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 $\mathrm{St}(T)$. Let $\pi(T) = \frac{\mu_j(T)}{|\mathcal{H}_j|}$. Our goal is to sample a state $\mathrm{St}(T)$ with probability $\pi(T)$. We do this by generating a random walk that has a stationary distribution $\pi[\mathrm{St}(T)]$.

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 $\pi[\mathrm{St}(T)] > 0$. The resulting set $N_g(T)$ constitutes the possible transitions from $\mathrm{St}(T)$.

Let $J(T_{\mathrm{old}}, T_{\mathrm{new}})$ be defined as:

$$J(T_{\mathrm{old}}, T_{\mathrm{new}}) = \begin{cases} \frac{1}{|N_g(T_{\mathrm{old}})|} & T_{\mathrm{new}} \in N_g(T_{\mathrm{old}}) \\ 0 & \text{otherwise} \end{cases} .$$

The sampling algorithm, which will be called "T-sampler," is as follows:

1. Start with an arbitrary table $T_{\mathrm{old}} \in \mathcal{C}_j$.
2. Choose an arbitrary table $T_{\mathrm{new}} \in Ng(T_{\mathrm{old}})$, and calculate

$$h = \min\left[1, \frac{\pi(T_{\mathrm{new}})J(T_{\mathrm{new}}, T_{\mathrm{old}})}{\pi(T_{\mathrm{old}})J(T_{\mathrm{old}}, T_{\mathrm{new}})}\right] .$$

3. With probability h, set $T_{\mathrm{old}} = T_{\mathrm{new}}$.
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 $\mathsf{g}(\boldsymbol{d}_i)$ *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 $Q(\mathbf{d}_i) = |\{j \mid 1 \leq j \leq m,\ S_j(\mathbf{d}_i) \geq S(\mathbf{d})\}|$.

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):

$$p = \lim_{N_R \to \infty} \frac{1}{N_R} \sum_{i=1}^{N_R} \frac{f(\mathbf{d}_i)}{g(\mathbf{d}_i)} = \lim_{N_R \to \infty} \frac{1}{N_R} \sum_{i=1}^{N_R} \frac{\frac{1}{\Phi}}{\frac{1}{\Gamma} Q(\mathbf{d}_i)} \qquad (4)$$

$$= \frac{\Gamma}{\Phi} \lim_{N_R \to \infty} \frac{1}{N_R} \sum_{i=1}^{N_R} \frac{1}{Q(\mathbf{d}_i)} .$$

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, $\mathrm{Var}[\frac{1}{Q(\mathbf{e})}]$ 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

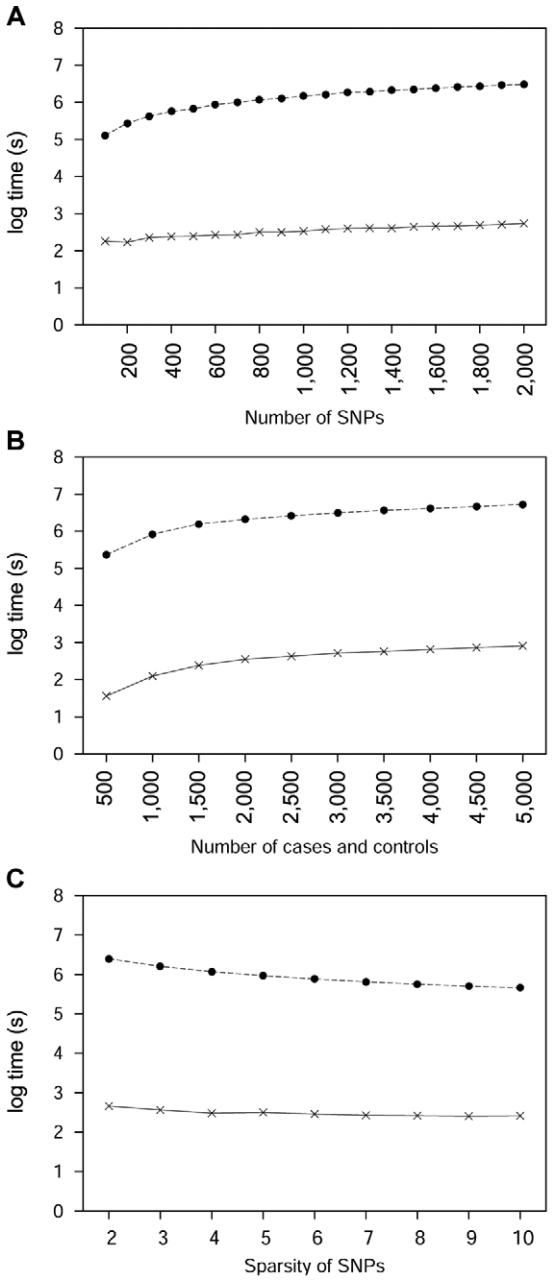

Figure 1. Comparison of running times of the two algorithms that test the disease association of individual SNPs. We present run times of RAT (×) and SPT (*circles*) on simulated data under the coalescent model with recombination. The target *P* value was 10^{-6} in all cases. Running times reflect savings 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

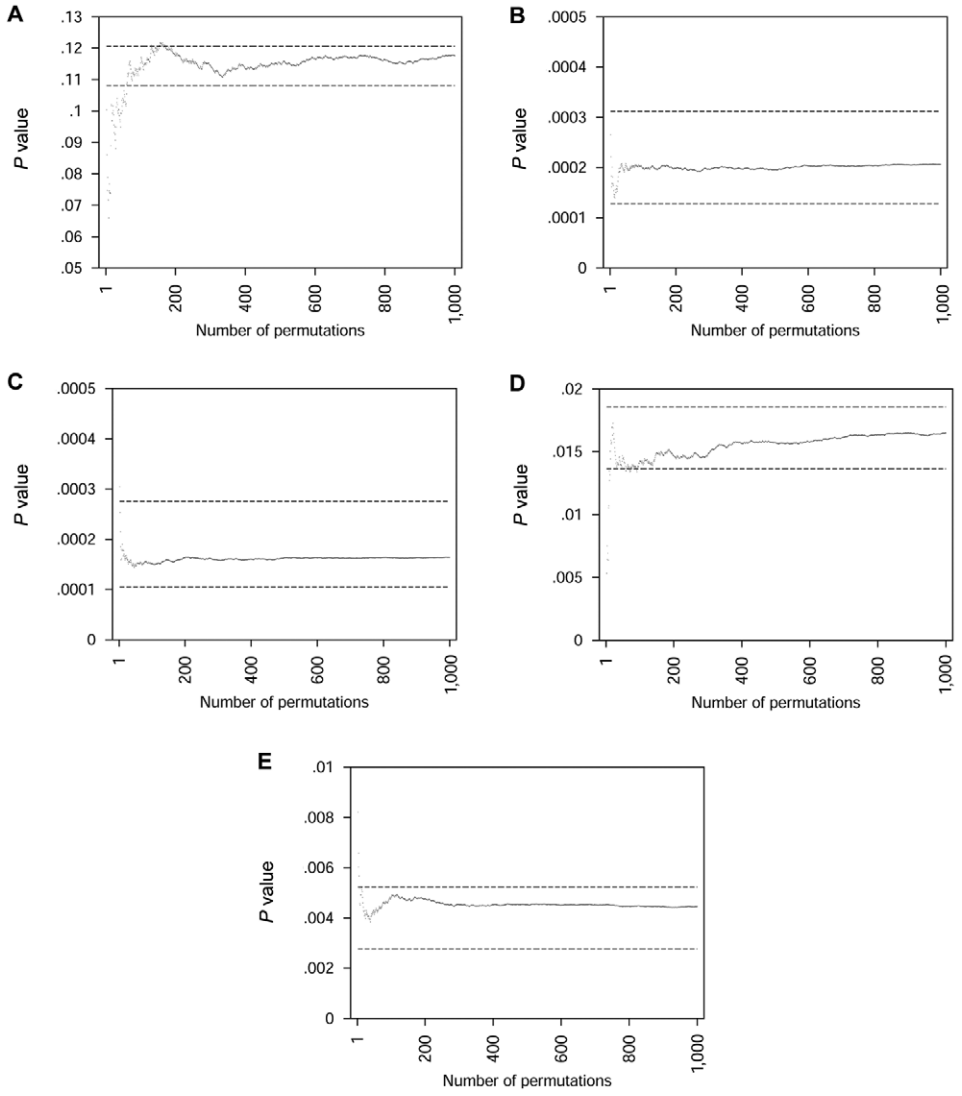

Figure 2. Convergence of RAT to the "true" *P* value. Each of the five figures represents a different experiment with 100 controls and 100 cases of simulated SNPs in a 1-Mb region (~3,000 SNPs), under the coalescent model. SPT *P* value was evaluated by applying 10,000 (*A, D,* and *E*) or 100,000 (*B* 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

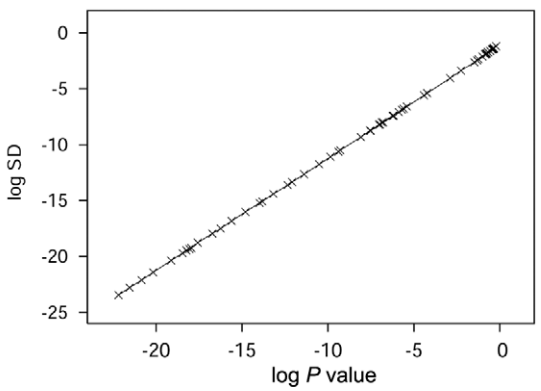

Figure 3. Dependence of accuracy on the *P* value. Data sets were simulated SNPs under the coalescent model with recombination of a 1-Mb region. To obtain different *P* values, we performed the simulations with different numbers of cases and controls ranging from 50 to 500.

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

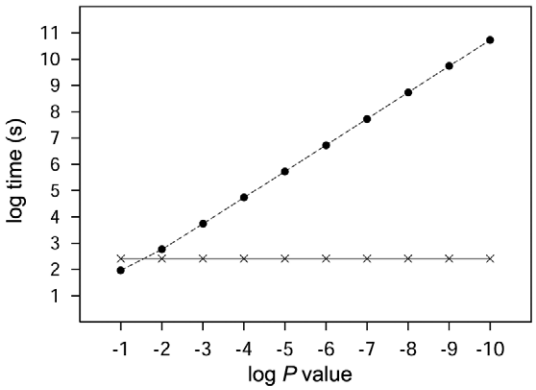

Figure 4. Running times of RAT and SPT at different *P* values. The data sets are simulated data under the coalescent model with recombination of a 1-Mb region (~3,300 SNPs) of 5,000 cases and 5,000 controls. To obtain different *P* values, the simulations were performed with different phenocopy rates (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.

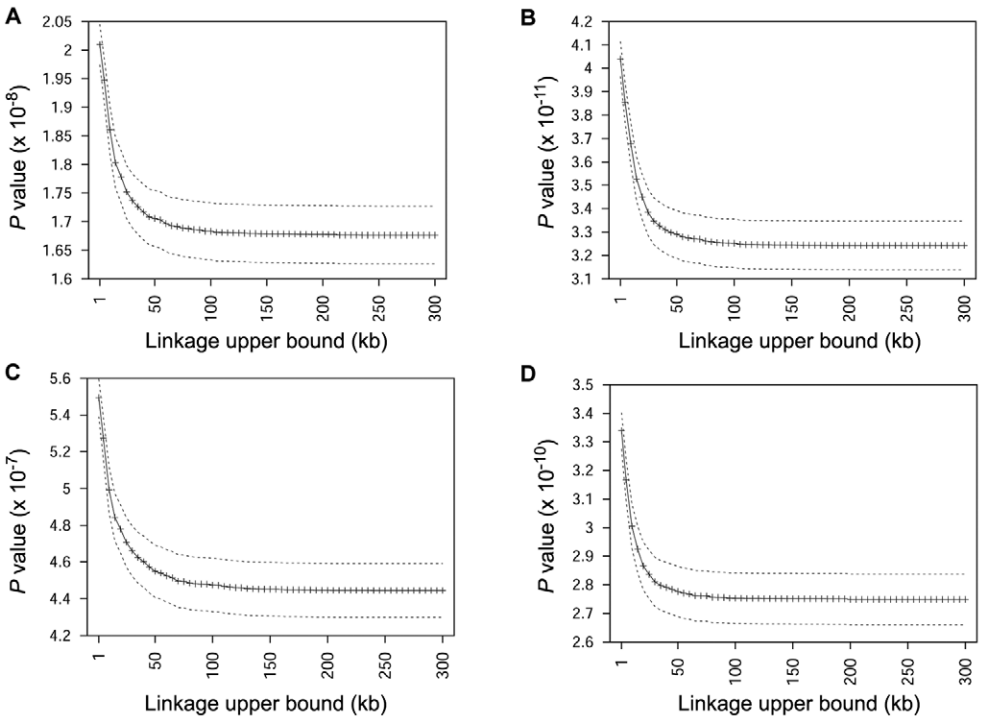

Figure 5. Effect of the linkage upper bound used on the *P* value calculated by RAT. Data sets *A*–*D* are the first 10,000 SNPs in chromosomes 1–4, respectively, of 200 cases and 200 controls, which were amplified from 60 unrelated individuals (the CEPH population from the HapMap project). The dashed 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.

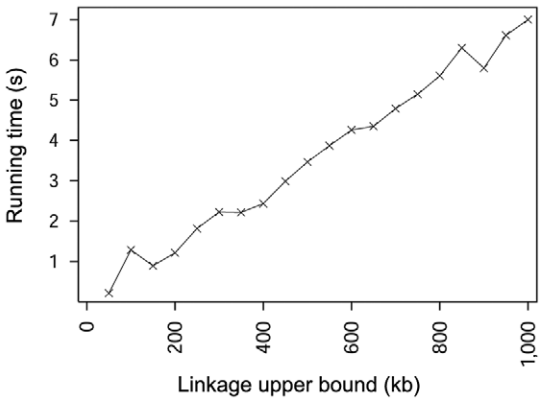

Figure 6. Effect of the LD decay on the speed of RAT. The *Y*-axis shows the time required by the permutation phase of the RAT algorithm. The *X*-axis shows the assumed linkage bound. The data are the first 10,000 SNPs in chromosome 1 of 1,000 cases and 1,000 controls, which were amplified from 60 unrelated 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