FastANOVA: an Efficient Algorithm for Genome-Wide Association Study

Xiang Zhang [1], Fei Zou [2], and Wei Wang [1]
[1]Department of Computer Science, [2]Department of Biostatistics
University of North Carolina at Chapel Hill
[1]xiang@cs.unc.edu, [2]fzou@bios.unc.edu, [1]weiwang@cs.unc.edu

ABSTRACT

Studying the association between quantitative phenotype (such as height or weight) and single nucleotide polymorphisms (SNPs) is an important problem in biology. To understand underlying mechanisms of complex phenotypes, it is often necessary to consider joint genetic effects across multiple SNPs. ANOVA (analysis of variance) test is routinely used in association study. Important findings from studying gene-gene (SNP-pair) interactions are appearing in the literature. However, the number of SNPs can be up to millions. Evaluating joint effects of SNPs is a challenging task even for SNP-pairs. Moreover, with large number of SNPs correlated, permutation procedure is preferred over simple Bonferroni correction for properly controlling family-wise error rate and retaining mapping power, which dramatically increases the computational cost of association study.

In this paper, we study the problem of finding SNP-pairs that have significant associations with a given quantitative phenotype. We propose an efficient algorithm, FastANOVA, for performing ANOVA tests on SNP-pairs in a batch mode, which also supports large permutation test. We derive an upper bound of SNP-pair ANOVA test, which can be expressed as the sum of two terms. The first term is based on single-SNP ANOVA test. The second term is based on the SNPs and independent of any phenotype permutation. Furthermore, SNP-pairs can be organized into groups, each of which shares a common upper bound. This allows for maximum reuse of intermediate computation, efficient upper bound estimation, and effective SNP-pair pruning. Consequently, FastANOVA only needs to perform the ANOVA test on a small number of candidate SNP-pairs without the risk of missing any significant ones. Extensive experiments demonstrate that FastANOVA is orders of magnitude faster than the brute-force implementation of ANOVA tests on all SNP pairs.

Categories and Subject Descriptors: H.2.8 [Database Applications]: Data Mining; J.3 [Life and Medical Sciences]: Biology and Genetics

General Terms: Algorithm, Performance

Keywords: Association study, ANOVA test

KDD'08, August 24–27, 2008, Las Vegas, Nevada, USA.

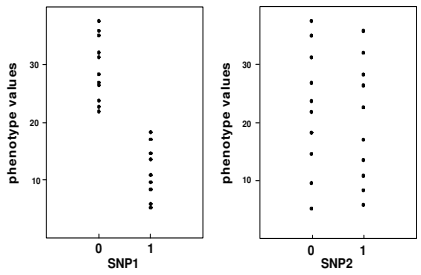
1. INTRODUCTION

Quantitative phenotype association study analyzes genetic variation across a population in order to find the genetic factors underlying continuous phenotypes (such as height or weight). These phenotypes are often complex in the sense that they are likely due to the effects of multiple genes [6, 24]. The most abundant source of genetic variation is represented by single nucleotide polymorphisms (SNPs). A SNP is a DNA sequence variation occurring when a single nucleotide (A, T, G, or C) in the genome differs between individuals of a species. For inbred species, a SNP usually shows variation between only two of the four possible nucleotide types [12], which allows us to represent it by a binary variable. The binary representation of a SNP is also referred to as the *genotype* of the SNP.

Various statistics can be applied to measure the association between SNPs and the phenotypes of interest, among which ANOVA (analysis of variance) test is one of the standard statistic methods and has been routinely used in quantitative phenotype association study [18]. The goal of ANOVA test is to determine whether the group means are significantly different after accounting for the variances within groups. It accomplishes the comparison by decomposing the total variance in the data into within-group variance and between-group variance. If the between-group variance is sufficiently larger than the within-group variance, then the test concludes that there is significant (phenotypic) difference between the groups.

In the application of phenotype-SNP association study, the individuals' phenotype values are grouped by the genotype of a SNP or a subset of SNPs. Figure 1(a) shows an example of strong association between a phenotype and a SNP. 0 and 1 on the x-axis represent the SNP genotype and the y-axis represents the phenotype. Each point in the figure represents an individual. It is clear from the figure that the phenotype values are partitioned into two groups with distinct means, hence indicating a strong association between the phenotype and the SNP. On the other hand, if the genotype of a SNP partitions the phenotype values into groups as shown in Figure 1(b), the phenotype and the SNP are not associated with each other.

Recent advances in high-throughput techniques enable genotyping SNPs in genome-wide scale, resulting in large datasets containing thousands to hundreds of thousands of SNPs [1, 3]. The vast number of SNPs has posed great computational challenge to genome-wide association study. In order to understand the underlying biological mechanisms of complex phenotype, one needs to consider the joint effect of multiple SNPs simultaneously. Although the idea of studying the association between phenotype and multiple SNPs is straightforward, the implementation is nontrivial. For a study with total N SNPs, in order to find the association between n SNPs and the phenotype, a brute-force approach



(a) Strong association (b) No association

Figure 1: Examples of associations between a phenotype and two different SNPs

is to exhaustively enumerate all $\binom{N}{n}$ possible SNP combinations and evaluate their associations with the phenotype. The computational burden imposed by this enormous search space often makes the complete genome-wide association study intractable.

The computational challenge of genome-wide association study is further compounded by another well-known statistical problem – the multiple testing problem [14]. The multiple testing problem can be described as the potential increase in Type I error when statistical tests are performed multiple times. Let α be the Type I error for each independent test. If n independent comparisons are performed, the experimental-wise error α' is given by

$$\alpha' = 1 - (1 - \alpha)^n.$$

For example, when $\alpha = 0.05$ and $n = 20$, $\alpha' = 1 - 0.95^{20} = 0.64$. We have 64% probability to get at least one spurious result. Determining the statistical significance of the association between the phenotype and SNPs is crucial. Bonferroni correction based on the assumption that all n tests are independent is too conservative for the genome-wise association studies since SNPs are often correlated. Alternatively, permutation procedure can be used and much preferred in association studies which automatically takes the correlation structure of SNPs into consideration.

The idea of permutation is to randomly permute the phenotype hundreds to thousands of times. For each permutated phenotype, the association analysis will be repeated. Then the null distribution of the test statistics is estimated and used to assess the statistical significance of the findings from the original phenotype. However, permutation test is very time-consuming since the test procedure needs to be performed in all permutations in order to find the threshold.

Algorithm development to support these large scale analysis is still in its infancy stage. Most existing work focuses on studying the association between the phenotype and SNP-pairs and can only handle a small number of SNPs. Given a pair of SNPs, the phenotype values can be partitioned into at most four groups by the genotype of the SNP-pair, i.e., 00, 01, 10, and 11. Since each SNP has a distinct location on the genome, the association study of a phenotype and SNP-pairs is also called ***two-locus association mapping***. Important findings are appearing in the literature from studying the association between phenotypes and SNP-pairs [21, 22, 27].

Although the standard ANOVA test has been a valuable tool to find association between SNP-pairs and phenotype, it is usually not performed in genome-wide scale. This is due to the fact that the search space of two-locus association mapping in genome-wide scale prohibits an exhaustive search. Suppose that the dataset consists of N SNPs and the number of permutations is K. The total number of ANOVA tests is $KN(N-1)/2$. Given a moderate number of SNPs $N = 10,000$ and number of permutations $K = 1,000$, the number of ANOVA tests is around 5×10^{10}. Therefore, ANOVA test is often reserved for validating a small set of candidates identified by other methods [17, 25].

In this paper, we examine the *computational aspect* of ANOVA test. We present an efficient algorithm, FastANOVA, and show that the standard ANOVA test can be applied in genome-wide scale for two-locus association mapping even when the permutation procedure is needed. Unlike algorithms applying heuristics, FastANOVA is a *complete* algorithm, i.e., it guarantees to find the optimal solution, though it does not explicitly examine all possible SNP-pairs. In fact, a large portion of the SNP-pairs are pruned without the need of performing the tests. FastANOVA establishes an upper bound on the two-locus ANOVA test. The upper bound is the sum of two terms: one based on the ANOVA test between phenotype and a single SNP, and the other based on the pair-wise SNP genotype and the ordered phenotype values. This formulation of the upper bound allows the algorithm to calculate the bound for a large number of SNPs together, which enables fast candidate retrieval. Moreover, the intermediate results for calculating the second term of the upper bound is independent of phenotype permutations. Hence they only need to be computed once and can be reused in all permutations. Applying this bound, FastANOVA is able to identify SNP-pairs with significant ANOVA test values using only a small fraction of the time required by performing ANOVA test on all SNP-pairs. The principles developed in FastANOVA are also applicable to the general case of testing SNP subsets containing more than two SNPs.

2. RELATED WORK

The problem of phenotype-SNP association study has attracted extensive research interests and is an ongoing research area in biology and statistic communities. In this section, we give a briefly review of the related work from a computational point of view. Please refer to [4, 7, 11] for excellent surveys of existing work.

Under the assumption that the number of SNPs is limited, e.g., from tens to a few hundreds, exhaustive algorithms that explicitly enumerate all possible SNP combinations and evaluate their associations with the phenotype have been developed [16, 19]. These methods are not well adapted to genome-wide association study.

To avoid exhaustively enumerating the search space, a common approach is to break the problem into two steps [8, 10]. First, a subset of important SNPs are selected. Second, within the selected subset, the association between SNPs and the phenotypes are searched. These methods are not complete since the SNPs with weak marginal effects may not be selected in the first place. Genetic algorithm [5, 15] has been applied in finding SNP-pairs for quantitative phenotypes. These methods cannot guarantee to find the optimal solution.

Feature selection methods [13] have been proposed to address the problem of finding important SNPs. In feature selection, the selected feature subset usually contains features that have low correlation with each other but have strong correlation with the target feature. In the application of selecting SNPs, the goal is to select a subset of SNPs that can be used as proxies for all SNPs in the genome [9, 23]. The selected SNPs can then be used as the input SNPs in the association study. Apparently, these methods are also not complete.

3. TWO-LOCUS ANOVA TEST

Let $\{X_1, X_2, \cdots, X_N\}$ be the set of SNPs of M individuals ($X_i \in \{0, 1\}, 1 \leq i \leq N$) and $Y = \{y_1, y_2, \cdots, y_M\}$ be the

$X_i = 1$	$X_i = 0$
group A	group B

(a) Grouping of Y by X_i

	$X_i = 1$	$X_i = 0$
$X_j = 1$	group a_1	group b_1
$X_j = 0$	group a_2	group b_2

(b) Grouping of Y by X_iX_j

Table 1: Possible groupings of phenotype values by the genotypes of X_i and (X_iX_j)

quantitative phenotype of interest, where y_m $(1 \leq m \leq M)$ is the phenotype value of individual m.

For any SNP X_i $(1 \leq i \leq N)$, we represent the F-statistic from the ANOVA test of X_i and Y as $F(X_i, Y)$. For any SNP-pair (X_iX_j), we represent the F-statistic from the ANOVA test of (X_iX_j) and Y as $F(X_iX_j, Y)$.

The basic idea of ANOVA test is to partition the total sum of squared deviations SS_T into between-group sum of squared deviations SS_B and within-group sum of squared deviations SS_W:

$$SS_T = SS_B + SS_W.$$

In our application of two-locus association study, Table 1(a) and Table 1(b) show the possible groupings of phenotype values by the genotypes of X_i and (X_iX_j) respectively.

Let A, B, a_1, a_2, b_1, b_2 represent the groups as indicated in Table 1(a) and Table 1(b). We use $SS_B(X_i, Y)$ and $SS_B(X_iX_j, Y)$ to distinct the one locus (i.e., single-SNP) and two locus (i.e., SNP-pair) analyses. Specifically, we have

$$SS_T(X_i, Y) = SS_B(X_i, Y) + SS_W(X_i, Y),$$

$$SS_T(X_iX_j, Y) = SS_B(X_iX_j, Y) + SS_W(X_iX_j, Y).$$

The F-statistics for ANOVA tests on X_i and (X_iX_j) are:

$$F(X_i, Y) = \frac{M-2}{2-1} \times \frac{SS_B(X_i, Y)}{SS_T(X_i, Y) - SS_B(X_i, Y)}, \quad (1)$$

$$F(X_iX_j, Y) = \frac{M-g}{g-1} \times \frac{SS_B(X_iX_j, Y)}{SS_T(X_iX_j, Y) - SS_B(X_iX_j, Y)}, \quad (2)$$

where g in Equation (2) is the number of groups that the genotype of (X_iX_j) partitions the individuals into. Possible values of g are 3 or 4, assuming all SNPs are distinct: If none of groups A, B, a_1, a_2, b_1, b_2 is empty, then $g = 4$. If one of them is empty, then $g = 3$.

Let $T = \sum_{y_m \in Y} y_m$ be the sum of all phenotype values. The total sum of squared deviations does not depend on the groupings of individuals:

$$SS_T(X_i, Y) = SS_T(X_iX_j, Y) = \sum_{y_m \in Y} y_m^2 - \frac{T^2}{M}.$$

Let $T_{group} = \sum_{y_m \in group} y_m$ be the sum of phenotype values in a specific group, and n_{group} be the number of individuals in that group. $SS_B(X_i, Y)$ and $SS_B(X_iX_j, Y)$ can be calculated as follows:

$$SS_B(X_i, Y) = \frac{T_A^2}{n_A} + \frac{T_B^2}{n_B} - \frac{T^2}{M},$$

$$SS_B(X_iX_j, Y) = \frac{T_{a_1}^2}{n_{a_1}} + \frac{T_{a_2}^2}{n_{a_2}} + \frac{T_{b_1}^2}{n_{b_1}} + \frac{T_{b_2}^2}{n_{b_2}} - \frac{T^2}{M}.$$

Note that for any group of A, B, a_1, a_2, b_1, b_2, if $n_{group} = 0$, then $\frac{T_{group}^2}{n_{group}}$ is defined to be 0.

The two-locus association mapping with permutation test is typically conducted in the following way [18].

First, for every SNP-pair (X_iX_j) $(1 \leq i < j \leq N)$, the ANOVA test is performed and $F(X_iX_j, Y)$ is recorded.

Second, a permutation test is performed to get a reference distribution in order to assess the statistical significance of previous findings. More specifically, a permutation Y_k of Y is generated by sampling the phenotype Y without replacement. In other words, phenotype values are randomly assigned to individuals in the dataset with no single phenotype value being assigned to more than one individual. Let $Y' = \{Y_1, Y_2, \cdots, Y_K\}$ be the set of K permutations of Y. For each permutation $Y_k \in Y'$, let F_{Y_k} represent the maximum F-statistic value of all SNP-pairs, i.e.,

$$F_{Y_k} = \max\{F(X_iX_j, Y_k) | 1 \leq i < j \leq N\}.$$

The distribution of $\{F_{Y_k} | Y_k \in Y'\}$ is then used as the reference distribution for assessing the statistical significance of $F(X_iX_j, Y)$ values found using the original phenotype Y: Given a Type I error threshold α, the *critical value* F_α is the αK-th largest value in $\{F_{Y_k} | Y_k \in Y'\}$. For example, suppose that $\alpha = 0.01$ and $K = 1000$, then F_α is the 10th largest value in $\{F_{Y_k} | Y_k \in Y'\}$. The SNP-pair (X_iX_j) whose F-statistic value $F(X_iX_j, Y) \geq F_\alpha$ is considered as significant at α.

Two computational problems need to be solved in this procedure. The first one is to find the critical value F_α for a given Type I error threshold α. The second one is to find all SNP-pairs (X_iX_j) whose F-statistics are greater than F_α. We formalize these two problems as follows.

Problem (1): Given the Type I error threshold α, find the critical value F_α, which is the αK-th largest value in $\{F_{Y_k} | Y_k \in Y'\}$.

Problem (2): Given the threshold F_α, find all SNP-pairs (X_iX_j) such that $F(X_iX_j, Y) \geq F_\alpha$.

A brute force approach to these two problems is to enumerate all SNP-pairs and find their F-statistics. In Problem (1), for each permutation $Y_k \in Y$, all SNP-pairs need to be enumerated in order to find the maximum value F_{Y_k}. In Problem (2), all SNP-pairs need to be enumerated to see if their test values are above the threshold F_α. Computationally, Problem (1) is more challenging, since the permutation number K can range form hundreds to thousands, which means the running time of finding the critical value F_α can be hundreds to thousands times longer than the running time of finding the significant SNP-pairs in Problem (2) using a brute-force search.

In the reminder of the paper, we first derive an upper bound on two-locus ANOVA test value and discuss how this upper bound enables an efficient ANOVA testing for a single phenotype. Then we show how this approach can be easily extended to handle the permutation procedure.

4. THE UPPER BOUND

4.1 Updating F-Statistic

Since the total sum of squared deviations does not change, from the calculation of $F(X_i, Y)$ and $F(X_iX_j, Y)$ (Equations (1) and (2)), we know that the relationship between these two tests only depends on the relationship between $SS_B(X_i, Y)$ and $SS_B(X_iX_j, Y)$. Next we show that $SS_B(X_iX_j, Y)$ can be updated from $SS_B(X_i, Y)$.

For groups A, a_1 and a_2, let

$$\begin{aligned}\Delta A &= \frac{T_{a_1}^2}{n_{a_1}} + \frac{T_{a_2}^2}{n_{a_2}} - \frac{T_A^2}{n_A} \\ &= \frac{n_{a_2}T_{a_1}^2 + n_{a_1}T_{a_2}^2}{n_{a_1}n_{a_2}} - \frac{(T_{a_1}+T_{a_2})^2}{n_{a_1}+n_{a_2}} \\ &= \frac{(n_{a_2}T_{a_1} - n_{a_1}T_{a_2})^2}{n_{a_1}n_{a_2}n_A} \\ &= \frac{(n_A T_{a_1} - n_{a_1}T_A)^2}{n_{a_1}(n_A - n_{a_1})n_A}.\end{aligned}$$

Similarly, we have

$$\Delta B = \frac{T_{b_1}^2}{n_{b_1}} + \frac{T_{b_2}^2}{n_{b_2}} - \frac{T_B^2}{n_B} = \frac{(n_B T_{b_1} - n_{b_1}T_B)^2}{n_{b_1}(n_B - n_{b_1})n_B}.$$

Thus, $SS_B(X_iX_j, Y)$ can be updated using $SS_B(X_i, Y)$:

$$SS_B(X_iX_j, Y) = SS_B(X_i, Y) + \Delta A + \Delta B. \quad (3)$$

Note that if any one of $\{n_{a_1}, n_{a_2}, n_A\}$ is 0, then $\Delta A = 0$. Similarly, if any one of $\{n_{b_1}, n_{b_2}, n_B\}$ is 0, then $\Delta B = 0$.

Next, we develop an upper bound of $SS_B(X_iX_j, Y)$. We first show the derivation of an upper bound of Δ_A. A similar idea can be applied to find an upper bound of Δ_B.

4.2 Bounds of ΔA and ΔB

Let $\{y_m | y_m \in A\} = \{y_{A_1}, y_{A_2}, \cdots, y_{A_{n_A}}\}$ be the phenotype values in group A. Without loss of generality, assume that these phenotype values are arranged in ascending order, i.e.,

$$y_{A_1} \le y_{A_2} \le \cdots \le y_{A_{n_A}}.$$

The derivative of ΔA with respect to T_{a_1} is:

$$\frac{d\Delta A}{dT_{a_1}} = \frac{2n_A(n_A T_{a_1} - n_{a_1}T_A)}{n_{a_1}(n_A - n_{a_1})n_A}.$$

Thus we have

$$\Delta A \text{ monotonically } \begin{cases} \text{increases} & \text{if } T_{a_1} \ge \dfrac{n_{a_1}T_A}{n_A}; \\ \text{decreases} & \text{if } T_{a_1} \le \dfrac{n_{a_1}T_A}{n_A}. \end{cases}$$

We have the range of T_{a_1}:

$$T_{a_1} \in [l_{a_1}, u_{a_1}] = [\sum_{i=1}^{n_{a_1}} y_{A_i}, \sum_{i=n_A-n_{a_1}+1}^{n_A} y_{A_i}].$$

The maximum value of ΔA is attained when $T_{a_1} = l_{a_1}$ or $T_{a_1} = u_{a_1}$, i.e.,

$$\Delta A \le \frac{\max\{(n_A l_{a_1} - n_{a_1}T_A)^2, (n_A u_{a_1} - n_{a_1}T_A)^2\}}{n_{a_1}(n_A - n_{a_1})n_A}. \quad (4)$$

We use $R_1(X_iX_jY)$ to denote this upper bound.

Let $\{y_m | y_m \in B\} = \{y_{B_1}, y_{B_2}, \cdots, y_{B_{n_B}}\}$ be the phenotype values in group B. Without loss of generality, assume that these phenotype values are arranged in ascending order, i.e.,

$$y_{B_1} \le y_{B_2} \le \cdots \le y_{B_{n_B}}.$$

Similarly, we can derive the bound on ΔB:

$$\Delta B \le \frac{\max\{(n_B l_{b_1} - n_{b_1}T_B)^2, (n_B u_{b_1} - n_{b_1}T_B)^2\}}{n_{b_1}(n_B - n_{b_1})n_B}. \quad (5)$$

Symbols	Formulas
l_{a_1}	$\sum_{i=1}^{n_{a_1}} y_{A_i}$
u_{a_1}	$\sum_{i=n_A-n_{a_1}+1}^{n_A} y_{A_i}$
$R_1(X_iX_jY)$	$\dfrac{\max\{(n_A l_{a_1} - n_{a_1}T_A)^2, (n_A u_{a_1} - n_{a_1}T_A)^2\}}{n_{a_1}(n_A - n_{a_1})n_A}$
l_{b_1}	$\sum_{i=1}^{n_{b_1}} y_{B_i}$
u_{b_1}	$\sum_{i=n_B-n_{b_1}+1}^{n_B} y_{B_i}$
$R_2(X_iX_jY)$	$\dfrac{\max\{(n_B l_{b_1} - n_{b_1}T_B)^2, (n_B u_{b_1} - n_{b_1}T_B)^2\}}{n_{b_1}(n_B - n_{b_1})n_B}$

Table 2: Notations for the bounds on ΔA and ΔB

We use $R_2(X_iX_jY)$ to denote this upper bound. The symbols used in Inequalities (4) and (5) are summarized in Table 2.

From Equation (3), Inequalities (4) and (5), we have the overall upper bound on $SS_B(X_iX_j, Y)$:

THEOREM 4.1. *(Upper bound of $SS_B(X_iX_j, Y)$)*

$$SS_B(X_iX_j, Y) \le SS_B(X_i, Y) + R_1(X_iX_jY) + R_2(X_iX_jY).$$

PROPERTY 4.2. *The upper bound in Theorem 4.1 is tight.*

The tightness of the bound is obvious from the derivation of the upper bound, since there exists some genotype of SNP-pair (X_iX_j) that makes the equality hold. For the same reason, we have the following property.

PROPERTY 4.3. *The upper bound in Theorem 4.1 does not exceeds the total sum of squared deviations, i.e.,*

$$SS_B(X_i, Y) + R_1(X_iX_jY) + R_2(X_iX_jY) \le SS_T(X_iX_j, Y).$$

5. THE FASTANOVA ALGORITHM

In this section, we show how our algorithm FastANOVA utilizes the upper bound in Theorem 4.1 to achieve efficient two-locus ANOVA testing. In Section 5.1, we describe the method for Problem (2) discussed in Section 3, that is, given the threshold F_α, find all SNP-pairs whose F-statistics are greater than F_α. Then in Section 5.2, we discuss how FastANOVA performs in permutation procedure, i.e., the scenario of Problem (2) in Section 3.

5.1 One Phenotype

Given the threshold F_α, to find all SNP-pairs whose F-statistics are greater than F_α, a brute-force approach is to enumerate all SNP-pairs. To expedite this process, we employ the inequality in Theorem 4.1 to prune SNP pairs that will have no chance to pass the significance threshold F_α. From Equation (2), we know that finding SNP-pairs (X_iX_j) whose F-statistics $F(X_iX_j, Y) \ge F_\alpha$ is equivalent to finding SNP-pairs satisfying

$$SS_B(X_iX_j, Y) \ge \frac{SS_T(X_i, Y)}{\frac{M-g}{(g-1)F_\alpha} + 1} = \theta.$$

Theorem 4.1 suggests that we only need to compute the F-statistics for the SNP-pairs that satisfy:

$$SS_B(X_i, Y) + R_1(X_iX_jY) + R_2(X_iX_jY) \ge \theta.$$

We refer to these SNP-pairs as *candidate* SNP-pairs.

We now discuss how to apply the upper bound in Theorem 4.1 in detail. The set of all SNP-pairs is partitioned into non-overlapping

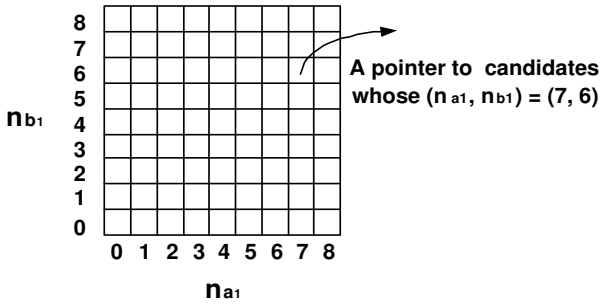


Figure 2: The index array $Array(X_i)$ for efficient retrieval of the candidate SNP-pairs

groups such that the upper bound can be readily applied to each group. For every X_i $(1 \leq i \leq N)$, let $AP(X_i)$ be the set of SNP-pairs

$$AP(X_i) = \{(X_iX_j)|i+1 \leq j \leq N\}.$$

For all SNP-pairs in $AP(X_i)$, n_A, T_A, n_B, T_B and $SS_B(X_i, Y)$ are constants. Moreover, l_{a_1}, u_{a_1} are determined by n_{a_1}, and l_{b_1}, u_{b_1} are determined by n_{b_1}. Therefore, in the upper bound, n_{a_1} and n_{b_1} are the only variables that depend on X_j and may vary for different SNP-pairs (X_iX_j) in $AP(X_i)$.

Note that n_{a_1} is the number of 1's in X_j when X_i takes value 1, and n_{b_1} is the number of 1's in X_j when X_i takes value 0. It is easy to prove that switching n_{a_1} and n_{a_2} does not change the F-statistic value and the correctness of the upper bound. This is also true if we switch n_{b_1} and n_{b_2}. Therefore, without loss of generality, we can always assume that n_{a_1} is the smaller one between the number of 1's and number of 0's in X_j when X_i takes value 1, and n_{b_1} is the smaller one between the number of 1's and number of 0's in X_j when X_i takes value 0. The following property specifies the values that n_{a_1} and n_{b_1} can take. The proof is straightforward and omitted here.

PROPERTY 5.1. *If there are m 1's and $(M-m)$ 0's in X_i, then for any $(X_iX_j) \in AP(X_i)$, the possible values that n_{a_1} can take are $\{0, 1, 2, \cdots, \lfloor m/2 \rfloor\}$. The possible values that n_{b_1} can take are $\{0, 1, 2, \cdots, \lfloor (M-m)/2 \rfloor\}$.*

To efficiently retrieve the candidates, the SNP-pairs (X_iX_j) in $AP(X_i)$ are grouped by their (n_{a_1}, n_{b_1}) values and indexed in a 2D array, referred to as $Array(X_i)$.

EXAMPLE 5.2. *Suppose that there are 32 individuals, and the genotype of X_i consists of half 0's and half 1's. Thus for the SNP-pairs in $AP(X_i)$, the possible values of n_{a_1} and n_{b_1} are $\{0, 1, 2, \cdots, 8\}$. Figure 2 shows the 9×9 array, $Array(X_i)$, whose entries represent the possible values of (n_{a_1}, n_{b_1}) for the SNP-pairs $(X_iX_j) \in AP(X_i)$. The entries in the same column have the same n_{a_1} value. The entries in the same row have the same n_{b_1} value. The n_{a_1} value of each column is noted beneath each column. The n_{b_1} value of each row is noted left to each row. Each entry of the array is a pointer to the SNP-pairs $(X_iX_j) \in AP(X_i)$ having the corresponding (n_{a_1}, n_{b_1}) values.*

Note that for a SNP-pair $(X_iX_j) \in AP(X_i)$, n_{a_1} and n_{a_2} can be calculated faster than performing the two-locus ANOVA test. To obtain n_{a_1} and n_{a_2}, we only need to count the numbers of 0's and 1's of X_j when X_i is equal to 0 and 1 respectively, which can be done by a linear scan of the $M \times 2$ binary matrix consisting of the genotypes of X_i and X_j. In contrast, to calculate the F-statistic, we first need to scan the $M \times 3$ binary matrix consisting of X_i, X_j and Y in order to find out how the phenotype values are grouped by the genotype of (X_iX_j). Then a constant time $O(t)$ is required to compute the F-statistic.

Algorithm 1: FastANOVA (no phenotype permutation)

Input: SNPs $X' = \{X_1, X_2, \cdots, X_N\}$, phenotype Y, and threshold F_α
Output: find the set of SNP-pairs $Result(Y) = \{(X_iX_j)|F(X_iX_j, Y) \geq F_\alpha, 1 \leq i < j \leq N\}$

```
for every X_i ∈ X', do
    index (X_iX_j) ∈ AP(X_i) by Array(X_i);
    access Array(X_i) to find the candidate SNP-pairs and
    store them in Cand(X_i, Y);
    for every (X_iX_j) ∈ Cand(X_i, Y) do
        if F(X_iX_j, Y) ≥ F_α then
            Result(Y) ← (X_iX_j);
        end
    end
end
return Result(Y).
```

PROPERTY 5.3. *For any SNP X_i, the maximum number of the entries in $Array(X_i)$ is $(\lceil \frac{M}{4} \rceil + 1)^2$.*

The proof of Property 5.3 is straightforward and omitted here. In order to find candidate SNP-pairs, we scan all entries in $Array(X_i)$ to calculate their upper bounds. Since the SNP-pairs indexed by the same entry share the same (n_{a_1}, n_{b_1}) value, they have the same upper bound. In this way, we can calculate the upper bound for a group of SNP-pairs together. Note that for typical genome-wide association studies, the number of individuals M is much smaller than the number of SNPs N. Therefore, the additional cost for accessing $Array(X_i)$ is minimal compared to performing ANOVA tests for all pairs $(X_iX_j) \in AP(X_i)$.

Algorithm 1 describes the FastANOVA algorithm for finding the SNP-pairs whose F-statistics are greater than the threshold F_α. The inputs of FastANOVA include the N SNPs, the phenotype Y and the critical value F_α. For each X_i, FastANOVA first indexes $(X_iX_j) \in AP(X_i)$ using $Array(X_i)$. Then it retrieves the candidate SNP-pairs by accessing $Array(X_i)$ and records them in $Cand(X_i, Y)$. The candidates in $Cand(X_i, Y)$ are then evaluated for their F-statistics. The candidates whose F-statistics are greater than or equal to F_α are reported by the algorithm.

5.2 Permutation Procedure

For multiple tests, permutation procedure is often used in genetic analysis for controlling family-wise error rate. For genome-wide association study, permutation is less commonly used because it often entails prohibitively long computation time. Our FastANOVA algorithm makes permutation procedure feasible in genome-wide association study.

Let $Y' = \{Y_1, Y_2, \cdots, Y_K\}$ be the K permutations of the phenotype Y. Following the idea discussed in Section 5.1, the upper bound in Theorem 4.1 can be easily incorporated in the algorithm to handle the permutations.

PROPERTY 5.4. *For every SNP X_i, the indexing structure $Array(X_i)$ is independent of the permuted phenotypes in Y'.*

The correctness of this property relies on the fact that, for any $(X_iX_j) \in AP(X_i)$, n_{a_1} and n_{b_1} only depend on the genotype

Algorithm 2: FastANOVA (for permutation test)

Input: SNPs $X' = \{X_1, X_2, \cdots, X_N\}$, phenotype permutations $Y' = \{Y_1, Y_2, \cdots, Y_K\}$, and the Type I error α

Output: find the critical value F_α

```
Tlist ← αK dummy phenotype permutations with F-statistics 0 ;
F_α = 0;
for every X_i ∈ X', do
    index (X_i X_j) ∈ AP(X_i) by Array(X_i);
    for every Y_k ∈ Y', do
        access Array(X_i) to find the candidate SNP-pairs and store them in Cand(X_i, Y_k);
        for every (X_i X_j) ∈ Cand(X_i, Y_k) do
            if F(X_i X_j, Y_k) ≥ F_α then
                update Tlist;
                F_α = the smallest test value in Tlist;
            end
        end
    end
end
return F_α.
```

of the SNP-pair and thus remain constant for different phenotype permutations. Therefore, for each X_i, once we build $Array(X_i)$, it can be reused in all permutations.

The FastANOVA algorithm for permutation test is described in Algorithm 2. The inputs include the N SNPs, K phenotype permutations, and the Type I error threshold α. The goal is to find the critical value F_α, which is the αK-th largest value in $\{F_{Y_k} | Y_k \in Y'\}$. Recall that F_{Y_k} is the maximum F-statistic value for phenotype Y_k. We use $Tlist$ to keep the αK phenotype permutations having the largest F-statistics found by the algorithm so far. Initially, $Tlist$ contains αK dummy phenotype permutations with test values 0. The smallest F-statistic value in $Tlist$, initially 0, is used as the threshold to prune the SNP-pairs. For each X_i, FastANOVA first indexes $(X_i X_j) \in AP(X_i)$ using $Array(X_i)$. Then it finds the set of candidate SNP-pairs $Cand(X_i, Y_k)$ by accessing $Array(X_i)$ for every phenotype permutation Y_k. The candidates in $Cand(X_i, Y_k)$ are then evaluated for their F-statistics. If a candidate's F-statistic value is greater than the current threshold, then $Tlist$ is updated accordingly: If the candidate's phenotype Y_k is not in the $Tlist$, then the phenotype in $Tlist$ having the smallest F-statistic value is replaced by Y_k. If the candidate's phenotype Y_k is already in $Tlist$, we only need to update its corresponding F-statistic value to be the maximum value found for the phenotype so far. The threshold is also updated to be the smallest F-statistic value in $Tlist$.

5.3 Complexity Analysis

In this section, we study the time and space complexities of the FastANOVA algorithm for permutation test. The complexity for a single phenotype can be analyzed in a similar way.

Time complexity: For each X_i, FastANOVA needs to index $(X_i X_j)$ in $AP(X_i)$. The complexity to build the indexing structure for all SNPs is $O(N(N-1)M/2)$. The worst case for accessing all $Array(X_i)$ for all permutations is $O(N \times K \times (\lceil \frac{M}{4} \rceil + 1)^2) = O(NKM^2)$. Let $C = \sum_{i,k} |Cand(X_i, Y_k)|$ represent the total number of candidates. The overall time complexity of FastANOVA is thus $O(N(N-1)M/2) + O(NK \times (\lceil \frac{M}{4} \rceil + 1)^2) + O(\sum_{i,k} |Cand(X_i, Y_k)|M) = O(N^2M + NKM^2 + CM)$. The experimental results show that the overhead of building the indexing structures and accessing them for candidate retrieval are negligible when large permutation tests are needed. Note that the time complexity of the brute-force approach is $O(KN(N-1)M/2) = O(KN^2M)$.

Space complexity: The total number of variables in the dataset, including the SNPs and the phenotype permutations, is $N+K$. The maximum space of the indexing structure $Array(X_i)$ is $O((\lceil \frac{M}{4} \rceil + 1)^2 + N)$. Note that for each SNP X_i, FastANOVA only needs to access one indexing structure, $Array(X_i)$, for all permutations. Once the evaluation process for X_i is done for all permutations, $Array(X_i)$ can be cleared from the memory. Therefore, the space complexity of FastANOVA is $O((N+K)M) + O((\lceil \frac{M}{4} \rceil + 1)^2 + N) = O((N+K)M)$ since $M \ll N$. The space complexity is linear to the dataset size.

6. EXPERIMENTAL RESULTS

In this section, we present extensive experimental results on evaluating the performance of the FastANOVA algorithm. We show (1) the runtime comparison between FastANOVA and the brute-force approach under various experimental settings, (2) the punning effect of the upper bound, and (3) the relative computational cost of each component of FastANOVA. FastANOVA is implemented in C++. The experiments are performed on a 2.4 GHz PC with 1G memory running WindowsXP system.

Dataset: The SNP dataset used for the experiments is extracted from a set of combined SNPs from the 140k Broad/MIT mouse dataset [26] and 10k GNF [2] mouse dataset. This merged dataset has 156,525 SNPs for 71 individuals. The missing values in the dataset are imputed using NPUTE [20]. We use both real phenotypes and synthetic phenotypes in our experiments. The real phenotype data is available from the Jackson Lab [3].

6.1 Real Phenotypes

We use three real phenotypes in our experiments: cardiovascular (blood pressure), metabolism (water intake), and neurosensory (acoustic startle response). Table 3 shows the statistics of the genotype datasets corresponding to the three phenotypes. The number of SNPs in the table indicates the number of unique SNPs in each genotype dataset.

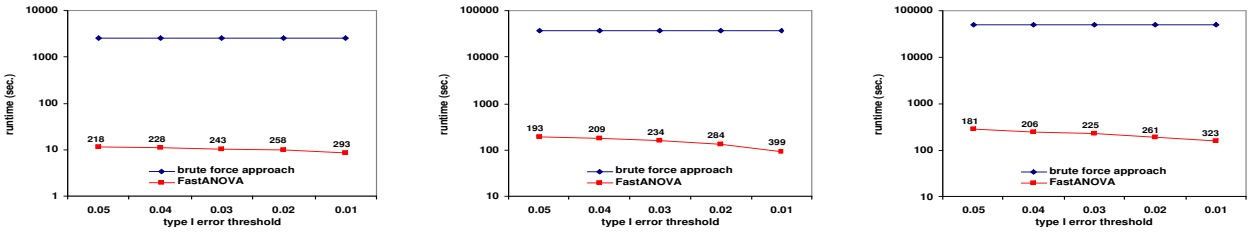
	cardiovascular	metabolism	neurosensory
# individuals	19	26	34
# SNPs	14,513	43,856	66,006

Table 3: Statistics of the genotype datasets

We first show the results on finding the critical value F_α, which is more time-consuming than finding the significance SNP-pairs given the critical value F_α for a single phenotype.

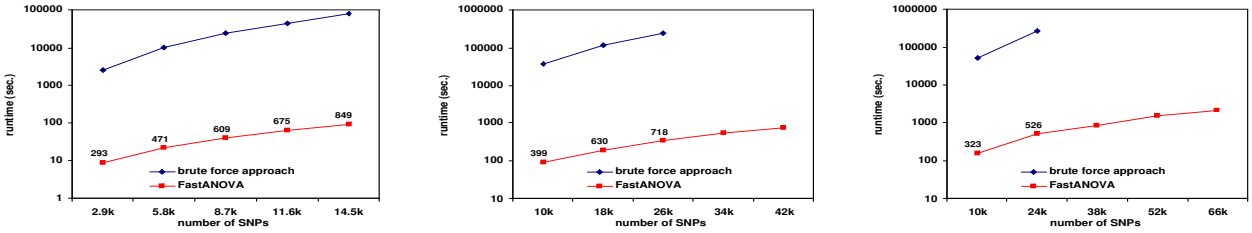
6.1.1 Finding critical value F_α

FastANOVA v.s. the brute-force approach We compare FastANOVA with the brute-force approach under various experimental settings. Since the brute-force approach is very time-consuming, we use a moderate number of SNPs and permutations in the default setting in order to show the performance comparisons. The default setting is as follows: The Type I error threshold $\alpha = 0.01$. The number of permutations is 100. The number of SNP is 10,000 for the two larger datasets of metabolism and neurosensory, and 2,900 for the cardiovascular SNP dataset. These experimental settings are chosen to demonstrate the performance gain and enhanced



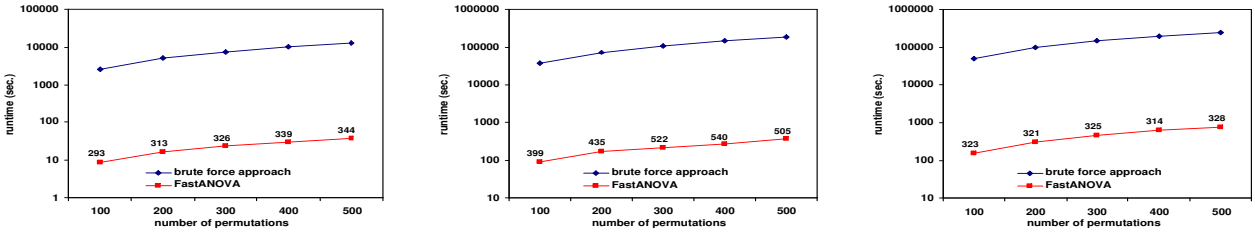
(a) cardiovascular (b) metabolism (c) neurosensory

Figure 3: Performance comparison between FastANOVA and the brute-force approach when varying Type I error thresholds



(a) cardiovascular (b) metabolism (c) neurosensory

Figure 4: Performance comparison between FastANOVA and the brute-force approach when varying the number of SNPs



(a) cardiovascular (b) metabolism (c) neurosensory

Figure 5: Performance comparison between FastANOVA and the brute-force approach when varying the number of permutations

scalability offered by FastANOVA over the brute-force implementation. FastANOVA can handle much larger SNP panels and larger number of permutation tests. The performance of FastANOVA is expected to follow the same trends presented in the remainder of this section.

Figures 3, 4, and 5 show the running time comparison of FastANOVA and the brute-force approach on the three genotype phenotype datasets using different settings. The y-axis is in logarithm scale. The numbers above the runtime line of FastANOVA indicate the ratio of the runtimes of the brute-force approach over FastANOVA. We terminate the programs that have run over 72 hours without completion.

Figure 3 shows the runtime comparison when varying the Type I error thresholds. For each dataset, the runtime of the brute-force approach does not change over different Type I error thresholds. The runtime of FastANOVA decreases as the threshold decreases. FastANOVA offers 218 fold speedup when $\alpha = 0.05$ and 293 fold speedup when $\alpha = 0.01$ on cardiovascular dataset. We can also observe a similar two-orders-of-magnitude speedup in the metabolism and neurosensory datasets. This is consistent with the pruning effect of the upper bound, which will be presented later in this section. In general, the lower the Type I error threshold, the more powerful the pruning effect, hence the faster the algorithm.

Figure 4 depicts the comparison of these two approaches when the number of SNPs changes. From these figures, it is clear that FastANOVA is about two orders of magnitude faster than the brute-force approach. The brute-force approach cannot finish in 72 hours when the number of unique SNPs is greater than 26k in the metabolism dataset and greater than 24k in the neurosensory dataset. We observe that the runtime ratio tends to increase (approaching three-orders-of-magnitude speedup) as the number of SNPs increases. This indicates that the performance gain of FastANOVA is even higher for larger SNP datasets.

Figure 5 shows the runtime comparison when the number of phenotype permutations changes. The runtime of the brute-force approach is linear with respect to the number of permutations. Fas-

		cardiovascular	metabolism	neurosensory
α	0.05	99.881%	99.724%	99.701%
	0.04	99.907%	99.758%	99.751%
	0.03	99.928%	99.797%	99.792%
	0.02	99.949%	99.877%	99.853%
	0.01	99.974%	99.929%	99.911%
# SNPs	1st	99.974%	99.929%	99.911%
	2nd	99.991%	99.985%	99.979%
	3rd	99.996%	99.996%	99.997%
	4th	99.998%	99.996%	99.997%
	5th	99.998%	99.993%	99.998%
# Perm.	100	99.974%	99.929%	99.911%
	200	99.966%	99.935%	99.917%
	300	99.977%	99.962%	99.919%
	400	99.977%	99.961%	99.914%
	500	99.974%	99.953%	99.907%

Table 4: Pruning effects on cardiovascular, metabolism and neurosensory datasets when finding critical value F_α

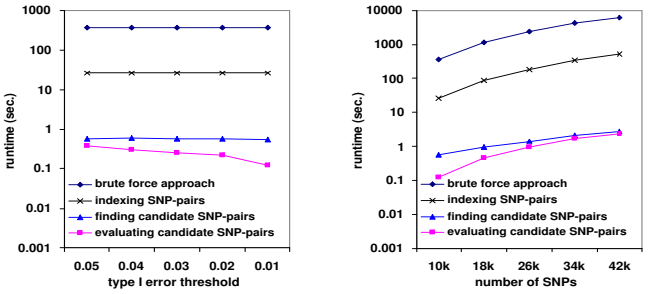
tANOVA is consistently two orders of magnitude faster than the brute-force approach. The performance gap increases as the number of permutations increases.

Pruning effect of the upper bound Table 4 shows the percentage of SNP-pairs pruned under different experimental settings. Since the three datasets have different numbers of SNPs, the 1st to 5th rows in the category of "# SNPs" correspond to the settings from left to right on x-axis in each plot in Figure 4. Most SNP-pairs are pruned under all settings. Moreover, as the Type I error threshold α decreases, the pruning ratio increases, which is consistent with runtime comparison shown in Figure 3. As the number of SNPs increases, the pruning ratio also increases. This is because, with more SNPs, the dynamic threshold used to prune the search space becomes higher. Hence a larger portion of SNPs are pruned. This is consistent with results shown in Figure 4. Note that from Table 4 we observe that the pruning ratio tends to remain steady when the number of permutations changes. However, we observe that the runtime ratio increases as the number of permutations increases. The reason for these two different trends will become clear after we show the results on the computational cost of each component of FastANOVA in the next subsection.

6.1.2 *Finding significant SNP-pairs*

In this subsection, we study the comparison between FastANOVA and the brute-force approach in finding significant SNP-pairs given a critical value F_α. Only the original phenotype (without permutations) is used in this procedure. We examine the detailed computation cost of each component of the FastANOVA algorithm. FastANOVA has three major components: building the indexing structure $Array(X_i)$ for every SNP X_i, accessing $Array(X_i)$ to find the candidate SNP-pairs, and performing ANOVA tests on these candidates.

Due to space limitation, we only show the performance comparison on the metabolism dataset. Similar behaviors are also observed on the other two datasets. The default experimental setting is the same as before. Figure 6(a) and Figure 6(b) show the runtime of these three components when varying the Type I error threshold and number of SNPs in the dataset respectively. Since F_α is a function of α, in Figure 6(a), we plot the runtime with respect to α. In both figures, the three lines from the bottom show the runtime of these three components. The runtime of the brute-force approach is the top line. As we can see from these two figures, performing two-locus ANOVA tests on candidate SNP pairs is two to three orders of



(a) Varying threshold values (b) Varying number of SNPs

Figure 6: Runtime of each component of FastANOVA v.s. runtime of the brute-force approach in the process of finding significant SNP-pairs

cardiovascular	metabolism	neurosensory
97.865%	97.844%	98.061%

Table 5: Pruning effect on cardiovascular, metabolism and neurosensory datasets when finding F_{Y_k} for all permutations

magnitude faster than performing such tests on all SNP-pairs. This is the benefit of the upper bound pruning since most SNP-pairs have been pruned and only a very small portion of candidates need to be evaluated for their F-statistics. The cost for accessing the indexing structures is also small, which demonstrates the efficiency of the method introduced in Section 5.1 for candidate retrieval. Among the three components of FastANOVA, the most time-consuming one is building the index structures. Yet, its runtime is only a small fraction of the runtime of performing the two-locus ANOVA tests on all SNP pairs. Note that, in permutation test, building the index structures is a one time cost. Once the index structures are built, they can be reused in all permutations. Therefore, the amortized overhead per permutation decreases when the number of permutations increases. This is why the pruning ratio remains steady as in Table 4 while the runtime ratio increases as in Figure 5 when the number of permutations increases.

6.1.3 *Finding F_{Y_k} for all permutations*

Sometimes the users may be interested in finding F_{Y_k} values of all phenotype permutations. In this way, the users can get the critical value F_α for any Type I error threshold α ranging from 0 to 1, without re-running the permutation tests for different thresholds. Recall that, given a set of phenotype permutations $Y' = \{Y_1, Y_2, \cdots, Y_K\}$, $F_{Y_k} = \max\{F(X_iX_j, Y_k) | 1 \leq i < j \leq N\}$ is the maximum F-statistic value for permutation Y_k. F_α is the αK-th largest value in $\{F_{Y_k} | Y_k \in Y'\}$. In this subsection, we show the pruning effect of the upper bound when it is applied to determine F_{Y_k} for every Y_k ($1 \leq k \leq K$). Note that in this case, for each permutation Y_k, the dynamic threshold used to prune the search space is the largest F-statistic value of Y_k identified by the algorithm so far.

Table 5 shows the pruning ratio of applying the upper bound to the three real phenotype datasets. The experimental setting is the same as the default setting before. As expected, the pruning ratios are slightly lower than those in Table 4, where smaller Type I error thresholds are used to prune the search space. However, the pruning ratios on all three datasets are still above 97%. Moreover, finding

		uniform	normal	exponential
α	0.05	96.469%	97.793%	99.335%
	0.04	96.888%	98.222%	99.401%
	0.03	97.695%	98.631%	99.502%
	0.02	98.712%	99.072%	99.617%
	0.01	99.605%	99.506%	99.737%
# SNPs	10k	99.605%	99.506%	99.737%
	22k	99.864%	99.814%	99.924%
	34k	99.907%	99.905%	99.967%
	46k	99.928%	99.889%	99.965%
	58k	99.941%	99.942%	99.963%
# Perm.	100	99.605%	99.506%	99.737%
	200	98.891%	99.398%	99.726%
	300	98.897%	99.072%	99.780%
	400	98.623%	99.315%	99.762%
	500	98.709%	99.199%	99.759%
# indiv.	28	99.756%	99.695%	99.893%
	30	99.422%	99.577%	99.880%
	32	99.605%	99.506%	99.737%
	34	99.073%	99.289%	99.773%
	36	98.736%	98.832%	99.745%

Table 6: Pruning effect when finding critical value F_α using three synthetic phenotypes

all F_{Y_k} provides the advantage that we can get the F_α values for all possible α values instead of just for a specific one.

6.2 Synthetic Phenotypes

To further study the performance of FastANOVA, we generate three synthetic phenotypes whose values follow three different distributions: uniform, standard normal, and standard exponential distribution. Our purpose is to study the pruning effect of the upper bound under different phenotype distributions. The default setting of the experiments in this subsection is as follows: #individuals = 32, #SNPs=10,000, #permutations=100, $\alpha = 0.01$. There are 60,970 unique SNPs for these 32 individuals.

Table 6 shows the pruning ratio of FastANOVA under different settings using permutation test. In this table, we also include the pruning ratio when the number of individuals varies. We observe that the pruning effects are similar to that of real phenotypes, which indicates that the upper bound pruning is effective and insensitive to different phenotype distributions.

7. CONCLUSION AND FUTURE WORK

The large number of available SNPs poses great computational challenge to the genome-wide association study. To assess the significance of the findings, permutation test is usually required. These factors make the association study a very time-consuming process. Thus tools that can improve the efficiency of the association study are in demand.

In this paper we present an efficient algorithm, FastANOVA, for genome-wide two-locus ANOVA test. FastANOVA is a complete algorithm which guarantees to find the optimal solution. Experimental results demonstrate that FastANOVA is two to three orders of magnitude faster than the brute-force alternative. The efficiency of FastANOVA is gained from two sources. First, it utilizes an upper bound of the two-locus ANOVA test value to prune a majority of the SNP-pairs. Second, it identifies and reuses computation units that are independent of the phenotype and hence are invariant in permutation test. By eliminating redundant computation of these invariant units, FastANOVA is much more efficient than the brute-force method.

Even though FastANOVA is designed for two-locus association study of binary SNPs, the principles used in FastANOVA are general and applicable to the association study on SNP subsets containing more than two SNPs, and the heterozygous case where SNPs are encoded as $\{0, 1, 2\}$. In our future work, we will investigate how to apply these principles for association study considering joint effects of more than two SNPs and the heterozygous case.

8. ACKNOWLEDGMENTS

This research was partially supported by EPA grant STAR-RD832720, NSF grant IIS-0448392, and a Micosoft New Faculty Fellowship.

9. REFERENCES

[1] *http://www.broad.mit.edu/.*
[2] *http://www.gnf.org/.*
[3] *http://www.jax.org/.*
[4] D. J. Balding. A tutorial on statistical methods for population association studies. *Nature Reviews Genetics*, 7(10):781–791, 2006.
[5] O. Carlborg, L. Andersson, and B. Kinghorn. The use of a genetic algorithm for simultaneous mapping of multiple interacting quantitative trait loci. *Genetics*, 155(4):2003–2010, 2000.
[6] C. S. Carlson, M. A. Eberle, L. Kruglyak, and D. A. Nickerson. Mapping complex disease loci in whole-genome association studies. *Nature*, 429:446–452, 2004.
[7] R. W. Doerge. Multifactorial genetics: Mapping and analysis of quantitative trait loci in experimental populations. *Nature Reviews Genetics*, 3:43–52, 2002.
[8] D. M. Evans, J. Marchini, A. P. Morris, and L. R. Cardon. Two-stage two-locus models in genome-wide association. *PLoS Genet.*, 2: e157, 2006.
[9] E. Halperin, G. Kimmel, and R. Shamir. Tag snp selection in genotype data for maximizing snp prediction accuracy. In *Proc. ISMB*, 2005.
[10] J. Hoh and et al. Selecting snps in two-stage analysis of disease association data: a model-free approach. *Ann. Hum. Genet.*, 64:413–417, 2000.
[11] J. Hoh and J. Ott. Mathematical multi-locus approaches to localizing complex human trait genes. *Nature Reviews Genetics*, 4:701–709, 2003.
[12] F. Y. Ideraabdullah and et al. Genetic and haplotype diversity among wild-derived mouse inbred strains. *Genome Res.*, 14(10a):1880–1887, 2004.
[13] H. Liu and H. Motoda. *Feature selection for knowledge discovery and data mining*. Boston: Kluwer Academic Publishers, 1998.
[14] R. G. Miller. *Simultaneous Statistical Inference*. Springer Verlag New York, 1981.
[15] R. Nakamichi and et al. Detection of closely linked multiple quantitative trait loci using a genetic algorithm. *Genetics*, 158(1):463–475, 2001.
[16] M. R. Nelson, S. L. Kardia, R. E. Ferrell, and C. F. Sing. A combinatorial partitioning method to identify multilocus genotypic partitions that predict quantitative trait variation. *Genome Research*, 11:458–470, 2001.
[17] Y. Ohno and et al. Selective genotyping with epistasis can be utilized for a major quantitative trait locus mapping in hypertension in rats. *Genetics*, 155:785–792, 2000.
[18] M. Pagano and K. Gauvreau. *Principles of Biostatistics*. Pacific Grove, CA: Duxbury Press, 2000.
[19] M. D. Ritchie, L. W. Hahn, N. Roodi, L. R. Bailey, W. D. Dupont, F. F. Parl, and J. H. Moore. Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer. *American Journal of Human Genetics*, 69:138–147, 2001.
[20] A. Roberts, L. McMillan, W. Wang, J. Parker, I. Rusyn, and D. Threadgill. Inferring missing genotypes in large snp panels using fast nearest-neighbor searches over sliding windows. In *Proc. ISMB*, 2007.
[21] R. Saxena and et al. Genome-wide association analysis identifies loci for type 2 diabetes and triglyceride levels. *Science*, 316:1331–1336, 2007.
[22] A. Scuteri and et al. Genome-wide association scan shows genetic variants in the fto gene are associated with obesity-related traits. *PLoS Genet.*, 3(7), 2007.
[23] P. Sebastiani, R. Lazarus, S. T. Weiss, L. M. Kunkel, I. S. Kohane, and M. F. Ramoni. Minimal haplotype tagging. *Proc. Natl. Acad. Sci. USA*, 100(17):9900–9905, 2003.
[24] D. Segrè, A. DeLuna, G. M. Church, and R. Kishony. Modular epistasis in yeast metabolism. *Nat. Genet.*, 37:77–83, 2005.
[25] K. Shimomura and et al. Genome-wide epistatic interaction analysis reveals complex genetic determinants of circadian behavior in mice. *Genome Res.*, 11(6):959–980, 2001.
[26] C. M. Wade and M. J. Daly. Genetic variation in laboratory mice. *Nat. Genet.*, 37:1175–1180, 2005.
[27] M. N. Weedon and et al. A common variant of hmga2 is associated with adult and childhood height in the general population. *Nat. Genet.*, 39:1245–1250, 2007.