{1}$$

We now define the probability measure on $\mathcal{G}$. For a permutation $\mathbf{e} \in \mathcal{H}$, let $Q(\mathbf{e}) = |\{j| \ 1 \leqslant j \leqslant m, \ S_j(\mathbf{e}) \geqslant S(\mathbf{d})\}|$. Namely, $Q(\mathbf{e})$ is the number of columns in M whose Pearson score with the disease vector $\mathbf{e}$ is at least $S(\mathbf{d})$. Observe that, since $\mathbf{e} \in \mathcal{H}$, $Q(\mathbf{e}) \geqslant 1$. The probability of $\mathbf{e}$ in $\mathcal{G}$ is defined as:

$$g(\mathbf{e}) = \frac{Q(\mathbf{e})}{\sum_{\mathbf{e} \in \mathcal{H}} Q(\mathbf{e})} \ . \tag{2}$$

Let $\mathcal{T}_j$ be the set of all possible contingency tables that correspond to column j of M and to different permutations of the vector $\mathbf{d}$. The number of different permutations of $\mathbf{d}$ that correspond to a specific contingency table T is denoted by $\mu_j(T)$ and can be calculated directly as follows:

$$\mu_j(T) = \prod_{i=0}^{s-1} \binom{T_{i,0} + T_{i,1}}{T_{i,1}} \ . \tag{3}$$

Let T be a contingency table that fits column j. Define

$$\mu_j(T) = \begin{cases} \mu_j(T) & S(T) \geqslant S(\mathbf{d}) \\ 0 & \text{otherwise} \end{cases} \ .$$

Let $\mathcal{H}_j$ be the set $\mathcal{H}_j = \{\mathbf{d}_i | S_j(\mathbf{d}_i) \geqslant S(\mathbf{d})\}$. Observe that $\mathcal{H} = \cup_{j=1}^m \mathcal{H}_j$. Define $\mathcal{C}_j = \{T \in \mathcal{T}_j | \ S(T) \geqslant S(\mathbf{d})\}$.

The following sampling algorithm from $\mathcal{G}$ will be referred to as the "$\mathcal{G}$-sampler":

1. Sample a column j with probability $\frac{|\mathcal{H}_j|}{\sum_{a=1}^m |\mathcal{H}_a|}$.
2. Sample a contingency table T from $\mathcal{C}_j$ with probability $\frac{\mu_j(T)}{|\mathcal{H}_j|}$.
3. Sample a permutation that fits the contingency table T uniformly—that is, with probability $\frac{1}{\mu_j(T)}$.

Theorem 1: *The probability for a vector* $\mathbf{d}_i$ *to be sampled in the* $\mathcal{G}$*-sampler algorithm is* $g(\mathbf{d}_i)$.

Proof: Let $\mathbf{d}_i \in \mathcal{G}$, and suppose that $Q(\mathbf{d}_i) = q$. Let T be the corresponding contingency table of $\mathbf{d}_i$. With the use of the $\mathcal{G}$-sampler, there is a probability of

$$q \times \frac{|\mathcal{H}_j|}{\sum_{a=1}^m |\mathcal{H}_a|} \times \frac{\mu_j(T)}{|\mathcal{H}_j|} \times \frac{1}{\mu_j(T)} = \frac{q}{\sum_{a=1}^m |\mathcal{H}_a|}$$

to choose an element $\mathbf{d}_i$ from $\mathcal{H}$. Since $\Sigma_{\mathbf{d}_i \in \mathcal{H}} Q(\mathbf{d}_i) = \Sigma_{j=1}^m |\mathcal{H}_j|$, the probability is $\frac{Q(\mathbf{d}_i)}{\Sigma_{\mathbf{d}_i \in \mathcal{H}} Q(\mathbf{d}_i)}$.

Step 3 in the $\mathcal{G}$-sampler can be easily performed, given T. For example, in the case of binary traits, one has to randomly select $T_{0,0}$ out of the controls and $T_{0,1}$ out of the cases. When performing steps 1 and 2, there are two computational challenges: (1) calculating $|\mathcal{H}|$ and (2) sampling a contingency table T from $\mathcal{C}_j$ with probability $\frac{\mu_j(T)}{|\mathcal{H}_j|}$. We present two different schemes for these problems: an exact algorithm and a faster approximation algorithm.

An exact algorithm.—For a column j, we enumerate all $O(n^{s-1})$ possible contingency tables and construct the set $\mathcal{C}_j$. For each table

T, we calculate $\mu_j(T)$ according to formula (3), and $|\mathcal{H}_j|$ is calculated by $|\mathcal{H}_j| = \sum_{T \in \mathcal{C}_j} \mu_j(T)$. The total time complexity of this algorithm is $O(n^{s-1}m + N_R nm)$.

An approximation algorithm.—To calculate $|\mathcal{H}|$, let β be a constant. We randomly sample a set L of β columns and calculate $|\mathcal{H}_j|$ for each of the columns in set L, by using the exact algorithm. $|\mathcal{H}|$ is then approximated by $\frac{m}{\beta} \sum_{\mathcal{H}_j \in L} |\mathcal{H}_j|$. In practice, for the problem sizes we tested, this approach was very accurate when used with $\beta = 100$. The running time of this step is $O(n^{s-1}\beta)$, and the total running time of the algorithm is $O(n^{s-1}\beta + N_R nm)$. In our case, $s = 2$, since there are two possible alleles in each position, so the time complexity becomes $O(n\beta + N_R nm)$.

For sampling a contingency table T from $\mathcal{C}_J$ with probability $\frac{\mu_j(T)}{|\mathcal{H}_j|}$, we use a Metropolis-Hastings sampling algorithm.[26,27] We define a directed graph with nodes corresponding to Markov states and with edges corresponding to transitions between states. Each state represents a specific contingency table T and is denoted by St(T). Let $\pi(T) = \frac{\mu_j(T)}{|\mathcal{H}_j|}$. Our goal is to sample a state St(T) with probability $\pi(T)$. We do this by generating a random walk that has a stationary distribution π[St(T)].

To define the edges in the graph, we first need some definitions. We say that a row is "extreme" if one of its cells has value 0. T is a "boundary table" if it has fewer than two nonextreme rows. A "tweak" to a contingency table is obtained by taking a 2 × 2 submatrix, decreasing by one the elements on one diagonal, and increasing by one the elements on the other diagonal. A tweak is "legal" if the resulting table is nonnegative.

Let $N_g(T)$ be the set of all contingency tables that can be obtained by a legal tweak of T. In addition, if T is a boundary table, then $N_g(T)$ also contains all other possible boundary tables that maintain π[St(T)] > 0. The resulting set $N_g(T)$ constitutes the possible transitions from St(T).

Let $J(T_{old}, T_{new})$ be defined as:

$$J(T_{old}, T_{new}) = \begin{cases} \frac{1}{|N_g(T_{old})|} & T_{new} \in N_g(T_{old}) \\ 0 & \text{otherwise} \end{cases}.$$

The sampling algorithm, which will be called "T-sampler," is as follows:

1. Start with an arbitrary table $T_{old} \in \mathcal{C}_j$.
2. Choose an arbitrary table $T_{new} \in Ng(T_{old})$, and calculate

$$h = \min\left[1, \frac{\pi(T_{new})J(T_{new}, T_{old})}{\pi(T_{old})J(T_{old}, T_{new})}\right].$$

3. With probability h, set $T_{old} = T_{new}$.
4. Return to step 2.

The T-sampler algorithm is stopped after a predefined constant number of steps, denoted by ζ, and outputs the final contingency table T. It is guaranteed that, when ζ is large enough, T is sampled with probability close to $\pi(T)$. The last sentence holds true, since the sampler is irreducible (this is proved in appendix A). The running time of the T-sampler algorithm is bounded by a constant, since ζ is a predefined constant.

Once a permutation $\mathbf{d}_i$ is drawn, calculating $Q(\mathbf{d}_i)$ takes $O(nm)$, so the total running time of the algorithm (applying the $\mathcal{G}$-sampler for N_R permutations) is $O(N_R nm)$. We note that, when n is not too large, the sampling of the contingency table can be done by calculating the probability of all $O(n^{s-1})$ possible contingency tables. This is relevant, in particular, when testing individual SNPs (and, thus, $s = 2$).

Calculating g($\mathbf{d}_i$) *and the* P *value.*—After a random permutation $\mathbf{d}_i$ is drawn from $\mathcal{G}$, $g(\mathbf{d}_i)$ is calculated in the following way: according to equation (2), we need to calculate both $Q(\mathbf{d}_i)$ and $\sum_{\mathbf{d}_i \in \mathcal{H}} Q(\mathbf{d}_i)$. The second term, $\sum_{\mathbf{d}_i \in \mathcal{H}} Q(\mathbf{d}_i)$, equals $\sum_{j=1}^{m} |\mathcal{H}_j|$ and is calculated only once, as a prepossessing step. We denote this value by Γ. The first term is calculated in $O(m)$ time, by going over all columns and counting $Q(\mathbf{d}_i) = |\{j|\ 1 \leq j \leq m,\ S_j(\mathbf{d}_i) \geq S(\mathbf{d})\}|$.

To calculate the P value, define

$$\Phi = \frac{1}{f(\mathbf{d}_i)} = \binom{n}{\xi}.$$

The P value is calculated using equations (1) and (2):

$$p = \lim_{N_R \to \infty} \frac{1}{N_R} \sum_{i=1}^{N_R} \frac{f(\mathbf{d}_i)}{g(\mathbf{d}_i)} = \lim_{N_R \to \infty} \frac{1}{N_R} \sum_{i=1}^{N_R} \frac{\frac{1}{\Phi}}{\frac{1}{\Gamma} Q(\mathbf{d}_i)} \quad (4)$$

$$= \frac{\Gamma}{\Phi} \lim_{N_R \to \infty} \frac{1}{N_R} \sum_{i=1}^{N_R} \frac{1}{Q(\mathbf{d}_i)}.$$

Hence, p_R is calculated by $\frac{\Gamma}{\Phi} \times \frac{1}{N_R} \sum_{i=1}^{N_R} \frac{1}{Q(\mathbf{d}_i)}$.

It follows from equation (4) (with the assumption that Γ was correctly computed) that the only factor that determines the accuracy of the importance sampling is the *variance* of $\frac{1}{Q(\mathbf{e})}$ and not whether it is small or large. The smaller the variance, the better the accuracy. This relationship is discussed theoretically in appendix A. In practice, as described in the "Results" section, the variance of $\frac{1}{Q(\mathbf{e})}$ (or of the calculated P value) was small, though not zero, when real data were used. Intuitively, this can be explained by the limited range of linkage between markers: if the linkage is limited to, at most, c markers, $Q(\mathbf{e})$ will not be much larger than c, and, hence, Var$[\frac{1}{Q(\mathbf{e})}]$ will be bounded.

Using LD Decay to Improve Time Complexity

In this section, we show how to improve time complexity, under assumptions of biological properties of the data. Assume that two SNPs separated by $\geq c$ SNPs along the genome are independent, because of the LD decay along the chromosome. c is called the "linkage upper bound" of the data. Hence, when calculating $Q(\mathbf{d}_i)$ for each permutation $\mathbf{d}_i$, it is unnecessary to go over all m SNPs. Let b_i be the position of the SNP that induces the permutation $\mathbf{d}_i$ that achieves maximum score. Only SNPs within a distance of c—that is, SNPs whose positions are between $b_i - c$ and $b_i + c$—are checked.

The remaining $m - 2c - 1$ SNPs are independent of b_i, so the expected number of columns that give scores $> S(\mathbf{d})$ is $(m - 2c - 1)q$, where q is the probability for a single column to result with a score $> S(\mathbf{d})$. q can be calculated only once at the preprocessing step. Consequently, only $O(cn)$ operations are needed to calculate $Q(\mathbf{d}_i)$, instead of $O(nm)$. Since $O(n\beta)$ operations are needed for the preprocessing phase, the total time complexity is $O(n\beta + N_R nc)$. Observe that, by increasing the value of c, one can improve the accuracy of the procedure at the expense of longer run time.

It should be pointed out that, by using this scheme, the correct expectation of $Q(\mathbf{d}_i)$ is obtained, since the remote markers are

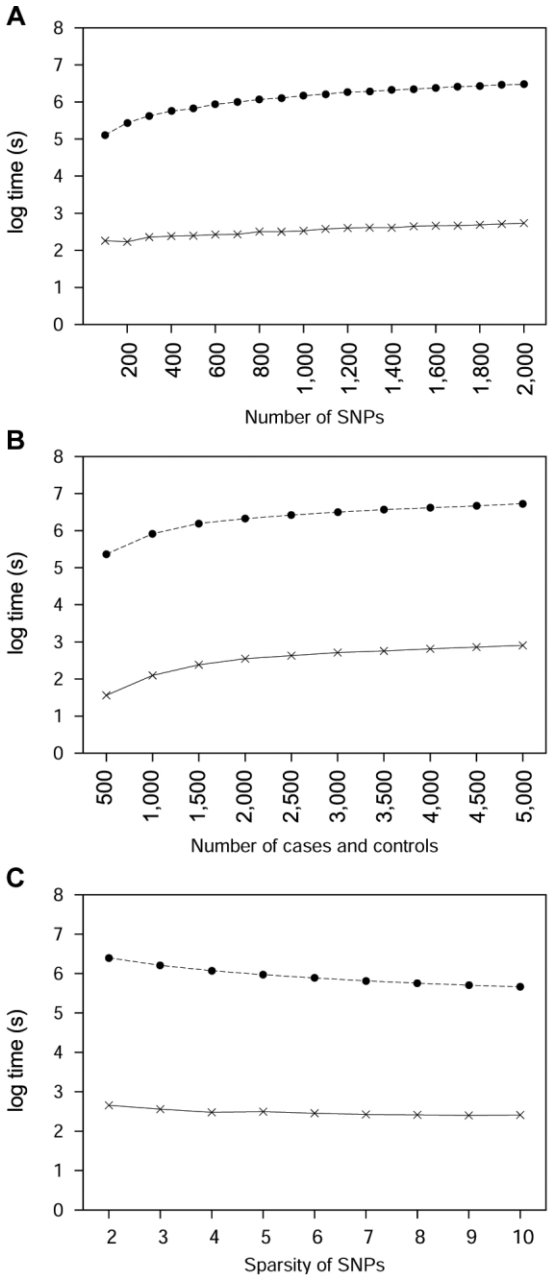


Figure 1. Comparison of running times of the two algorithms that test the disease association of individual SNPs. We present run times of RAT (×) and SPT (*circles*) on simulated data under the coalescent model with recombination. The target *P* value was 10^{-6} in all cases. Running times reflect savings due to importance sampling only, without the additional possible savings due to LD decay. The *Y*-axis gives the logarithm (base 10) of the running time in seconds.

independent of b_i. Theoretically, the remote markers need not necessarily be independent of each other, and, hence, the calculated $Q(\mathbf{e})$ may be biased. In practice, as we shall show (in the "Results" subsection "Real Biological Data"), this is a faithful approximation. Upper bounds on the number of permutations required to search for a *P* value *p* are derived in appendix A, both for SPT and RAT.

Results

We implemented our algorithm in the software package RAT in C++ under LINUX.

Simulated Data

To simulate genotypes, we used Hudson's program that assumes the coalescent process with recombination[23] (ms software). We followed Nordborg et al.,[28] using a mutation rate of 2.5×10^{-8} per nucleotide per generation, a recombination rate of 10^{-8} per pair of nucleotides per generation, and an effective population size of 10,000. Of all the segregating sites, only the ones with minor-allele frequency >5% were defined as SNPs and were used in the rest of the analysis. We used the strategy described elsewhere[29] in choosing the disease marker—that is, we chose a SNP locus as the disease locus if it satisfied two conditions: (1) the frequency of the minor allele is between 0.125 and 0.175, and (2) the relative position of the marker among the SNPs is between 0.45 and 0.55 (i.e., the disease locus is approximately in the middle). The chosen disease SNP was removed from the SNP data set. We then generated case-control data according to a multiplicative disease model. The penetrances of genotypes aa, aA, and AA are λ, $\lambda\gamma$, and $\lambda\gamma^2$, respectively, where λ is the phenocopy rate and γ is the genotype relative risk. As in Zhang et al.,[29] we set $\gamma = 4$ and $\lambda = 0.024$, which corresponds to a disease prevalence of 0.05 and a disease-allele frequency of 0.15. Finally, *N* cases and *N* controls were randomly chosen for each experiment.

We compared the times until convergence in both algorithms, where convergence was declared when the SD of the computed *P* value drops below 10^{-6}. In all our tests, the actual *P* values were $\leqslant 10^{-6}$ (see the "Discussion" section). We set $c = m$ in RAT, so no LD decay is assumed, and the running time is measured using only the importance-sampling component. The approximation algorithm was used in all cases, with the parameter β set to 100. The running times of SPT are very large and, therefore, were extrapolated as follows: since at least 10^6 permutations are needed to achieve an accuracy of 10^{-6} (see appendix A), we measured the running time for 100 permutations, excluding the setup cost (e.g., loading the files and memory allocation), and multiplied by 10^4 to obtain the evaluated running time. We validated this extrapolation by conducting several experiments with 1,000 permutations. The differences between different runs of 1,000 permutations were <1.5%. All runs were done on a Pentium 4 2-GHz machine with 0.5 gigabytes of memory.

In the first setup, we simulated 20,000 haplotypes in a region of 1 Mb. Overall, 3,299 SNPs were generated. We compared the running times when varying three parameters: (1) the number of SNPs (100, 200,…, 2,000), (2) the

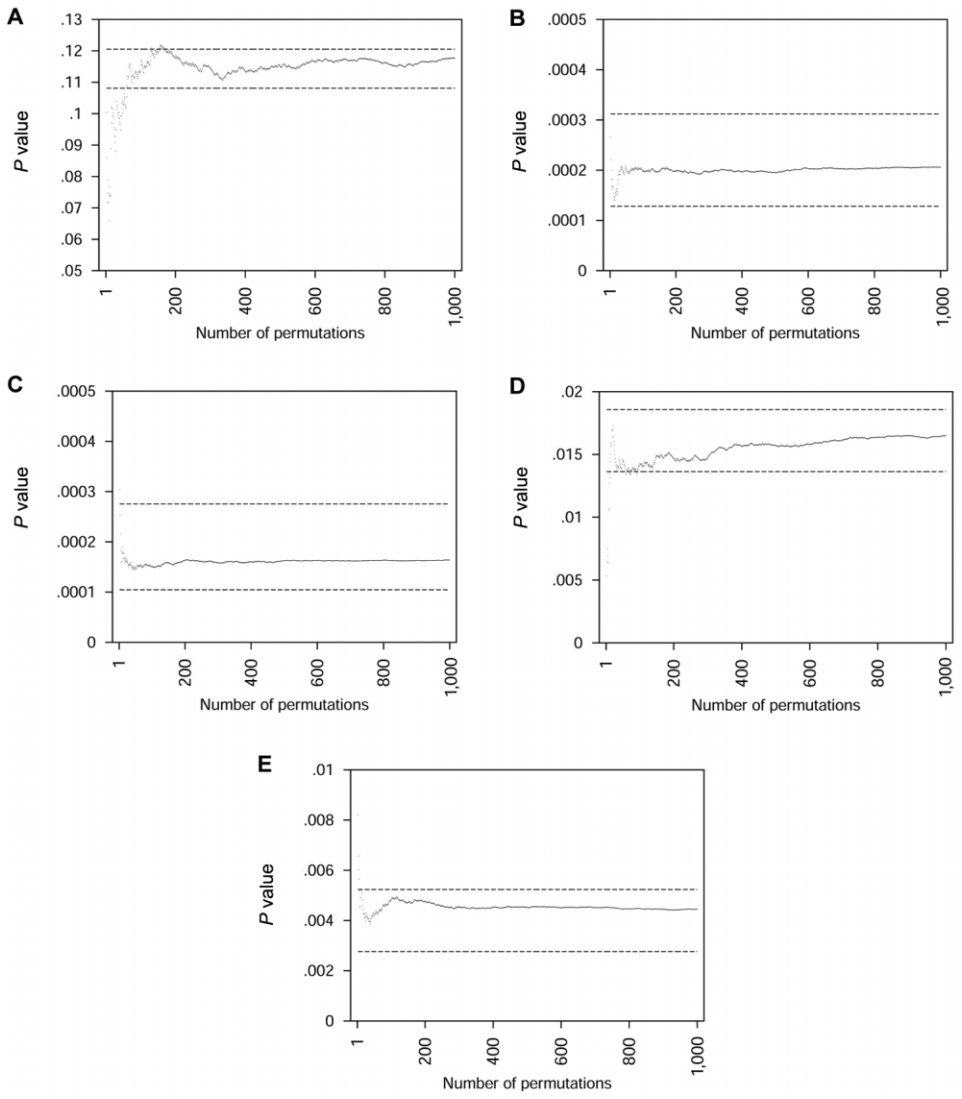


Figure 2. Convergence of RAT to the "true" *P* value. Each of the five figures represents a different experiment with 100 controls and 100 cases of simulated SNPs in a 1-Mb region (~3,000 SNPs), under the coalescent model. SPT *P* value was evaluated by applying 10,000 (*A, D,* and *E*) or 100,000 (*B* and *C*) permutations. The horizontal dashed lines correspond to the 95% CI of SPT *P* value. Each graph corresponds to the RAT *P* value.

number of sampled cases and controls (N = 500, 1,000,…, 5,000), and (3) the SNP density. We chose every ith SNP, where i varies from 1 to 10 (this corresponds to SNP densities between 3,299 and 329 SNPs/Mb). The results are summarized in figure 1. On average, RAT is faster than SPT by a factor of >5,000. For example, it would take ~62 d for SPT to evaluate all 3,299 SNPs for 5,000 cases and 5,000 controls, whereas RAT needs 13 min to obtain the result.

We also tested both algorithms on a very large data set consisting of 10 different regions of 1 Mb each. This data set, generated as described above, contained 5,000 cases and 5,000 controls with 30,556 SNPs. For RAT, we used a linkage upper-bound value of c = 100 kb, on the basis of our observations of LD decay in real biological data (see the "Real Biological Data" subsection). The evaluation of the running time of SPT was performed by the same extrapolation method described above. For this data set, SPT would take 4.62 years to achieve the required accuracy of 10^{-6}, whereas RAT's running time is 24.3 min (i.e., 100,000 times faster).

Since RAT and SPT are both based on sampling, their computed *P* values are distributed around the exact one. Does RAT provide accuracy similar to SPT, in terms of the spread of their distributions? To answer this question, we tested whether RAT converges to the *P* value obtained by SPT. To obtain a reliable estimate of the *P* value obtained by SPT, we used a relatively small number of cases and controls and ran SPT for a large number of permutations. We simulated five different data sets, each with 3,299 SNPs and with 100 cases and 100 controls. We ran SPT for

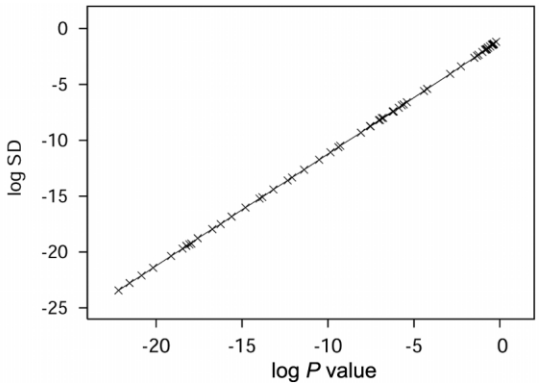


Figure 3. Dependence of accuracy on the *P* value. Data sets were simulated SNPs under the coalescent model with recombination of a 1-Mb region. To obtain different *P* values, we performed the simulations with different numbers of cases and controls ranging from 50 to 500.

10,000 permutations, to calculate 95% CIs of the "true" *P* values. Since a small *P* value was obtained (<.001) in two of these experiments, we increased the number of permutations to 100,000, to improve the accuracy. The results are summarized in figure 2. In all five cases, convergence of the *P* value calculated by RAT to the CI was obtained after <100 permutations.

Our theoretical analysis (see appendix A) shows that, when RAT with a linkage upper bound is used, the accuracy (measured by SD) increases as the *P* value decreases. For evaluation of the actual connection between these two measures, we used simulated data of a 1-Mb region, as described above. We conducted several experiments with different values of *N,* to obtain a range of *P* values. In each experiment, we generated 100 permutations, to estimate the SD. The results are presented in figure 3. For the whole range of *P* values, the SD is, on average, 1/15 of the *P* value.

The complexity analysis of both algorithms (see table A1) shows the theoretical advantage of RAT over SPT when the required *P* value is sufficiently small. At what level of *P* value does RAT have an advantage in practice? To answer this question, we tested both algorithms on data generated by the simulation described above. The data contain ~3,300 SNPs from 5,000 cases and 5,000 controls. To obtain different *P* values, the simulations were performed with different phenocopy rates (λ parameter) of the multiplicative disease model. The results are presented in figure 4. A shorter running time for RAT can be observed, starting from $P = 10^{-2}$.

Real Biological Data

We also tested RAT on HapMap project data. We used SNPs from chromosomes 1–4 of 60 unrelated individuals in the CEPH population. We used the GERBIL algorithm and trios information[15,30] to phase and complete missing SNPs in the data. We amplified the number of samples by adapting the stochastic model of Li and Stephens for haplotype generation.[31] When there are k haplotypes, the $(k + 1)$st haplotype is generated as follows: first, the recombination sites are determined assuming a constant recombination rate along the chromosome (we used 10^{-8} per pair of adjacent nucleotides). Second, for each stretch between two neighboring recombination sites, one of the k haplotypes is chosen, with probability $1/k$. The process is repeated until the required number of haplotypes is achieved. After amplification of the number of samples, cases and controls were chosen as described in the "Simulated Data" subsection.

We wanted to test the effect of the linkage upper bound of the algorithm on real data. Different linkage upper bounds ranging from 1 to 500 kb were checked. For each of the four chromosomes, we used the first 10,000 SNPs (~85 Mb) in 200 cases and 200 controls and applied RAT with varying values of *c*. The results are presented in figure 5. A linkage upper bound of 75 kb (which corresponds to 9 SNPs, on average) appears to be enough to obtain very accurate evaluation of the *P* value.

For a scenario of genomewide association studies that requires typing and checking numerous sites, we used the first 10,000 SNPs of chromosome 1, which span ~84 Mb. We used 1,000 cases and 1,000 controls. For this data set, the running time of RAT for testing disease association of individual SNPs was 361 s (~6 min), compared with the 2.6×10^6 s (~30 d) needed for SPT.

The contribution of the LD decay property is larger when the data set contains more SNPs. To evaluate it, we measured the running times of RAT while using different linkage upper bounds, with 1,000 cases and 1,000 controls

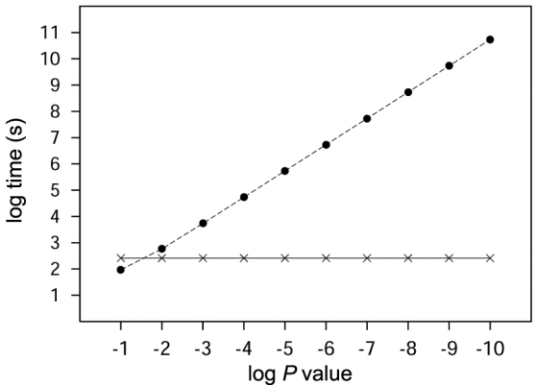


Figure 4. Running times of RAT and SPT at different *P* values. The data sets are simulated data under the coalescent model with recombination of a 1-Mb region (~3,300 SNPs) of 5,000 cases and 5,000 controls. To obtain different *P* values, the simulations were performed with different phenocopy rates (λ parameter) of the multiplicative disease model. × = RAT; circles = SPT. The *Y*-axis shows the logarithm (base 10) of the running time in seconds, and the *X*-axis shows the logarithm (base 10) of the *P* value.

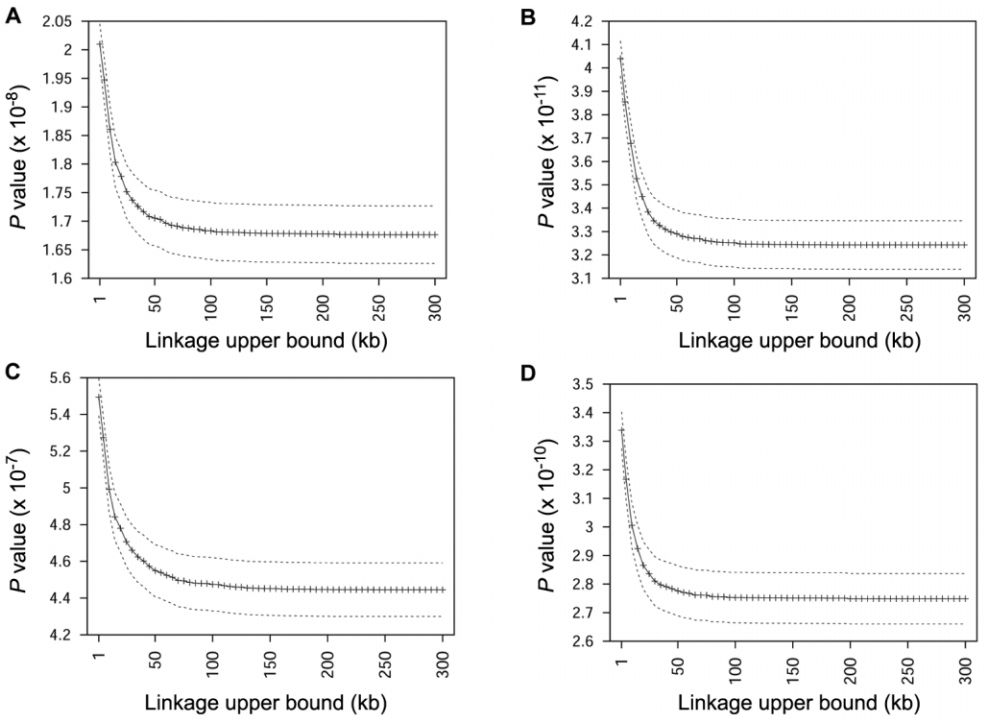


Figure 5. Effect of the linkage upper bound used on the *P* value calculated by RAT. Data sets *A*–*D* are the first 10,000 SNPs in chromosomes 1–4, respectively, of 200 cases and 200 controls, which were amplified from 60 unrelated individuals (the CEPH population from the HapMap project). The dashed lines correspond to the 95% CI of the calculated *P* value. The wide range of *P* values obtained is probably due to the random choice of the disease SNP, the stochastic model of the disease, and chromosomal characteristics.

for the 10,000 SNPs of chromosome 1. The permutation phase of RAT takes 7 s when the linkage upper bound is 1,000 kb and <2 s when it is set to 200 kb (fig. 6). Without the use of this property, 265 s are required (a factor of 132). An additional preprocessing time of 96 s is needed in both cases.

Discussion

The faithful calculation of disease association is becoming more important as more large-scale studies involving thousands of persons and thousands of SNPs are conducted. Testing not only individual SNPs but also haplotypes and loci interactions will further increase this need. Unfortunately, as the size of the data increases, the running time of SPT becomes prohibitively long. In this work, we present an algorithm called "RAT" that dramatically reduces the running time. Our analysis shows that RAT indeed calculates the permutation test *P* value with the same level of accuracy as SPT, but much faster. Our experiments illustrate that the running time of our algorithm is faster by 4–5 orders of magnitude on realistic data sets. This vast difference in the running time enables an evaluation of high-significance association for larger data sets, including evaluations of possible loci interactions and haplotypes.

It is important to emphasize that the advantage of RAT over SPT applies only when the sought *P* value is low. Consider a case-control–labeled data set of SNPs, and suppose there is no association with the disease (e.g., *P* = .5. Using SPT, one can halt the test after very few permutations and conclude that no association exists.

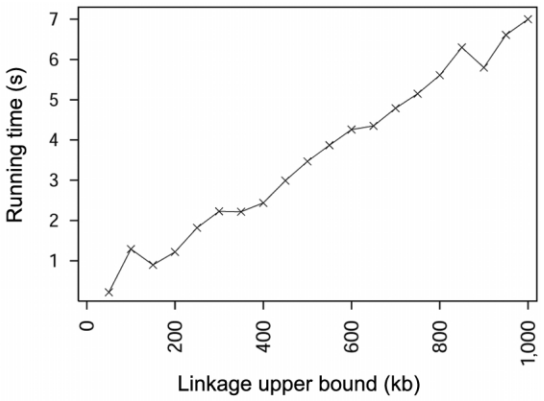


Figure 6. Effect of the LD decay on the speed of RAT. The *Y*-axis shows the time required by the permutation phase of the RAT algorithm. The *X*-axis shows the assumed linkage bound. The data are the first 10,000 SNPs in chromosome 1 of 1,000 cases and 1,000 controls, which were amplified from 60 unrelated individuals (the CEPH population from the HapMap project).

An important reason for achieving high-significance results was presented by Ioannidis et al.,[32] who asked why different studies on the same genetic association sometimes have discrepant results. Their aim was to assess how often large studies arrive at conclusions different from those of smaller studies and whether this situation arises more often when there is a contradiction between the first study and subsequent works. They examined the results of 55 meta-analyses of genetic association and tested whether the magnitude of the genetic effect differs in large, as opposed to smaller, studies. They showed that, in only 16% of the meta-analyses, the genetic association was significant and the same result was obtained independently by several studies, without bias. In a later work, Ioannidis[33] discussed possible reasons for bias in relatively small association studies. He argued that, when many research groups conduct similar association studies, the negative results in studies that do not reach a sufficient significance might never be published. Hence, the scientific literature may be biased. It is hard, or maybe impossible, to correct this multiple-testing effect, since a researcher may not be aware of other groups that study the same question. The solution to this problem is to conduct larger association studies, which, one would hope, would yield lower P values. In that sense, knowing that the P value is below, say, 10^{-2} is not sufficient, and obtaining the most accurate evaluation possible of the P value is crucial.

Our procedure also has an advantage in testing a large population for more than a single disease, where different diseases may be associated with the genotypes at different intensities. Here, one also has to correct for testing multiple diseases. Consider a study that addresses 100 diseases. In such a scenario, a P value of .01 for a specific phenotype obtained by SPT with 100 permutations is not sufficient. In this case, a more accurate evaluation of the significance of association for each of the phenotypes is required. This can be done either by increasing the number of permutations of SPT, which may be time prohibitive, or by using RAT.

Unlike several previous methods, we do not assume any distribution function of the trait, given the SNPs. The random model (adopted from Zhang et al.[13]) assumes only that the cases and controls are sampled independently from a specific population, without any additional requirements about the distribution. However, even this assumption does not always hold. One of the crucial problems in drawing causal inferences from case-control studies is the confounding caused by the population structure. Differences in allele frequencies between cases and controls may be due to systematic differences in ancestry rather than to association of genes with disease.[34–36] In this article, this issue is not addressed, and we intend to study it in the future. We believe that this problem can be solved by incorporating methods for population structure inference[37,38] into RAT.

Using the LD decay property improves the theoretical running time of our method, from $O(n\beta + N_R nm)$ to $O(n\beta + N_R nc)$. This improvement is meaningful when the tested region is much larger than c, the linkage upper bound. In practice, in our experiments, the reduction in the running time due to the importance sampling was much more prominent. We are not aware of a method that can take advantage of LD decay to reduce the running time in SPT. As we show, the importance-sampling approach can readily exploit the LD decay property. Since each drawn permutation in the importance-sampling procedure is induced by a known locus, testing only $2c$ neighboring loci is possible.

RAT can also expedite association analysis when the phenotypic information available for each individual is more complex. For instance, there may be several additional phenotype columns in the input that describe smoking status, sex, age group, or existence of another specific disease. Obviously, with certain factors one cannot use the property of LD decay, but the speed-up due to the importance-sampling algorithm still applies.

We have focused here on the problem of finding association between a genotype matrix and a binary trait (cases and controls), but our algorithm can easily be adapted to also handle continuous traits. A possible score function for a specific column j can be the score used in the ANOVA model, denoted by F_j. The statistic is $\max_j F_j$, and the P value can be calculated by permuting the trait values of individuals, similarly to the binary-traits case. We can use the same methodologies presented here to efficiently calculate the P value.

This work improves the methodologies for the upcoming large-scale association problems. We achieve a dramatic reduction in the time complexity, enabling us to evaluate low-probability (and high-significance) associations with many loci, which was previously time prohibitive. Nevertheless, much more research should be done in this direction. If the number of loci is in the tens of thousands, testing all pairwise interactions is too time consuming, even with our algorithm. If one wants to examine k loci interactions, the running time increases exponentially with k and becomes prohibitive, even for a relatively small number of SNPs. Additional assumptions, such as nonnegligible marginal effects,[20] may help to reduce complexity. We hope that, eventually, combining such assumptions with faster algorithms like RAT may facilitate better analysis of very large association studies.

Acknowledgments

R.S. was supported by a grant from the German-Israeli Fund (grant 237/2005). We thank Jacqui Beckmann (Lausanne), Irit Gat-Viks, Isaac Meilijson (Tel Aviv University), and Jonathan Marchini (Oxford University), for fruitful discussions.

Appendix A
Theoretical Upper Bounds on the Accuracy

We use the SD of the estimated P value in both algorithms, as a measure of accuracy. Obviously, in both al-

gorithms, when more permutations are sampled, the SD is lower. Here, we provide mathematical analysis that relates the number of permutations, the data parameters, and the accuracy.

For SPT, given that N_S permutations are performed, if none of the permutations yields a score $S(\mathbf{d}_i) > S(\mathbf{d})$, we can evaluate the SD by

$$\mathrm{SD}(p_S) = \sqrt{\frac{\frac{1}{N_S}(1-\frac{1}{N_S})}{N_S}} = \Theta\left(\frac{1}{N_S}\right), \tag{A1}$$

which implies that, to achieve an accuracy of ϵ, $\sim 1/\epsilon$ permutations are needed. In particular, when an accuracy equal to the true P value p is desired, $N_S \approx 1/p$.

For the RAT algorithm, let $\Upsilon = |\mathcal{H}|$, and let c_i be $Q(\mathbf{d}_i)$, where $\mathbf{d}_i$ is the ith permutation out of all possible Υ permutations in $\mathcal{H}$. Let Q denote the random variable $Q(e)$, where e is a permutation sampled from $\mathcal{G}$.

The expectation of $1/Q$ is

$$\mathrm{E}\left(\frac{1}{Q}\right) = \sum_{i=1}^{\Upsilon} \frac{c_i}{\Gamma} \times \frac{1}{c_i} = \frac{\Upsilon}{\Gamma},$$

and the variance of the calculated P value is

$$\begin{aligned}
\mathrm{Var}(p_R) &= \mathrm{Var}\left[\frac{\Gamma}{\Phi} \times \frac{1}{N_R}\sum_{i=1}^{N_R}\frac{1}{Q(\mathbf{d}_i)}\right] \qquad \text{(A2)}\\
&= \left(\frac{\Gamma}{\Phi}\right)^2 \mathrm{Var}\left[\frac{1}{N_R}\sum_{i=1}^{N_R}\frac{1}{Q(\mathbf{d}_i)}\right]\\
&= \frac{1}{N_R}\left(\frac{\Gamma}{\Phi}\right)^2\left\{\mathrm{E}\left[\left(\frac{1}{Q}\right)^2\right] - \left[\mathrm{E}\left(\frac{1}{Q}\right)\right]^2\right\}\\
&= \frac{1}{N_R}\left(\frac{\Gamma}{\Phi}\right)^2\left[\frac{1}{\Gamma}\sum_{i=1}^{\Upsilon}\frac{1}{c_i} - \left(\frac{\Upsilon}{\Gamma}\right)^2\right]\\
&= \frac{1}{N_R}\left(\frac{\Gamma}{\Phi}\right)^2\left[\frac{\Upsilon}{\Upsilon}\frac{1}{\Gamma}\sum_{i=1}^{\Upsilon}\frac{1}{c_i} - \left(\frac{\Upsilon}{\Gamma}\right)^2\right]\\
&= \frac{1}{N_R}\left(\frac{\Gamma}{\Phi}\right)^2\frac{\Upsilon}{\Gamma}\left[\frac{1}{\Upsilon}\sum_{i=1}^{\Upsilon}\frac{1}{c_i} - \left(\frac{\Upsilon}{\Gamma}\right)\right].
\end{aligned}$$

Observe that

$$\Upsilon = \Phi p. \tag{A3}$$

Substituting equation (A3) into equation (A2) yields

$$\begin{aligned}
\mathrm{Var}(p_R) &= \frac{1}{N_R}\left(\frac{\Gamma p}{\Upsilon}\right)^2\frac{\Upsilon}{\Gamma}\left[\frac{1}{\Upsilon}\sum_{i=1}^{\Upsilon}\frac{1}{c_i} - \left(\frac{\Upsilon}{\Gamma}\right)\right] \qquad \text{(A4)}\\
&= \frac{1}{N_R}\frac{\Gamma}{\Upsilon}p^2\left[\frac{1}{\Upsilon}\sum_{i=1}^{\Upsilon}\frac{1}{c_i} - \left(\frac{\Upsilon}{\Gamma}\right)\right] \leqslant \frac{p^2}{N_R E(\frac{1}{Q})},
\end{aligned}$$

where the last inequality follows from $0 \leqslant \frac{1}{\Upsilon}\sum_{i=1}^{\Upsilon}\frac{1}{c_i} - (\frac{\Upsilon}{\Gamma}) \leqslant 1$.

Without additional assumptions, the expectation of $1/Q$ is $\geqslant 1/m$. Substituting in equation (A4), we have

$$\mathrm{SD}(p_R) \leqslant \sqrt{\frac{m}{N_R}}p.$$

Hence, to obtain accuracy p, m permutations are needed.

This bound can be improved if we exploit the LD decay property of biological data. Since LD decay is limited to 100 kb (see the "Real Biological Data" subsection in the "Results" section) and the SNP density is, at most, 1:300 bases, $c < 350$ in practice. With the assumption of a linkage upper bound c for a specific locus l, there are, at most, $2c$ loci that may depend on l. For each of the other loci, the probability that its score with a permutation of the vector at locus l is $>S(\mathbf{d})$ is $\leqslant p$. Hence, we can write

$$\mathrm{E}(Q) \leqslant 2c + (m-c)p. \tag{A5}$$

Since $1/[E(1/Q)] \leqslant E(Q)$ always holds true because of Jensen's inequality, when substituting equation (A5) in equation (A4), we get

$$\mathrm{SD}(p_R) \leqslant \sqrt{\frac{2c + (m-c)p}{N_R}}p. \tag{A6}$$

Equation (A6) establishes the connection between the data's parameters and the accuracy. A prominent difference from the accuracy of SPT, described in equation (A1), is the strong dependence on p. Interestingly, when all other data parameters and N_R are fixed, the smaller p is, the more accurate the RAT algorithm is. In other words, as p decreases, the convergence rate of RAT increases.

Arranging equation (A6) differently,

$$N_R \leqslant \frac{2cp^2 + (m-c)p^3}{[\mathrm{SD}(p_R)]^2}.$$

If we set the required accuracy, $\mathrm{SD}(p_R)$, to be p, we have

$$N_R \leqslant 2c + (m-c)p \leqslant 2c + mp.$$

Hence, to search for P values as low as p, the number of required permutations is $<(2c + mp)$. In that case, the time complexity of RAT can be written as $O(n\beta + nc^2 + pcnm)$. The theoretical complexity of the algorithms is summarized in table A1.

Proof of Irreducibility of the T-Sampler Algorithm

We provide a proof that the T-sampler algorithm presented in the subsection "An approximation algorithm" (in the "Methods" section) is irreducible. Consider two tables, T_1 and T_2, from the sample space, such that $\pi(T_1) > 0$ and $\pi(T_2) > 0$. Our goal is to show that there is a path with probability >0 between T_1 and T_2.

If both T_1 and T_2 are boundary tables, then $T_2 \in$

$N_s(T_1)$ and, hence, $J(T_1,T_2) > 0$, and there is positive probability to move from T_1 directly to T_2.

Suppose that, without loss of generality, T_1 is not a boundary table. In that case, there are at least two nonextreme rows a and b in T_1. There are two tables in $N_s(T_1)$ that are created by legal tweaks on the submatrix

$$\begin{pmatrix} T_{a,0} & T_{a,1} \\ T_{b,0} & T_{b,1} \end{pmatrix}.$$

We use T_x to denote the table in which $T_{a,0}$ is increased by one and T_y to denote the other table. The difference in the Pearson score of the tables T_x and T_1 is

$$S(T_x) - S(T_1) = 2\left(\frac{T_{a,0}}{T_{Ea,0}} + \frac{T_{b,1}}{T_{Eb,1}} - \frac{T_{a,1}}{T_{Ea,1}} - \frac{T_{b,0}}{T_{Eb,0}}\right) + \left(\frac{1}{T_{Ea,0}} + \frac{1}{T_{Eb,1}} + \frac{1}{T_{Ea,1}} + \frac{1}{T_{Eb,0}}\right) = \delta + \psi ,$$

where

$$\delta = 2\left(\frac{T_{a,0}}{T_{Ea,0}} + \frac{T_{b,1}}{T_{Eb,1}} - \frac{T_{a,1}}{T_{Ea,1}} - \frac{T_{b,0}}{T_{Eb,0}}\right)$$

and

$$\psi = \left(\frac{1}{T_{Ea,0}} + \frac{1}{T_{Eb,1}} + \frac{1}{T_{Ea,1}} + \frac{1}{T_{Eb,0}}\right).$$

Similarly, $S(T_y) - S(T_1) = -\delta + \psi$.

Since $\psi > 0$, at least one of the expressions $S(T_x) - S(T_1)$ and $S(T_y) - S(T_1)$ is positive. Suppose that, without loss of generality, $\delta > 0$. Then, $S(T_x) > S(T_1) \geq S(d)$, and $\pi(T_x) > 0$. This means that the probability that the sampler moves from T_1 to T_x is positive.

If rows a and b still do not have extreme values in T_x, the exact same procedure can be repeated again and again, until we obtain a table T_1^* in which at least one of these rows has an extreme value.

Suppose α steps were performed, generating a sequence of tables $T_1, T_2, \ldots, T_{\alpha+1} = T_1^*$. A straightforward inductive argument shows that, for all k, $S(T_{k+1}) - S(T_k) = \delta + 2k\psi + \psi > 0$. The last inequality follows by the assumption that $\delta > 0$. Hence, all the tables in the sequence have positive probability. The same argument is repeated with additional nonextreme rows until a boundary table is reached.

Consequently, there is a path with positive probability from any nonboundary table to some boundary table. Since, by definition, transitions between boundary tables have positive probability, it follows that there is a path of positive probability between any two tables with $\pi(T) > 0$, which proves the irreducibility of the sampler.

Table A1. Summary of the Theoretical Time Complexities of SPT and RAT

Algorithm	Prepossessing Phase	Permutations Phase	No. of Permutations[a]	Total Running Time[a]
SPT	...	$\Theta(N_s nm)$	$1/p$	$\Theta(\frac{1}{p} nm)$
RAT (no assumptions)	$O(n\beta)$	$O(N_R nm)$	$\leq m$	$O(n\beta + nm^2)$
RAT (LD decay assumption)	$O(n\beta)$	$O(N_R nc)$	$\leq 2c + mp$	$O(n\beta + nc^2 + pcnm)$

NOTE.—For RAT with LD decay, as the true P value decreases, fewer permutations are needed, and the relative weight of the preprocessing phase increases.

[a] Needed to achieve accuracy p.

Web Resources

URLs for data presented herein are as follows:

Affymetrix GeneChip Human Mapping 500K Array Set, http://www.affymetrix.com/products/arrays/specific/500k.affx
Haploview, http://www.broad.mit.edu/mpg/haploview/
HapMap project, http://www.hapmap.org/
ms, http://home.uchicago.edu/~rhudson1/source.html (software that generates samples under a variety of natural models)
RAT, http://www.cs.tau.ac.il/~rshamir/rat

References

1. Couzin J (2005) To what extent are genetic variation and personal health linked? Science 309:81
2. Martin ER, Lai EH, Gilbert JR, Rogala AR, Afshari AJ, Riley J, Finch KL, Stevens JF, Livak KJ, Slotterbeck BD, Slifer SH, Warren LL, Conneally PM, Schmechel DE, Purvis I, Pericak-Vance MA, Roses AD, Vance JM (2000) SNPing away at complex diseases: analysis of single-nucleotide polymorphisms around *APOE* in Alzheimer's disease. Am J Hum Genet 67:383–394
3. Morris RW, Kaplan NL (2002) On the advantage of haplotype analysis in the presence of multiple disease susceptibility alleles. Genet Epidemiol 23:221–233
4. Risch NJ (2000) Searching for genetic determinants in the new millennium. Nature 405:847–856
5. Shi MM (2001) Enabling large-scale pharmacogenetic studies by high-throughput mutation detection and genotyping technologies. Clin Chem 47:164–172
6. Nickerson DA, Taylor SL, Fullerton SM, Weiss KM, Clark AG, Stengard JH, Salomaa V, Boerwinkle E, Sing CF (2000) Sequence diversity and large-scale typing of SNPs in the human apolipoprotein-E gene. Genome Res 10:1532–1545
7. Tsuchihashi Z, Dracopoli NC (2002) Progress in high throughput SNP genotyping methods. Pharmacogenomics 2:103–110
8. Bonnen PE, Pe'er I, Plenge RM, Salit J, Lowe JK, Shapero MH, Lifton RP, Breslow JL, Daly MJ, Reich DE, Jones KW, Stoffel

M, Altshuler D, Friedman JM (2006) Evaluating potential for whole-genome studies in Kosrae, an isolated population in Micronesia. Nat Genet 38:214–217
9. Olson JM, Wijsman EM (1994) Design and sample size considerations in the detections of linkage disequilibrium with a disease locus. Am J Hum Genet 55:574–580
10. Zaykin DV, Westfall PH, Young SS, Karnoub MA, Wagner MJ, Ehm MG (2002) Testing association of statistically inferred haplotypes with discrete and continuous traits in samples of unrelated individuals. Hum Hered 53:79–91
11. Schaid DJ, Rowland CM, Tines DE, Jacobson RM, Poland GA (2002) Score tests for association between traits and haplotypes when linkage phase is ambiguous. Am J Hum Genet 70:425–434
12. Lin DY (2006) Evaluating statistical significance in two-stage genomewide association studies. Am J Hum Genet 78:505–509
13. Zhang K, Calabrese P, Nordborg M, Sun F (2002) Haplotype block structure and its applications to association studies power and study designs. Am J Hum Genet 71:1386–1394
14. Daly MJ, Rioux JD, Schaffner SF, Hudson TJ, Lander ES (2001) High-resolution haplotype structure in the human genome. Nat Genet 29:229–232
15. Kimmel G, Shamir R (2005) GERBIL: genotype resolution and block identification using likelihood. Proc Natl Acad Sci USA 102:158–162
16. Kimmel G, Shamir R (2005) A block-free hidden Markov model for genotypes and its application to disease association. J Comput Biol 12:1243–1260
17. Hoh J, Ott J (2003) Mathematical multi-locus approaches to localizing complex human trait genes. Nat Rev Genet 4:701–709
18. Culverhouse R, Suarez BK, Lin J, Reich T (2002) A perspective on epistasis: limits of models displaying no main effect. Am J Hum Genet 70:461–471
19. Moore JH, Ritchie MD (2004) The challenges of whole-genome approaches to common diseases. JAMA 291:1642–1643
20. Marchini J, Donnelly P, Cardon LR (2005) Genome-wide strategies for detecting multiple loci that influence complex diseases. Nat Genet 37:413–417
21. Halperin E, Kimmel G, Shamir R (2005) Tag SNP selection in genotype data for maximizing SNP prediction accuracy. Bioinformatics 21:i195–i203
22. Stephens M, Scheet P (2005) Accounting for decay of linkage disequilibrium in haplotype inference and missing-data imputation. Am J Hum Genet 76:449–462
23. Hudson RR (1983) Properties of neutral-allele model with intragenic recombination. Theor Popul Biol 23:183–201
24. Arnold SF (1990) Mathematical statistics. Prentice-Hall, New Jersey, pp 487–501
25. Kalos MH (1986) Monte Carlo methods. John Wiley and Sons, New Jersey
26. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equations of state calculations by fast computing machines. J Chem Phys 21:1087–1092
27. Gilks WR (1994) Markov chain Monte Carlo in practice. Chapman & Hall, United States
28. Nordborg M, Tavare S (2002) Linkage disequilibrium: what history has to tell us. Trends Genet 18:83–90
29. Zhang K, Qin Z, Liu J, Chen T, Waterman MS, Sun F (2004) Haplotype block partitioning and tag SNP selection using genotype data and their applications to association studies. Genome Res 14:908–916
30. Kimmel G, Shamir R (2004) Maximum likelihood resolution of multi-block genotypes. In: Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 04). The Association for Computing Machinery, New York, pp 2–9
31. Li N, Stephens M (2003) Modelling linkage disequilibrium and identifying recombinations hotspots using SNP data. Genetics 165:2213–2233
32. Ioannidis JPA, Trikalinos TA, Ntzani EE, Contopoulos-Ioannidis DG (2003) Genetic association in large versus small studies: an empirical assessment. Lancet 361:567–571
33. Ioannidis JPA (2003) Genetic association: false or true? Trends Mol Med 9:135–138
34. Balding DJ, Bishop M, Canning C (2001) Handbook of statistical genetics: population association. John Wiley and Sons, New Jersey
35. Freedman ML, Reich D, Penney KL, McDonald GJ, Mignault AA, Patterson N, Gabriel SB, Topol EJ, Smoller JW, Pato CN, Pato MT, Petryshen TL, Kolonel LN, Lander ES, Sklar P, Henderson B, Hirschhorn JN, Altshuler D (2004) Assessing the impact of population stratification on genetic association studies. Nat Genet 36:388–393
36. Clayton DG, Walker NM, Smyth DJ, Pask R, Cooper JD, Maier LM, Smink LJ, Lam AC, Ovington NR, Stevens HE, Nutland S, Howson JMM, Faham M, Moorhead M, Jones HB, Falkowski M, Hardenbol P, Willis TD, Todd JA (2005) Population structure, differential bias and genomic control in a large-scale, case-control association study. Nat Genet 37:1243–1246
37. Pritchard JK, Stephens M, Donnelly P (2000) Inference of population structure using multilocus genotype data. Genetics 155:945–959
38. Falush D, Stephens M, Pritchard JK (2003) Inference of population structure using multilocus genotype data: linked loci and correlated allele frequencies. Genetics 164:1567–1587

Chapter 3

Discussion

In the course of this thesis I studied computational problems in modern human genetics, and developed novel algorithms and methods to analyze data sets in this field. In most of our articles four steps of research were done:

1. Exploration of the possible mechanisms for generation of genetics data.

2. Formalization of the these mechanisms in new mathematical models and using these models to formulate new biological problems (e.g., tag SNPs selection and phasing).

3. Development of algorithms that solve these problems for large-scale data sets.

4. Application of these new methods on real biological data sets, and comparing the performance of our methods to recent published ones.

We continuously validated and developed our methods in two channels. First, we utilized public genetics data as they become available to analyze our methods. Second, we established collaborations with leading biological laboratories in Israel and Germany, and conducted joint research that combines our computational methods and their experimental data. Several of these collaborated researches were submitted for publication, and two have already been published [11, 31]. These works are not included in this thesis.

Two performance criteria were considered and compared to previous works. The first and foremost is accuracy. However, when a program becomes imprac-

tically slow as one attempts to use it on larger and larger problems, one should apply the criterion of speed, and test the trade-off between accuracy and speed.

Accuracy is not always easy to evaluate, and the golden standard for comparison is not always clear. In some cases, it is possible to use some additional biological information that is not used by the algorithm, to evaluate the error rate. For instance, in testing phasing algorithms, we used pedigree information to obtain the true solution. In other cases, one has to use simulated data in order to test one's method. A common model in the literature for simulating SNPs and haplotype is the coalescent model with recombination [19, 57]. In our work we tried to use as much real biological data as possible, and used simulated data only when there was no other alternative. For example, the data set of the Hapmap project [60] contains millions of SNPs, but only on a limited number of persons. To generate a larger sample of genotypes based on the Hapmap data, we used an amplification method due to Li and Stephens [32] that generates additional haplotypes based on the available real ones. In this way, the data that were generated were only partially simulated, and were derived from real ones.

Reducing the time complexity of the algorithms was done in two different approaches: First, we exploited the biological knowledge to improve the running time of the algorithm. For example, in order to test associations between SNPs and traits we used the property that SNPs that are in close physical proximity to each other are often correlated, and that lower correlation is often observed between loci that are far apart, to reduce the running time of the algorithm. Second, we used computational methods for developing algorithms that are more efficient.

3.1 Identifying Blocks and Resolving Genotypes

The thesis research was started by redefining and studying the concept of blocks [27, 28]. In simulations, our score leads to more accurate block detection than does the LD-based method of [10]. The latter methods apparently tend to over-partition the data into blocks, as they demand a very stringent criterion between every pair of SNPs in the same block. This criterion is very hard to satisfy as block size increases, and the number of pairwise comparisons grows

quadratically. On the data of [7] we generated a slightly more concise block description than do extant approaches, with a somewhat better fraction of high-LD pairs. We also treated the question of partitioning a set of haplotypes into sub-populations based on their different block structures, and devised a practical heuristic for the problem. On a genotype data set of [10] we were able to identify two sub-populations correctly, in spite of ignoring all heterozygous types. While in some studies the partition into sub-populations is known, others may not have this information, or further, finer partition may be detectable using our algorithm. In our model we implicitly assumed that block boundaries in different sub-populations are independent. In practice, some boundaries may be common due to the common lineage of the sub-populations.

Next, we concentrated in performing phasing in a specific block. We investigated the incomplete perfect phylogeny haplotype problem [21, 25]. The goal is phasing of genotypes into haplotypes, under the perfect phylogeny model, where some of the data are missing. We proved that the problem in its rooted version is NP-complete. We also provided a practical expected polynomial-time algorithm, under a biologically motivated probabilistic data generation model. We applied our algorithm on simulated data, and concluded that the running time and the number of distinct candidate phylogeny solutions are relatively small, under a broad range of biological conditions and parameters, even when the missing data rate is 50%. An accurate treatment for phasing of genotypes with missing entries can therefore be obtained in practice. In addition, due to the small number of phylogenetic solutions observed in simulations, incorporation of additional statistical and combinatorial criteria with our algorithm is feasible.

In order to obtain more accurate phasing, we formulated a stochastic model for generation of genotypes. We presented a model for haplotype resolution and block partitioning as one process, and an algorithm to resolve the model's parameters (GERBIL) [22, 24]. In tests on real data, our algorithm gave more accurate results than two previously published phasing algorithms [8, 12]. Most of our comparisons concentrated on PHASE [52], at that time the leading algorithm for haplotyping. PHASE (version 2.0.2), run with default parameters, was slightly more accurate than GERBIL, but required two orders of magnitude more time. The difference becomes crucial on larger data sets, containing 500 or more genotypes. On such data sets PHASE required several days of computing time, and on 800 genotypes or more it completely failed to provide a solution.

In a subsequent study [23], we further improved the model to reflect the somewhat blocky structure of haplotypes, but also to allow deviations, i.e. intra-block transitions. A first order Markov model is used, without the need to maintain a strict block structure. We have shown how to resolve the model parameters using an EM algorithm. Our new model (HINT) was examined on a broad spectrum of biological data sets. Prediction rate was used as a measure for the validity of the model. The goal in our experiments was to predict a missing causative SNP, given a training set of genotypes. We have shown that HINT gives more accurate results when compared to simpler models. The advantage is not very large, but is statistically significant. An additional interesting byproduct of our analysis is the conclusion that better predictions are made when using haplotypes compared to using genotypes. The strict blocky structure, on the other hand, seems to cause loss of information, and was less accurate in predicting diseases.

It has been argued that haplotype block structures can be helpful for association studies because each haplotype block can be treated as a single locus with several alleles (the block-specific haplotypes) [7]. It was shown that finding the blocks of SNPs is expected to contribute to association studies, by decreasing the number of SNPs needed to be genotyped, with minimal loss of statistical power [62]. A major problem is that, currently, there are different ways of defining and identifying haplotype blocks (for example, [27, 63, 30]). The advantage of blocks is in reducing the number of multiple tests one has to perform when conducting association studies. On the down side, this approach causes some information loss. In HINT we try to take some advantage of the blocks, and by relaxing the model to a "mosaic-like" structure, less information is lost.

After our paper was accepted, we learned of a novel study by Stephens and Scheet, who used a variation of our HINT model [23] to develop a software that can handle very large data sets. Their software, called fastPHASE [53], gave very promising results, especially in terms of the accuracy of imputing missing data.

3.2 Selecting the Most Predictive SNPs

In our study [16] we have defined a novel measure for evaluating the quality of tag SNP selection. The measure we use, *prediction accuracy*, has a very simple and intuitive meaning: It aims to maximize the expected accuracy of predicting untyped SNPs, given the unphased (genotype) information of the tag SNPs. The

prediction itself is done using a simple majority vote. By making an additional natural assumption that SNP values can be determined best based on the values of their nearest tag SNPs on each side, the prediction becomes quite simple, and the optimal selection of tag SNPs can be done in polynomial time.

We presented a method for tag SNPs selection and for SNP prediction based on the genotype values of the tag SNPs. Our selection method, called STAMPA, is unique in its treatment of the prediction part. Most extant methods for tag SNP selection (e.g., [63, 2, 3, 43]) rely on haplotype information. Since such information is often not readily available in real life scenarios, phasing of the genotypes is assumed to have been done separately. One exception is the HapBlock algorithm [64], which selects the tag SNPs based on the genotypes and not on the haplotypes. However, HapBlock selects the tag SNPs in order to maximize diversity of the common haplotypes in blocks, and it is not clear whether that method could be easily extended to a SNP prediction algorithm using genotype data for the tag SNPs.

What is the best measure for selecting tag SNPs? The answer is not clear yet, and also depends on the context. We proposed the expected prediction accuracy, and showed that under reasonable assumptions it yields an efficient and accurate method for selection. Still, other criteria have been proposed. If the ultimate goal is to detect disease association, the power of a selection method may be evaluated using this criterion. Our work shows that if the expected number of errors is of interest, then our algorithms provide more accurate prediction compared to existing algorithms.

3.3 Evaluating the Significance in Association Studies

The last part of the thesis studied one of the important problems in modern human genetics today: evaluating the significance in association studies [26]. The importance of calculating the probability for association will increase as more large-scale studies, involving thousands of persons and thousands of SNPs, are conducted. Testing not only individual SNPs but also haplotypes and loci interactions will further intensify this need. Unfortunately, as the size of the data increases, the running time of the standard permutation test limits it utility. In

this work, we presented an algorithm (RAT) that aims to dramatically reduce the running time. Our analysis shows that RAT indeed calculates the permutation test p-value with the same level of accuracy much faster than the standard permutation test. Our experiments illustrate that the running time of our algorithm is faster by a factor of 10^4-10^5 on realistic data sets. This huge difference in the running time is more than just a theoretical complexity improvement, since it enables an evaluation of association for larger data sets with possible loci interactions and haplotypes. It is important to emphasize that the advantage of RAT over the standard permutation test applies only when the p-value is low, i.e., for highly significant association.

New genotyping technologies like DNA chips [20] are expected to decrease the cost of SNP typing studies substantially. However, the error rate and noise in such high throughput experiments may be high. Moreover, in multi-factorial diseases, such as cancer, high blood pressure or Alzheimer, the disorder is sometimes a composition of several distinct diseases, and the complexity of the phenotype adds to the noise in the data. For instance, a person may be mistakenly labeled as healthy, just because during the research the disease has not became prominent enough to be diagnosed. To tackle this problem, one should increase the number of individuals in the study. Assuming there is an association between the SNPs and the disease, this will decrease the p-value. Moreover, an experimenter may want to sample more in order to increase her belief in the results before continuing to the next step of drug development, which is very expensive. It is thus plausible that studies in the future will contain thousands of people, resulting in relatively low p-values.

Our work is only the first step in improving the methodologies in the upcoming large-scale association problems. We achieve a dramatic reduction of the time complexity, enabling us to evaluate associations with many loci, which were otherwise time-prohibitive. Nevertheless, much more research should be done in this direction. If the number of loci is dozens of thousands, testing all pairwise interactions is too time consuming, even with our algorithm. If one wants to examine the possibility of k-loci interactions, the running time increases exponentially, and becomes prohibitive, even for a relatively small number of SNPs. Additional assumptions, such as non-negligible marginal effects [35], may help to reduce complexity. There is usually a trade-off between adding more assumptions (which implies performing less tests) and the running time. Eventually, we

hope that combining such assumptions with sped-up algorithms like RAT may facilitate better analysis of future association studies.

3.4 Concluding Remarks

The initial goal of our research was to study current central problems in modern human genetics, to formalize them, and to develop computational tools and methods to analyze them. In all of our works, we applied the tools on real data, and compared them to previous methods. An advantage in accuracy or in the running time (or both) was demonstrated in all studies.

We have started by studying combinatorial problems concerning SNPs and genotypes (blocks of high LD and perfect phylogeny). Next, our main research focus was combining algorithms, probability and statistical theory in order to model mutations and recombination processes, for developing improved methods for genotype analysis (haplotypes inference and tag SNPs selection). Finally, we used advanced techniques in probability and statistics to improve the accuracy in evaluation of significance in association studies for very large data sets.

Notably, all of these problems are strongly associated and intertwined. Phasing algorithms rely on partition into blocks to perform resolving. The phasing information is crucial for tag SNPs selection, and for predicting the other SNPs using the tagged ones. Tag SNPs selection is significant for association studies: Selecting less SNPs in a study reduces the typing cost, so that more persons can be typed. This increases the power of the study. The last step, association studies testing, is also strongly related to the definition of blocks, since association is also tested between the haplotypes in the blocks and the disease.

During our research, a better understanding of several of the main computational problems in modern human genetics was achieved. This has led to the development of more accurate and faster tools for analyzing nowadays large-scale genotype data sets. As is often true in research, further improvements in all theoretical and practical aspects of the problems are probably possible and desirable.

Bibliography

[1] V. Bafna, D. Gusfield, G. Lancia and S. Yooseph. Haplotyping as perfect phylogeny: A direct approach. *J. Comput. Biol.* **10(3-4)**, 323–340 (2003).

[2] V. Bafna, B. V. Halldorsson, R. Schwartz, A. Clark and S. Istrail. Haplotypes and informative SNP selection algorithms: Don't block out information. In *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)* (The Association for Computing Machinery), 19–27 (2003).

[3] C. S. Carlson, M. A. Eberle, M. Rieder, Q. Yi, L. Kruglyak and D. A. Nickerson. Selecting a maximally informative set of single-nucleotide polymorphisms for association analysis using linkage disequilibrium. *Am. J. Hum. Genet.* **74**(1), 106–120 (2004).

[4] A. Clark. Inference of haplotypes from PCR-amplified samples of diploid populations. *Mol. Biol. Evol.* **7(2)**, 111–122 (1990).

[5] J. Couzin. To what extent are genetic variation and personal health linked? *Science* **309 (5731)**, 81 (2005).

[6] R. Culverhouse, B. K. Suarez, J. Lin and T. Reich. A perspective on epistasis: limits of models displaying no main effect. *Am. J. Hum. Genet.* **70**, 461–471 (2002).

[7] M. J. Daly, J. D. Rioux, S. F. Schaffner, T. J. Hudson and E. S. Lander. High-resolution haplotype structure in the human genome. *Nat. Genet.* **29(2)**, 229–232 (2001).

[8] E. Eskin, E. Halperin and R. M. Karp. Large scale reconstruction of haplotypes from genotype data. In *Proceedings of the Seventh Annual Inter-*

national Conference on Research in Computational Molecular Biology (RECOMB 03) (The Association for Computing Machinery), 104–113 (2003).

[9] L. Excoffier and M. Slatkin. Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Mol. Biol. Evol.* **12(5)**, 912–917 (1995).

[10] S. B. Gabriel, S. Schaffner, H. Nguyen, J. M. Moore, J. Roy, B. Blumenstiel, J. Higgins, M. DeFelice, A. Lochner, M. Faggart, S. N. Liu-Cordero, C. Rotimi, A. Adeyemo, R. Cooper, R. Ward, E. S. Lander, M. J. Daly and D. Altshuler. The structure of haplotype blocks in the human genome. *Science* **296**, 2225–2229 (2002).

[11] I. Gal, G. Kimmel, R. Gershoni-Baruch, M. Z. Papa, E. Dagan, R. Shamir and E. Friedman. A specific RAD51 haplotype increases breast cancer risk in Jewish non-Ashkenazi high-risk women. *Eur. J. Cancer* **42(8)**, 1129–1134 (2006).

[12] G. Greenspan and D. Geiger. Model-based inference of haplotype block variation. In *Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)* (The Association for Computing Machinery), 131–137 (2003).

[13] D. Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks* **21**, 19–28 (1991).

[14] D. Gusfield. Haplotyping as perfect phylogeny: conceptual framework and efficient solutions. In *Proceedings of the Sixth Annual International Conference on Research in Computational Molecular Biology (RECOMB 02)* (The Association for Computing Machinery), 166–175 (2002).

[15] B. V. Halldorsson, V. Bafna, N. Edwards, R. Lippert, S. Yooseph and S. Istrail. Combinatorial problems arising in SNP. In *Proceedings of the Fourth International Conference on Discrete Mathematics and Theoretical Computer Science (DMTCS 03)* (Springer), volume 2731 of *Lecture Notes in Computer Science*, 26–47. ISBN 3-540-40505-4 (2003).

[16] E. Halperin, G. Kimmel and R. Shamir. Tag SNP selection in genotype data for maximizing snp prediction accuracy. *Bioinformatics* **21(1)**, 195–203 (2005).

[17] G. H. Hardy. Mendelian proportions in a mixed population. *Science* **18**, 49–50 (1908).

[18] J. Hoh and J. Ott. Mathematical multi-locus approaches to localizing complex human trait genes. *Nat. Rev. Genet.* **4**, 701–709 (2003).

[19] R. R. Hudson. Properties of neutral-allele model with intragenic recombination. *Theor. Popul. Biol.* **23**, 183–201 (1983).

[20] G. C. Kennedy, H. Matsuzaki, S. Dong, W. M. Liu, J. Huang, G. Liu, X. Su, M. Cao, W. Chen, J. Zhang, W. Liu, G. Yang, X. Di, T. Ryder, Z. He, U. Surti, M. S. Phillips, M. T. Boyce-Jacino, S. P. Fodor and K. W. Jones. Large-scale genotyping of complex DNA. *Nat. Biotechnol.* **21(10)**, 1233–1237 (2003).

[21] G. Kimmel and R. Shamir. The incomplete perfect phylogeny haplotype problem. In *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes*. 59–70 (2004).

[22] G. Kimmel and R. Shamir. Maximum likelihood resolution of multi-block genotypes. In *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 04)* (The Association for Computing Machinery), 2–9 (2004).

[23] G. Kimmel and R. Shamir. A block-free hidden Markov model for genotypes and its application to disease association. *J. Comput. Biol.* **12(10)**, 1243–1260 (2005).

[24] G. Kimmel and R. Shamir. GERBIL: Genotype resolution and block identification using likelihood. *P. Natl. Acad. Sci. USA* **102**, 158–162 (2005).

[25] G. Kimmel and R. Shamir. The incomplete perfect phylogeny haplotype problem. *J. Bioinform. Comput. Biol.* **3(2)**, 359–384 (2005).

[26] G. Kimmel and R. Shamir. A fast method for computing high significance disease association in large population-based studies. *Am. J. Hum. Genet.* **79**, 481–492 (2006).

[27] G. Kimmel, R. Sharan and R. Shamir. Identifying blocks and subpopulations in noisy SNP data. In *proceedings of the Third Workshop on Algorithms in Bioinformatics (WABI 03)*. 303–319 (2003).

[28] G. Kimmel, R. Sharan and R. Shamir. Computational problems in noisy SNP and haplotype analysis: Block scores, block identification and population stratification. *INFORMS J. Comput.* **16(4)**, 360–370 (2004).

[29] J. F. C. Kingman. The coalescent. *Stochastic Processes and Their Applications* **13**, 235–248 (1982).

[30] M. Koivisto, M. Perola, T. Varilo, W. Hennah, J. Ekelund, M. Lukk, L. Peltonen, E. Ukkonen and H. Mannila. An MDL method for finding haplotype blocks and for estimating the strength of haplotype block boundaries. In *Proceedings of the Pacific Symposium on Biocomputing (PSB 03)* (World Scientific), volume 8, 502–513 (2003).

[31] M. Koren, G. Kimmel, E. Ben-Asher, I. Gal, M. Z. Papa, J. S. Beckmann, D. Lancet, R. Shamir and E. Friedman. ATM haplotypes and breast cancer risk in Jewish high risk women. *Br. J. Cancer* **94**, 1537–1543 (2006).

[32] N. Li and M. Stephens. Modelling linkage disequilibrium and identifying recombinations hotspots using SNP data. *Genetics* **165**, 2213–2233 (2003).

[33] D. Y. Lin. Evaluating statistical significance in two-stage genomewide association studies. *Am. J. Hum. Genet.* **78**, 505–509 (2006).

[34] J. Long, R. C. Williams and M. Urbanek. An EM algorithm and testing strategy for multiple-locus haplotypes. *Am. J. Hum. Genet.* **56(3)**, 799–810 (1995).

[35] J. Marchini, P. Donnelly and L. R. Cardon. Genome-wide strategies for detecting multiple loci that influence complex diseases. *Nat. Genet.* **37**, 413–417 (2005).

[36] E. R. Martin, E. H. Lai, J. R. Gilbert, A. R. Rogala, A. J. Afshari, J. Riley, K. L. Finch, J. F. Stevens, K. J. Livak, B. D. Slotterbeck, S. H. Slifer, L. L. Warren, P. M. Conneally, D. E. Schmechel, I. Purvis, M. A. Pericak-Vance, A. D. Roses and J. M. Vance. SNPing away at complex diseases: analysis of single-nucleotide polymorphisms around APOE in Alzheimer's disease. *Am. J. Hum. Genet.* **67**, 383–394 (2000).

[37] J. H. Moore and M. D. Ritchie. The challenges of whole-genome approaches to common diseases. *J. Am. Med. Assoc.* **291**, 1642–1643 (2004).

[38] R. W. Morris and N. L. Kaplan. On the advantage of haplotype analysis in the presence of multiple disease susceptibility alleles. *Genet. Epidemiol.* **23**, 221–233 (2002).

[39] D. A. Nickerson, S. L. Taylor, S. M. Fullerton, K. M. Weiss, A. G. Clark, J. H. Stengard, V. Salomaa, E. Boerwinkle and C. F. Sing. Sequence diversity and large-scale typing of SNPs in the human apolipoprotein-E gene. *Genome Res.* **10:10**, 1532–1545 (2000).

[40] T. Niu, Z. S. Qin, X. Xu and J. S. Liu. Bayesian haplotype inference for multiple linked single-nucleotide polymorphisms. *Am. J. Hum. Genet.* **70(1)**, 157–169 (2002).

[41] J. M. Olson and E. M. Wijsman. Design and sample size considerations in the detections of linkage disequilibrium with a disease locus. *Am. J. Hum. Genet.* **55**, 574–580 (1994).

[42] N. Patil, A. J. Berno, D. A. Hinds, W. A. Barrett, J. M. Doshi, C. R. Hacker, C. R. Kautzer, D. H. Lee, C. Marjoribanks, D. P. McDonough, B. T. Nguyen, M. C. Norris, J. B. Sheehan, N. Shen, D. Stern, R. P. Stokowski, D. J. Thomas, M. O. Trulson, K. R. Vyas, K. A. Frazer, S. P. Fodor and D. R. Cox. Blocks of limited haplotype diversity revealed by high-resolution scanning of human chromosome 21. *Science* **294**, 1719–1723 (2001).

[43] I. Pe'er, P. Bakker, J. C. Barret, D. Altshuler and M. J. Daly. A branch and bound algorithm for the chromosome tagging problem. In *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes*. 89–98 (2004).

[44] J. Pritchard and M. Przeworski. Linkage disequilibrium in humans: models and data. *Am. J. Hum. Genet.* **69**, 1–14 (2001).

[45] P. Rastas, M. Koivisto, H. Mannila and E. Ukkonen. A hidden Markov technique for haplotype reconstruction. In *LNBI of the Fifth Workshop on Algorithms in Bioinformatics (WABI 2005)*. 140–151 (2005).

[46] N. J. Risch. Searching for genetic determinants in the new millennium. *Nature* **405**, 847–856 (2000).

[47] R. Sachidanandam, D. Weissman, S. C. Schmidt, J. M. Kakol, L. D. Stein, G. Marth, S. Sherry et al. A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms. *Nature* **291**, 1298–2302 (2001).

[48] D. J. Schaid, C. M. Rowland, D. E. Tines, R. M. Jacobson and G. A. Poland. Score tests for association between traits and haplotypes when linkage phase is ambiguous. *Am. J. Hum. Genet.* **70**, 425–434 (2002).

[49] M. M. Shi. Enabling large-scale pharmacogenetic studies by high-throughput mutation detection and genotyping technologies. *Clin. Chem.* **47(2)**, 164–172 (2001).

[50] R. S. Spielman and W. J. Ewens. The TDT and other family based tests for linkage disequilibrium and association. *Am. J. Hum. Genet.* **59**, 983–989 (1996).

[51] R. S. Spielman, R. E. McGinnis and W. J. Ewens. Transmission test for linkage disequilibrium: the Insulin gene region and Insulin-dependent diabetes mellitus (IDDM). *Am. J. Hum. Genet.* **52**, 506–516 (1993).

[52] M. Stephens and P. Donnelly. A comparison of Bayesian methods for haplotype reconstruction from population genotype data. *Am. J. Hum. Genet.* **73(6)**, 1162–1169 (2003).

[53] M. Stephens and P. Scheet. Accounting for decay of linkage disequilibrium in haplotype inference and missing-data imputation. *Am. J. Hum. Genet.* **76**, 449–462 (2005).

[54] M. Stephens, N. J. Smith and P. Donnelly. A new statistical method for haplotype reconstruction from population data. *Am. J. Hum. Genet.* **68(4)**, 978–989 (2001).

[55] Z. Tsuchihashi and N. C. Dracopoli. Progress in high throughput SNP genotyping methods. *Pharmacogenomics J.* **2(2)**, 103–110 (2002).

[56] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt, J. D. Gocayne et al. The sequence of the human genome. *Science* **291**, 1304–1351 (2001).

[57] http://home.uchicago.edu/∼rhudson1/source.html.

[58] http://www.affymetrix.com/products/arrays/specific/500k.affx.

[59] http://www.broad.mit.edu/mpg/haploview.

[60] http://www.hapmap.org.

[61] D. V. Zaykin, P. H. Westfall, S. S. Young, M. A. Karnoub, M. J. Wagner and M. G. Ehm. Testing association of statistically inferred haplotypes with discrete and continues traits in samples of unrelated individuals. *Hum. Hered.* **53**, 79–91 (2002).

[62] K. Zhang, P. Calabrese, M. Nordborg and F. Sun. Haplotype block structure and its applications to association studies power and study designs. *Am. J. Hum. Genet.* **71**, 1386–1394 (2002).

[63] K. Zhang, M. Deng, T. Chen, M. Waterman and F. Sun. A dynamic programming algorithm for haplotype block partitioning. *Proc. Natl. Acad. Sci. USA* **99(11)**, 7335–9 (2002).

[64] K. Zhang, Z. Qin, J. Liu, T. Chen, M. S. Waterman and F. Sun. Haplotype block partitioning and tag SNP selection using genotype data and their applications to association studies. *Genome Res.* **14(5)**, 908–916 (2004).

[65] K. Zhang, F. Sun, M. S. Waterman and T. Chen. Dynamic programming algorithms for haplotype block partitioning: applications to human chromosome 21 haplotype data. In *Proceedings of The Seventh Annual International Conference on Research in Computational Molecular Biology (RECOMB 03)* (The Association for Computing Machinery), 332–340 (2003).

על דגימה לפי חשיבות (importance sampling) ועל כך שיש דעיכה בחוסר שיווי המשקל בתאחיזה לאורך הכרומוזום. השיטה מהירה יותר בכמה סדרי גודל בהשוואה לשיטה של מבחן תמורות רגיל, אשר בה משתמשים לשם הערכת המובהקות הסטטיסטית של אסוציאציה למחלות. כך למשל, על מאגרי נתונים בגודל בינוני עד גדול, האלגוריתם היה מהיר יותר פי 5,000 עד 100,000, והוריד את זמן הריצה משנים למספר בודד של דקות. לכן, שיטה זו מגדילה משמעותית את גודל הבעיות, אשר ניתן לחשב עבורן מובהקות סטטיסטית גבוהה ומדוייקת, בביצוע מחקרי אסוציאציה בין שב"בים למחלות.

5. **A block-free hidden Markov model for genotypes and its application to disease association.**

Gad Kimmel and Ron Shamir.

Published in *Journal of Computational Biology (JCB)* [23].

במאמר זה הוצג מודל הסתברותי חדש ליצירה של גנוטיפים. המודל מציע "פשרה" בין מודל בלוק קשיח לבין חוסר מבנה: הוא משקף את המבנה הבלוקי של הפלוטיפים, אך מאפשר "החלפות" בין הפלוטיפים גם באיזורים שאינם גבולות בין הבלוקים. המודל הוא למעשה הכללה של המודל שהוצג בשני המאמרים הקודמים. הפרמטרים של המודל ההסתברותי מחושבים ע"י אלגוריתם EM. האלגוריתם יושם בתוכנה בשם HINT (Haplotype INference Tool). התוכנה נבדקה על 58 קבצי נתונים ביולוגיים שונים, והשיגה דיוק רב יותר במיפוי אסוציאציה של מחלות מהדיוק שהושג ע"י שלושה מודלים אחרים שהוצגו בעבר.

6. **Tag SNP selection in genotype data for maximizing SNP prediction accuracy.**

Eran Halperin[*], Gad Kimmel[*] and Ron Shamir ([*]equal contribution).

Published in *Bioinformatics* journal supplement for the proceedings of *The 13th Annual International Conference on Intelligent Systems for Molecular Biology (ISMB 2005)* [16].

בעבודה זו הגדרנו מידה טבעית להערכה של חיזוי שב"בים באמצעות שב"בי תיוג, והשתמשנו במידה זו על מנת לפתח אלגוריתם לבחירת שב"בי התיוג. בניגוד לשיטות קודמות לבחירת שב"בי תיוג, שיטת החיזוי של השב"בים האחרים, בהנתן שב"בי התיוג, משתמשת בגנוטיפים ולא בהפלוטיפים. האלגוריתם מאד יעיל, ואינו תלוי בחלוקה לבלוקים לצורך מתן פיתרון. בהפעלת האלגוריתם על מספר רב של קבצי נתונים ביולוגיים נצפה שיפור משמעותי בדיוק לעומת שתי שיטות לבחירת שב"בי תיוג, אשר פורסמו קודם לכן.

7. **A fast method for computing high significance disease association in large population-based studies.**

Gad Kimmel and Ron Shamir.

An invited oral presentation in *The Biology of Genomes meeting*, Cold Spring Harbor Laboratory, 2006. To appear in *American Journal of Human Genetics* [26].

במאמר זה הצגנו אלגוריתם מהיר לשם חישוב המובהקות הסטטיסטית של מחקר אסוציאציה של נתונים מבוססי אוכלוסיה (case - control). בניגוד לשיטות אחרות, לא הנחנו קיום פונקציית ההתפלגות ידועה של המחלה בהנתן הגנוטיפים. האלגוריתם מבוסס

3. **Maximum likelihood resolution of multi-block genotypes.**

Gad Kimmel and Ron Shamir.

Published in *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology (RECOMB 04)* [22].

בעבודה זו פיתחנו מודל הסתברותי ואלגוריתם בכדי להתמודד עם שתי הבעיות של בידול וחלוקה לבלוקים בתהליך אחד משותף. הניתוח מבוסס על מודל הסתברותי ליצירה של בלוקים, כאשר בכל בלוק יש מספר נמוך יחסית של הפלוטיפים נפוצים, ועליהם המודל מאפשר מוטציות ורקומבינציות נדירות. הבעייה הוצגה מתמטית כבעייה של מיקסום פוונקציית נראות, ונפתרה ע"י אלגוריתם Expectation - Maximization (EM). בבדיקת האלגוריתם על נתונים ביולוגיים הושג דיוק טוב יותר משל שני אלגוריתמים אחרים, שפורסמו במאמרים קודמים.

4. **GERBIL: genotype resolution and block identification using likelihood.**

Gad Kimmel and Ron Shamir.

Published in *Proceedings of the National Academy of Sciences of the United States of America (PNAS)* [24].

מאמר זה הינו המשך ישיר של המאמר הקודם. כאן שיפרנו את השיטות המתמטיות לביצוע בידול, שהוצגו במאמר הקודם. הוספנו שיטות חדשות על מנת למדל את הנתונים הביולוגיים בצורה מדוייקת יותר, למשל באמצעות שימוש ב-Minimum Description Length (MDL). הדגש בעבודה זו היה לבדוק את האלגוריתם על מספר רב של מאגרי נתונים ביולוגיים ממקורות שונים, ולהשוות את הדיוק לאלגוריתמי בידול אחרים. האלגוריתם יושם בחבילת תוכנה בשם GERBIL (Genotype Resolution and Block Identification using Likelihood). בדקנו את GERBIL על ארבע קבוצות גדולות של נתונים ביולוגיים ממקורות שונים. השווינו את הדיוק ואת זמן הריצה לאלגוריתם נפוץ בספרות המדעית - PHASE. אלגוריתם זה היה יותר מדוייק מ-GERBIL כאשר הרצנו אותו עם פרמטרי ברירת המחדל שלו, אולם דרש זמן ריצה ארוך יותר בסדרי גודל. כאשר השווינו את שני האלגוריתמים, תוך כדי מתן זמן ריצה זהה, GERBIL היה מדוייק יותר. על נתונים הכוללים מאות גנוטיפים, זמן הריצה הארוך הדרוש לצורך הפעלת האלגוריתם PHASE הופך אותו לבלתי שמיש. לכן, GERBIL הינו בעל יתרון ברור בהפעלה על נתונים ממחקרים גדולים הכוללים מאות גנוטיפים.

עבודה זו מבוססת על שבעה מאמרים, אשר פורסמו בכתבי עת מדעיים והוצגו בכנסים מדעיים. להלן פירוט תקציר המאמרים:

1. **Computational problems in noisy SNP and haplotype analysis: block scores, block identification and population stratification.**

Gad Kimmel, Roded Sharan and Ron Shamir.

Published in *Proceedings of the Workshop on Algorithms in Bioinformatics (WABI 03)* [27] and in *INFORMS Journal on Computing* [28].

בעבודה זו חקרנו מספר בעיות חישוביות שעולות בחלוקה של הפלוטיפים לבלוקים. פונקציית המטרה היתה למזער את מספר ההפלוטיפים בבלוק. הראינו כי הבעייה היא NP-קשה, כאשר יש טעויות מדידה ונתונים חסרים, והצגנו אלגוריתמי קירוב למספר גרסאות של הבעייה. בנוסף, הצגנו אלגוריתם שפותר את הבעייה בהסתברות גבוהה, תחת מודל מתמטי שמבוסס על הנחות בילוגיות. כאשר הרצנו את האלגוריתם על נתוני סימולציה, האלגוריתם שלנו יצר בלוקים דומים מאוד לבלוקים האמיתיים. על נתונים ביולוגיים, הראינו כי חלוקת הבלוקים שלנו יותר מדוייקת מחלוקה המבוצעת ע"י אלגוריתם אחר, המבוסס על חוסר שיווי משקל בתאחיזה.

2. **The incomplete perfect phylogeny haplotype problem.**

Gad Kimmel and Ron Shamir.

Published in *Proceedings of the Second RECOMB Satellite Workshop on Computational Methods for SNPs and Haplotypes* [21] and in *Journal of Bioinformatics and Computational Biology (JBCB)* [25].

במאמר זה הוכחנו, שהבעייה של ביצוע בידול על פי מודל העץ הפילוגנטי המושלם, כאשר חלק מהנתונים חסרים, היא בעייה NP-שלמה. חשיבות הבעייה גדולה, שכן, על פי רוב, מדידת השב"בים בנתונים בילוגיים אינה מלאה עקב סיבות טכניות. לשם פיתרון הבעייה, הגדרנו מודל מתמטי המבוסס על הנחות בילוגיות, ותחת מודל זה, פיתחנו אלגוריתם אשר רץ בתוחלת זמן פולינומיאלית בגודל הקלט. בבדיקות שערכנו על נתוני מסימולציה, הראינו שהאלגוריתם משיג את הפתרון הנכון בזמן ריצה נמוך יחסית.

כאשר מבוצעת מדידה של שב"בי התיוג, קיימת מדידה חלקית בלבד של השב"בים ולפיכך הקרבה הפיזית בין שב"בי התיוג עלולה לא להספיק לצורך ביצוע בידול מדוייק.

מבחני אסוציאציה למחלות

מבחני האסוציאציה באים לשרת את המטרה העיקרית של מדידת שב"בים, שהיא, כאמור, קישורם למחלות נפוצות שונות. ישנם שני סוגים עיקריים של מחקרי אסוציאציה - מחקר מבוסס משפחות, בו החולים והבריאים נבחרים ממשפחה אחת או ממספר משפחות, ומחקר מבוסס אוכלוסייה (case - control study), בו חולים ובריאים נבחרים אקראית מהאוכלוסיה.

במסגרת המחקר שלנו התרכזנו בעיקר בשיטות החישוביות של מחקר מבוסס אוכלוסיה. המבחן הסטטיסטי שמבוצע במחקר מבוסס אוכולוסיה הוא מבחן טיב התאמה עבור SNP בודד [41]. כאשר יש שב"בים רבים, קיימות גישות שונות למדידת הקשר הסטטיסטי בין השב"בים למחלה. הבעייה העיקרית בהערכת המובהקות הסטטיסטית היא שמחד יש מספר רב של שב"בים ולכן יש לתקן להשערות מרובות, ומאידך, השב"בים תלויים זה בזה, כך שהתיקון הקלאסי ע"ש בונפרוני הוא שמרני מדי. לשם פיתרון הבעייה הוצע אלגוריתם [62] אשר מעריך את המובהקות הסטטיסטית ע"י שיטת מונטה - קרלו, שהרעיון העומד מאחוריו הינו ביצוע תמורות על התוויות של החולים והבריאים, תוך כדי חישוב הסטטיסטי. היתרון בשיטה זו הוא שהיא פותרת את בעיית המבחנים המרובים ישירות, ללא צורך בשימוש בשיטות שמרניות מדי. משום דיוקה השיטה נפוצה יחסית. עם זאת, חסרונה הוא אורך זמן הריצה שלה. כאשר יש צורך לבצע הערכה מדוייקת של המובהקות הסטטיסטית של מאות אלפי שב"בים באלפי אנשים, האלגוריתם הנ"ל ידרוש זמן ריצה של חודשים עד שנים על מחשב סטנדרטי.

כיום קיימים מחקרים המודדים שב"בים במאות אנשים, וצפוי כי בעתיד יתקיימו מחקרים שיכללו גם אלפים [39]. גם מספר השב"בים הנמדדים צפוי לעלות עם שיפור הטכנולוגיה, כאשר כבר היום יש בנמצא טכנולוגיה שמסוגלת למדוד 500,000 שב"בים בניסוי אחד [58]. אי לכך, יש חשיבות רבה בפיתוח כלים חישוביים שיוכלו להתמודד עם כמות הנתונים הרבה הזאת.

ההפלוטיפים של אותו אדם נקרא גנוטיפ (genotype). רוב הטכניקות המעבדתיות אינן מספקות מידע הפלוטיפי על כל כרומוזום בנפרד, אלא מידע גנוטיפי, המציג רצף של זוגות לא סדורים של השב"בים מכל עותק של הכרומוזום. למשל, עבור רצף הגנוטיפ הבא: {A,A} {A,C} {C,G} של שלושה שב"בים באדם מסויים, יש שתי אפשרויות לזוג של הפלוטיפים: ACC ו-AAG, או ACG ו-AAC. הואיל ויש ערך רב באיתור ההפלוטיפים עצמם התעורר הצורך בחישובם מתוך הגנוטיפים. תהליך חישוב זה נקרא בידול (phasing). מכיוון שבהעדר מידע נוסף יש פתרונות רבים אפשריים לבידול של גנוטיפ, פותחו שיטות חישוביות רבות אשר מבצעות את תהליך הבידול על אוכלוסיה של אנשים באופן סימולטני.

גישה נפוצה ושימושית היא ביצוע בידול על פי מודל עץ פילוגנטי מושלם (perfect phylogeny tree) [14]. גישה זו מניחה שקיים עץ הפלוטיפים, אשר יצר את ההפלוטיפים בהווה, ואשר מקיים תכונות מתמטיות מסויימות, הנסמכות על הנחות ביולוגיות (כדוגמת ההנחה שמוטציה מתרחשת במיקום מסויים פעם אחת בלבד במהלך ההסטוריה הגנטית). הבעייה החישובית המתעוררת בהקשר זה, היא מציאת ההפלוטיפים שמתאימים לגנוטיפים אשר רצפם נתון, כך שקיים עץ פילוגנטי מושלם שמסביר את קיומם. קיימים אלגוריתמים הנסמכים על מודל זה [8], אשר הראו תוצאות מדוייקות על נתונים ביולוגיים.

גישה נוספת, היא הגדרת מודל הסתברותי ליצירת ההפלוטיפים והגנוטיפים. רוב המחקרים שעושים שימוש בגישה זו, מניחים שיווי משקל Hardy-Weinberg [17] וכן שהאוכלוסיה נוצרה מזיווגים מקריים, כאשר הבידול עצמו מבוצע בתוך כל בלוק בנפרד. לאחרונה הוצעו גישות חדשות [12,52] אשר מבצעות בידול וחלוקה לבלוקים כתהליך אחד.

שב"בי תיוג

העלות הכוללת של מחקר מושפעת ממספר השב"בים שנמדדים בו. לכן, על מנת לחסוך בהוצאות כלכליות, חוקרים מנסים למזער את מספר השב"בים הנמדד בכל אדם, תוך איבוד מינמלי של מידע. תהליך זה מבוצע ע"י בחירה של תת-קבוצה של השב"בים הנקראת שב"בי תיוג (tag SNPs). כך, בזמן עריכת ניסוי, יכול גנטיקאי למדוד את ערכי שב"בי התיוג בלבד. מדידה חלקית זאת מאפשרת חיסכון בעלויות כספיות או הרחבת היקף המחקר לאנשים רבים יותר, תוך כדי העלאת הכוח הסטטיסטי של המחקר וחיזוק מסקנותיו.

מציאת שב"בי תיוג הינה אתגר חישובי מסיבות רבות. ראשית, עולה שאלת הגדרת שב"בי התיוג. שיטות רבות מחלקות את הגנוטיפים לבלוקים ומחפשות שב"בי תיוג בכל בלוק בנפרד. הגם ששיטות אלו מסייעות בפיתרון הבעייה, הרי שהן מובילות לאובדן המידע המקשר בין בלוקים שונים, ולתלות של הפיתרון בחלוקת הבלוקים. שנית, לרוב המידע ההפלוטיפי לא ידוע ונתונים רק הגנוטיפים. כמו שתואר לעיל, קיימים כלים חישוביים אשר מבצעים בידול, אולם כלים אלו מדוייקים רק כאשר יש קירבה רבה בין השב"בים השונים.

תקציר

רקע כללי

השלמת ריצוף הגנום האנושי מאפשרת חיפוש הבדלים בין רצפי ד.נ.א. (DNA) של אנשים שונים לכל אורך הגנום, וקישורם עם מחלות נפוצות כדוגמת סרטן, שבץ מוחי, מחלות לב וכלי דם, סכרת ואסתמה. לצורך השגת מטרה זו, חוקרים מתרכזים במיקומים לאורך הד.נ.א., אשר מראים שונות בתכולת חומצות האמינו שלהם באוכלוסיה. מיקומים אלו נקראים שונות בבסיס בודד (שב"ב) [Single Nucleotide Polymorphisms (SNPs)]. מיליוני שב"בים נמצאו עד היום [47,56] ומספרם הכולל מוערך ב-10 מיליון. מספר מחקרים ראשוניים [36,38,46] הראו כי הסיכון ללקות במחלה נפוצה מושפע מגורמים גנטיים עם שונות גבוהה באוכלוסיה. טרם נאספו מספיק נתונים על מנת להבין עד כמה השערה זו ניתנת להכללה. לכן, זיהוי ומחקר שב"בים הוא כיום מאמץ מרכזי של הקהילה המדעית הבינלאומית [60].

בלוקים של חוסר שיווי משקל בתאחיזה

רצף השב"בים לאורך כרומוזום נקרא הפלוטיפ (haplotype). שב"בים הקרובים זה לזה בגנום, נמצאים לרוב בקורלציה גבוהה. קורלציה זו נמדדת ע"י חוסר שיווי משקל בתאחיזה [(Linkage Disequilibrium (LD)] [44]. חוסר שיווי משקל בתאחיזה נוטה לדעוך עם המרחק, כך שערכים נמוכים יותר שלו נצפים בין מיקומים רחוקים יותר.

מספר מחקרים [10,42] הראו לאחרונה, כי הפלוטיפים נוטים להשמר לאורך מקטעים גנומיים ארוכים. באזורים אלו, חוסר שיווי המשקל בתאחיזה בין השב"בים השונים הוא גבוה יחסית. הסבר מקובל לתופעה זו הוא שרקומבינציה מתרחשת על פני הגנום בעיקר באזורים צרים המכונים "נקודות חמות" ("hot spots"). האזורים בין שתי נקודות חמות נקראים בלוקים (blocks), ומספר ההפלוטיפים השונים בכל בלוק הוא קטן, שכן 70-90% מההפלוטיפים בתוך בלוק שייכים בד"כ לקבוצה קטנה של 2-5 הפלוטיפים נפוצים. השאר נקראים הפלוטיפים נדירים. ממצא זה הינו בעל חשיבות רבה בעיקר לצורך ביצוע מבחני אסוציאציה למחלות, כאשר התקווה היא שניתן יהיה לקשר הפלוטיפים למחלות בצורה חזקה יותר מאשר שב"בים בודדים. מספר מחקרים התרכזו בבעיית החלוקה לבלוקים, והוצעו קריטריונים שונים לחלוקה זו.

גנוטיפים ובידול

בעיית החלוקה לבלוקים קשורה לבעייה אחרת באורגניזמים דיפלואידים (כמו האדם), אשר הינם בעלי שני עותקים כמעט זהים לכל כרומוזום. המידע המאוחד של שני

בשלב הבא, חקרנו את בעיית בחירת שב"בי התיוג. הגדרנו מידה טבעית להערכת מידת הדיוק של קבוצת שב"בי תיוג, והשתמשנו בה לצורך פיתוח שיטה לבחירה של שב"בי תיוג. השווינו את השיטה שפיתחנו לשתי שיטות ידועות בספרות לבחירה של שב"בי תיוג. השיטה שלנו מצאה באופן עיקבי שב"בי תיוג עם יכולת חיזוי טובה יותר מאשר שיטות אחרות.

במחקרנו האחרון, פיתחנו אלגוריתם מהיר יותר מכפי שהיה קיים עד כה לחישוב p-value מדוייק במחקרי אסוציאציה למחלות. פיתחנו שיטה המבוססת על דגימה לפי חשיבות (importance sampling) וניצול תכונת הדעיכה של חוסר שיווי משקל בתאחיזה (linkage disequilibrium), שקיימת לאורך הכרומוזום. האלגוריתם שפיתחנו מהיר ב-5-3 סדרי גודל מאלגוריתם מבחן התמורות הרגיל (standard permutation test), אשר בו משתמשים לשם הערכת המובהקות הסטטיסטית של אסוציאציה למחלות, אך שומר על אותה רמת מהיימנות ודיוק. כך למשל, על מאגרי נתונים בגודל בינוני עד גדול, האלגוריתם היה פי 5,000 עד 100,000 מהיר יותר, והוריד את זמן הריצה משנים למספר בודד של דקות. שיטה זו מגדילה משמעותית את גודל הבעיות בהן ניתן לבצע מבחני אסוציאציה מדוייקים.

תמצית

רוב השונות הגנטית בין בני אדם מתבטאת בנקודות מסויימות בגנום הנקראות שונות בבסיס בודד (שב"ב) [Single Nucleotide Polymorphisms (SNPs)]. חקר השונות הגנטית הוא מרכיב הכרחי במבחני אסוציאציה למחלות. מסיבה זו זיהוי וניתוח של שב"בים מהווה אחת ממטרותיה של הקהילה המדעית הבינלאומית בימים אלו. בעבודה זו, חקרנו חלק מהבעיות החישוביות העיקריות המתעוררות באנליזה של שב"בים. לצורך כך, השתמשנו בטכניקות חישוביות מתורת הגרפים, הסתברות וסטטיסטיקה, ותוך התבססות על עקרונות ביולוגיים, פיתחנו מודלים לבעיות אלו. בשיטות שפיתחנו השתמשנו על מנת לחקור מאגרי נתונים נרחבים של שב"בים אנושי.

ראשית, חקרנו את בעיית החלוקה לבלוקים של הפלוטיפים. בהנתן קבוצה של הפלוטיפים, המטרה היתה למצוא חלוקה לבלוקים, אשר ממזערת את המספר הכולל של ההפלוטיפים בבלוקים. הראינו שהבעייה היא NP-קשה כאשר יש טעויות במדידה או נתונים חסרים, ופיתחנו אלגוריתמי קירוב למספר גרסאות של הבעייה. על נתונים ביולוגיים אמיתיים, הראינו כי מתקבלת חלוקה מדוייקת יותר לבלוקים לעומת שיטות קודמות.

הצעד הבא היה לחקור את בעייה הבידול (phasing). בתחילה ניתחנו את בעיית הבידול בהנחת עץ פילוגנטי מושלם (perfect phylogeny tree). הוכחנו כי הבעייה היא NP-שלמה כאשר חלק מהנתונים חסרים. פיתחנו אלגוריתם, אשר תוחלת זמן הריצה שלו פולינומיאלית בגודל הקלט, תחת הנחת מודל הסתברותי הגיוני מבחינה ביולוגית. בסמולציה, האלגוריתם שפיתחנו מצא את ההפלוטיפים הנכונים במהירות ודיוק גבוהים, גם כאשר שיעור הנתונים החסרים היה גבוה יחסית.

בהמשך, על מנת לקבל תוצאות בידול מדוייקות יותר, פיתחנו אלגוריתם אשר מבצע בידול וחלוקה לבלוקים כתהליך אחד. לצורך כך, הגדרנו מודל הסתברותי לבלוקים בהם הפלוטיפים נוצרים ממספר קטן של הפלוטיפי גרעין, וכן מאפשר קיום מוטציות, ארועי רקומבינציה נדירים וטעויות מדידה. פיתחנו אלגוריתם EM למודל זה, אשר השיג דיוק רב יותר מרוב האלגוריתמים של בידול, שהיו קיימים כאשר פורסמה העבודה. האלגוריתם שפותח אפשר טיפול במאגרי נתונים גדולים, אשר כללו מאות אנשים. זמן הריצה הארוך, אשר נדרש מאלגוריתמים מדוייקים אחרים לביצוע בידול, לא אפשר את השימוש בהם על נתונים באותו היקף. בעבודת המשך, פיתחנו מודל אשר מציע "פשרה" בין מודל בלוקים קשיח למצב בו אין כל הגבלה על המבנה: המודל משקף את מבנה הבלוקים של הפלוטיפים, אך מאפשר "החלפה" בין הפלוטיפים גם באזורים בהם אין גבול בין בלוקים. בנוסף, מאפשר המודל קיום מוטציות וטעויות מדידה. המודל נבדק על מאגרי נתונים רבים של גנוטיפים, והוכיח את עצמו כמדויק יותר בהשוואה לשלושה מודלים אחרים.

אוניברסיטת תל-אביב TEL AVIV UNIVERSITY

הפקולטה למדעים מדויקים ע"ש ריימונד ובברלי סאקלר

בית הספר למדעי המחשב

בעיות חישוביות בגנטיקה

אנושית מודרנית

חיבור לשם קבלת תואר "דוקטור לפילוסופיה"

מאת גד קימל

בהנחייתו של פרופ' רון שמיר

הוגש לסנאט של אוניברסיטת תל - אביב

ספטמבר 2006