Evaluating the Computational Efficiency of XFBAT and FBAT for Family Based Studies

Yanwei Ouyang

Utah State University

Follow this and additional works at: https://digitalcommons.usu.edu/gradreports

Part of the Mathematics Commons

Recommended Citation
EVALUATING THE COMPUTATIONAL EFFICIENCY OF XFBAT AND FBAT FOR FAMILY BASED STUDIES

by

Yanwei Ouyang

A report submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE in

Statistics

UTAH STATE UNIVERSITY
Logan, UT
2006
ABSTRACT

Family-based study designs are often employed when investigating the genetic causes of complex disease. While the transmission disequilibrium test (TDT) and its extensions were developed to use family data for assessing linkage between a known genetic marker and a disease-causing gene, the so-called FBAT approach proposed by Rabinowitz and Laird (2000) effectively subsumes these family-based procedures as special cases. FBAT is fully conditional, but its implementation in the freely available FBAT software package uses a large-sample distributional approximation to compute p-values. The exact distribution for FBAT can be enumerated, but doing so explicitly is computationally intensive, particularly for relatively larger sample sizes. Schneiter et al. (2005) proposed an efficient algorithm for computing the exact p-value, but the computational performance of this procedure has not been systematically evaluated. In this report, we carry out a simulation study to determine the relative computational efficiency of large-sample FBAT and exact FBAT (XFBAT). Characteristic of all exact significance tests in statistical practice, XFBAT is significantly more computationally burdensome, and its efficiency decreases exponentially for larger sample sizes. In addition, XFBAT is more computationally intensive under the additive inheritance model, as opposed to either the dominant or recessive models.
ACKNOWLEDGMENTS

I would like to express my gratitude to my major advisor, Dr. Chris Corcoran. I give thanks for his detailed direction, continuous encouragement, and kindly support for this report study. Working under his supervision was an invaluable and rewarding experience for me.

I am grateful to my committee members, Dr. John R. Stevens and Dr. Ronald G. Munger, who gave me valuable comments, suggestions and corrections. I am also grateful to all my friends, colleagues, and many others who gave me support and assistance throughout the entire process.

Finally, I thank to my dear parents and my sister for their love, care and support over all these years. Their loves are the best encouragement in my life.

Yanwei Ouyang
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>CONTENTS</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vii</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Overview</td>
<td>1</td>
</tr>
<tr>
<td>1.2 The Transmission Disequilibrium Test (TDT) and Related Procedures</td>
<td>2</td>
</tr>
<tr>
<td>1.3 FBAT and XFBAT Procedure</td>
<td>5</td>
</tr>
<tr>
<td>SIMULATION STUDY</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Simulation Procedure</td>
<td>7</td>
</tr>
<tr>
<td>SIMULATION RESULTS</td>
<td>11</td>
</tr>
<tr>
<td>3.1 Performance Comparison</td>
<td>11</td>
</tr>
<tr>
<td>3.2 Efficiency Comparison</td>
<td>13</td>
</tr>
<tr>
<td>CONCLUSION</td>
<td>17</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>18</td>
</tr>
<tr>
<td>APPENDIX</td>
<td>20</td>
</tr>
<tr>
<td>Appendix I. Data Simulation in R Code</td>
<td>20</td>
</tr>
<tr>
<td>Appendix II. A sample of simulation data file</td>
<td>23</td>
</tr>
<tr>
<td>Appendix III. Perl Code for Time Comparison between FBAT and XFBAT</td>
<td>24</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Type of transmission</td>
<td>3</td>
</tr>
<tr>
<td>2-1</td>
<td>Parameter values set in simulation</td>
<td>9</td>
</tr>
<tr>
<td>3-1</td>
<td>P-value for the exact and asymptotic tests</td>
<td>12</td>
</tr>
<tr>
<td>3-2</td>
<td>FBAT and XFBAT Computation Time Results (in seconds)</td>
<td>14</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

Figure                                      Page
2-1  Parental genotype simulation            9
2-2  Offspring’s genotype simulation         10
3-1  Plots of exact and asymptotic p-values  11
3-2  Computation CPU time comparison between FBAT and XFBAT  15
3-3  Relative Ratio between XFBAT and FBAT Computation Time  16
3-4  XFBAT computation time for additive, dominant, recessive model  16
1.1 Overview

On the heels of significant discoveries that better delineated the role of DNA in familial inheritance, genetic epidemiology emerged as a discipline in the 1960s to study the associations between genes and complex diseases. Genetic epidemiologic research usually includes the following steps: 1) establish a potential genetic component to an investigative disorder, 2) determine the relative size of a genetic effect in relation to other sources of variation in disease risk, and 3) identify the specific genetic variants responsible for increased or decreased disease risk. These objectives can be achieved either by sampling families (which can establish segregation, linkage, and/or association) or by using conventional population-based studies of unrelated individuals (which can establish association only) [1].

Among several conventional study designs employed by genetic epidemiologists, family-based methods are among the most widely used. Such designs might include sampling sibling pairs, nuclear families, complex pedigrees, or twins, in order to distinguish between the respective roles of genetics and environmental factors. Complex pedigrees (preferably with more than one affected member) allow so-called segregation analysis, which helps to determine the mode of inheritance (e.g., a recessive or dominant trait).
Family-based designs also allow linkage studies, which can establish the physical proximity of known genetic markers to an actual disease-related locus. Linkage analyses can be either parametric or nonparametric, depending upon the assumptions required regarding the underlying genetic model. Nonparametric linkage procedures include allele-sharing methods, which require no assumptions about the mode of inheritance, the level of penetrance (i.e., probability of disease given the genotype of interest), or the prevalence of the investigative allele. The underlying principle of linkage studies is the cosegregation of two loci – one of which is the disease-related gene.

Simple genetic association can be assessed using conventional population-based or case-control designs (both of which result in samples of unrelated subjects), or family-based designs. While studies of unrelated subjects can yield relatively powerful tests of association – detecting even modest genotypic effects – their significant disadvantage is that association does not automatically imply linkage. Confounding, due to genetic heterogeneity (i.e., population admixture) within the target population, can lead to spurious findings of association.

1.2 The Transmission Disequilibrium Test (TDT) and Related Procedures

In contrast to sampling unrelated individuals, family-based association studies have no difficulty with confounding due to admixture. To illustrate why this is so, consider a simple design for the nonparametric transmission disequilibrium test (TDT) proposed by
Spielman et al. (1993)[2]. They suggested sampling triples that include two parents with known genotypes and an affected offspring. In the presence of only two marker alleles, the TDT procedure compares the frequency of alleles transmitted to affected offspring to the frequency of alleles not transmitted to the affected offspring. Data gathered from such a design can be represented in the following table:

Table 1-1. Types of Transmission

<table>
<thead>
<tr>
<th>TRANSMITTED ALLELE</th>
<th>NONTRANSMITTED ALLELE</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>m</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>c</td>
<td>d</td>
</tr>
</tbody>
</table>

We assume two possible alleles, M and m, at the investigative marker locus. The terms a and d refer to transmission frequencies from homozygous parents. These contain information on association but not on linkage. The cell counts b and c denote transmission frequencies from heterozygous parents. These contain information on both linkage and association. The only informative counts are b and c, since no information about differential transmission rates between M and m is contained in a or d. Note that b counts the number of times parents transmit M but not m to an affected child, and c counts the number of times parents transmit m but not M.

Under the null hypothesis of no linkage or no association, we would expect the two alleles to be transmitted in roughly equal proportion to the affected offspring. If transmission is disproportionate, this provides evidence that the investigative marker is
physically proximate to the disease-causing gene (or that they are one in the same).

Hence, with no linkage or no association, $b$ and $c$ should be approximately equal. The form of the TDT statistic is given by

$$T = \frac{(b - c)^2}{b + c}$$

which has a large-sample $\chi^2$ distribution. Alternatively, for small sample sizes, exact probability values can be computed based on the binomial distribution with sample size $b + c$ and “success” probability $1/2$.

An advantage of the transmission disequilibrium test and the many subsequently developed related methods [3-4] is that they do not require a genetic model for disease transmission, which can be difficult to specify for complex traits. Moreover, by conditioning on parental marker genotypes we completely eliminate potential confounding due to admixture, bias due to nonrandom mating, and the need for allele frequencies. However, a significant limitation of the TDT test in particular is the need for parental genotypes, which can be nearly impossible to obtain in studies of diseases related to old age.

A number of methods have been developed to account for missing parents, such as analyzing only the sub-sample of complete data, reconstructing the missing parental genotypes [5], using “informative” parent-child pairs, enumeration of all possible genotypes for the missing parents along with their corresponding relative probabilities [6], or even using unaffected siblings as controls to replace the parental information [7-8].
1.3 FBAT and XFBAT Procedure

While many extensions of the TDT have been developed to address a variety of special cases, the specific study design required by each can prove difficult to implement in practice. Rabinowitz and Laird suggested a method (dubbed FBAT, for Family-Based Association Test) that unifies these designs under a single procedure, effectively incorporating the TDT and its extensions as special cases [9]. FBAT is a general method for assessing linkage that accommodates data from sampled sibships with arbitrary patterns of missing parental genotypes and arbitrary numbers of affected siblings. The FBAT procedure involves conditioning on appropriate information (the sufficient statistics) within each family to compute the testing distribution under the null hypothesis.

The FBAT statistic measures the association between the marker and the disease. It is defined as

\[ S = \sum_{i=1}^{N} \sum_{j=1}^{K} X_{ij} T_{ij} \]  

where \( i \) indexes the families, \( j \) indexes offspring within family, and \( T_{ij} \) is a function of the \((i, j)th\) child’s phenotype. For example, with a dichotomous trait \( T_{ij} = 1 \) if the \((i, j)th\) child is affected and \( T_{ij} = 0 \) if \((i, j)th\) child is unaffected. \( X_{ij} \) denotes some function of the genotype of the \((i, j)th\) child at the locus being tested. The value of \( X_{ij} \) depends on the genetic model (e.g., additive, recessive, or dominant). We illustrate the coding of \( X_{ij} \) by considering a biallelic marker with alleles denoted by \( A \) and \( B \). Under the additive
model, $X_{ij}$ is a count of the $A$ alleles carried by the $(i,j)th$ subject. Under the dominant model, $X_{ij}$ is 1 if the genotype is either $AA$ or $AB$, and $X_{ij}$ equals 0 if the genotype is $BB$. Under the recessive model, $X_{ij}$ equals 1 if the genotype is $AA$, and $X_{ij}$ equals 0 otherwise.

The distribution of the FBAT statistic under the null hypothesis is computed by considering the probability mass function of the offspring genotypes, conditional on known parental genotypes or on other sufficient statistics when parental genotypes are missing [9]. The large-sample conditional distribution of the FBAT statistic is chi-square. FBAT software and documentation are freely available on the web [10].

The conditional nature of the FBAT procedure suggests an exact test, which can prove particularly useful in the presence of small or sparse samples where the large-sample approximation may break down [11-12]. However, exact tests are generally computationally intensive, and exact FBAT is no exception. Schneiter et al. suggest an efficient algorithm (XFBAT) to make computation of the exact p-value faster and more feasible for relatively larger data sets [13]. The trade-off between large-sample and exact FBAT hence is a choice between p-value accuracy and computational efficiency. In this report, we carry out a simulation study to compare CPU computation time between large-sample FBAT and XFBAT.
CHAPTER 2

SIMULATION STUDY

2.1 Simulation Procedure

We carried out a simulation study in order to assess the computational efficiency of the XFBAT algorithm in the biallelic case [14]. We assume two alleles, $A$ and $B$, with respective frequencies $p_A$ and $p_B$. Let $q_0$, $q_1$, and $q_2$ denote the penetrance functions corresponding to zero, one, and two instances of the allele of interest, so that by definition under the additive model $q_1 = (q_0 + q_2) / 2$, under the recessive model $q_2 > q_1 = q_0$, and under the dominant model $q_2 = q_1 > q_0$. In addition, let $K$ denote the population prevalence of disease, and $AF$ denote the attributable fraction, representing the proportion of disease prevalence that may be ascribed to the presence of the disease-predisposing genotypes. $K$ and $AF$ are defined to satisfy $K = p_A^2q_0 + 2p_Ap_Bq_1 + p_B^2q_2$, and $AF = (K - q_0) / K$.

Given values of $p_A$, $K$ and $AF$, along with a given inheritance model, then $q_0$, $q_1$, and $q_2$ are uniquely determined as follows:

Additive Model: \[
\begin{align*}
q_0 &= K \times (1 - AF) \\
q_1 &= (q_0 + q_2) / 2 \\
q_2 &= q_0 + (K - q_0) / p_A
\end{align*}
\] (2-1)
\[
\begin{align*}
q_0 &= K \times (1 - AF) \\
q_1 &= ((1 - p_A)^2 \times q_0 - K)/(1 - p_A)^2 - 1) \\
q_2 &= q_1
\end{align*}
\] (2-2)

\[
\begin{align*}
q_0 &= K \times (1 - AF) \\
q_1 &= q_0 \\
q_2 &= (K - q_0)/p_A + q_0
\end{align*}
\] (2-3)

Parameters for the simulation data used in this report are presented in Table 2-1. Figures 2-1 and 2-2 describe how offspring data are generated using parameter values as specified above. Parental genotypes for a given family were generated first assuming given allele frequencies \(p_A\) and \(p_B\). Children's genotypes and disease status are respectively simulated conditional on parental genotypes, disease prevalence \(K\), and the attributable fraction \(AF\) of disease due to the investigative gene. Data were generated using the R programming language. The source code, along with a representative simulated family, are contained in Appendices I and II. The data shown in Appendix II demonstrate how a pedigree file must be stored for use with the FBAT software. The file line of the data file is the names of all markers in the sequence of the genotype data. For example, we use “Gene500” to represent simulation gene data with 500 pedigrees. The remaining lines are simulated data in the following order: pedigree id, patient id, father id, mother id, gender, affection status, \(A_i\). For gender, 1 means male and 2 means female. For affection status, 1 means unaffected and 2 means affected. \(A_i\) is a count of the allele of interest \(j\) of marker \(i\) \((j = 1,2; i = 1,2,\ldots)\).
Table 2-1. Parameter values set in simulation.

<table>
<thead>
<tr>
<th>Model</th>
<th>$K$</th>
<th>$AF$</th>
<th>$p_A$</th>
<th>$q_0$</th>
<th>$q_1$</th>
<th>$q_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additive</td>
<td>0.01</td>
<td>0.5</td>
<td>0.2</td>
<td>0.005</td>
<td>0.018</td>
<td>0.030</td>
</tr>
<tr>
<td>Additive</td>
<td>0.05</td>
<td>0.5</td>
<td>0.5</td>
<td>0.025</td>
<td>0.050</td>
<td>0.075</td>
</tr>
<tr>
<td>Dominant</td>
<td>0.01</td>
<td>0.5</td>
<td>0.2</td>
<td>0.005</td>
<td>0.019</td>
<td>0.019</td>
</tr>
<tr>
<td>Dominant</td>
<td>0.05</td>
<td>0.5</td>
<td>0.5</td>
<td>0.025</td>
<td>0.058</td>
<td>0.058</td>
</tr>
<tr>
<td>Recessive</td>
<td>0.01</td>
<td>0.5</td>
<td>0.2</td>
<td>0.005</td>
<td>0.005</td>
<td>0.030</td>
</tr>
<tr>
<td>Recessive</td>
<td>0.05</td>
<td>0.5</td>
<td>0.5</td>
<td>0.025</td>
<td>0.025</td>
<td>0.075</td>
</tr>
</tbody>
</table>

Figure 2-1 Parental genotype simulation
Generate Random Number 0≤ R2 ≤1

If R2<¼ then offspring Genotype is AA
if ¼≤R2<¾ then offspring Genotype is AB
if R2≥¾ then offspring genotype is BB

Offspring Genotype: BB
Offspring Genotype: AB
Offspring Genotype: AA

Generate Random Number 0≤ R3 ≤1

R3 < q0 if BB
R3 < q1 if AB
R3 < q2 if AA

Y
N

Offspring is not affected
Offspring is not affected

Offspring is affected

Figure 2-2 Offspring’s genotype simulation
CHAPTER 3

SIMULATION RESULTS

3.1 Performance Comparison

For this study, we generated 50, 100, 200, 300, 500 and 800 three-sibling families under each inheritance model with parameters $P_A = \{0.2, 0.5\}$, $K = \{0.01, 0.05\}$ and $AF=0.5$. Average p-values under each set of simulation conditions are shown in Table 3-1. The asymptotic test seems to consistently underestimate the exact p-value. Smaller sample sizes seem to yield a larger difference between large-sample FBAT and XFBAT.

Figure 3-1. Plots of exact and asymptotic p-values.
<table>
<thead>
<tr>
<th>MODEL</th>
<th>Nfams</th>
<th>Sibs</th>
<th>PA</th>
<th>K</th>
<th>FBAT</th>
<th>XFBAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDITIVE</td>
<td>50</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.8955</td>
<td>1.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.5254</td>
<td>0.5710</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0780</td>
<td>0.0796</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.3071</td>
<td>0.3316</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.8055</td>
<td>0.8536</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0608</td>
<td>0.0744</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.1365</td>
<td>0.1570</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0817</td>
<td>0.0920</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0011</td>
<td>0.0012</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0001</td>
<td>0.0000</td>
</tr>
<tr>
<td>DOMINANT</td>
<td>50</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0002</td>
<td>0.0002</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.1306</td>
<td>0.1606</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0593</td>
<td>0.0760</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.1569</td>
<td>0.2007</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0001</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0005</td>
<td>0.0006</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>RECESSIVE</td>
<td>50</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>#</td>
<td>0.4592</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0087</td>
<td>0.0107</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0253</td>
<td>0.0393</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0001</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0006</td>
<td>0.0007</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0007</td>
<td>0.0011</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0001</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>3</td>
<td>0.2</td>
<td>0.01</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.5</td>
<td>0.05</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
</tbody>
</table>
3.2 Efficiency Comparison

We combined the data simulation procedure with a PERL program to determine computation time for the two tests (see Appendix III). The CPU time for running each of the two tests for six different simulated data sizes is displayed in Table 3-2. For a given dataset, we ran each test five times to compute average CPU time, rounded to the nearest microsecond. As expected, the exact test takes substantially longer than the asymptotic test. However, what we are interested in is how much longer the exact test costs in light of its more accurate calculation of p-value. In contrast to the relatively uniform performance of the asymptotic test for increasing sample size, computational time for XFBAT increases exponentially.

Figure 3-2 plots time versus sample size, with the time axis on the log-scale, to illustrate how relative computational efficiency changes as sample size increases. Figure 3-3 additionally illustrates these differences as ratios, plotting \( \ln(X/F) \) versus sample size, where \( X \) and \( F \) are the CPU runtimes of XFBAT and FBAT, respectively.
Table 3-2. FBAT and XFBAT Computation Time Results (in seconds)

<table>
<thead>
<tr>
<th>SIZE</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>500</th>
<th>800</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBAT</td>
<td>0.023317</td>
<td>0.033868</td>
<td>0.054129</td>
<td>0.055290</td>
<td>0.063779</td>
<td>0.097872</td>
</tr>
<tr>
<td>XFBAT</td>
<td>0.575000</td>
<td>2.200000</td>
<td>6.615625</td>
<td>15.234375</td>
<td>38.971875</td>
<td>110.359375</td>
</tr>
<tr>
<td>X-F</td>
<td>0.551683</td>
<td>2.166132</td>
<td>6.561496</td>
<td>15.179085</td>
<td>38.908096</td>
<td>110.261503</td>
</tr>
<tr>
<td>X/F</td>
<td>24.660119</td>
<td>64.958073</td>
<td>122.219605</td>
<td>275.535811</td>
<td>611.045564</td>
<td>1127.588841</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIZE</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>500</th>
<th>800</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBAT</td>
<td>0.027438</td>
<td>0.028094</td>
<td>0.035472</td>
<td>0.044387</td>
<td>0.057593</td>
<td>0.088073</td>
</tr>
<tr>
<td>XFBAT</td>
<td>0.600000</td>
<td>1.584375</td>
<td>6.206250</td>
<td>13.206250</td>
<td>36.702765</td>
<td>88.018750</td>
</tr>
<tr>
<td>X-F</td>
<td>0.572562</td>
<td>1.556281</td>
<td>6.170778</td>
<td>13.161863</td>
<td>36.645172</td>
<td>87.930677</td>
</tr>
<tr>
<td>X/F</td>
<td>21.867483</td>
<td>56.395494</td>
<td>174.961942</td>
<td>297.525176</td>
<td>637.278228</td>
<td>999.384034</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIZE</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>500</th>
<th>800</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBAT</td>
<td>0.023437</td>
<td>0.025952</td>
<td>0.036435</td>
<td>0.047775</td>
<td>0.061469</td>
<td>0.094672</td>
</tr>
<tr>
<td>XFBAT</td>
<td>0.431686</td>
<td>1.259375</td>
<td>3.562500</td>
<td>6.846875</td>
<td>16.871875</td>
<td>40.543750</td>
</tr>
<tr>
<td>X-F</td>
<td>0.408249</td>
<td>1.233423</td>
<td>3.526065</td>
<td>6.799100</td>
<td>16.810406</td>
<td>40.449078</td>
</tr>
<tr>
<td>X/F</td>
<td>18.418996</td>
<td>48.527088</td>
<td>97.776863</td>
<td>143.315018</td>
<td>274.477786</td>
<td>428.254922</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIZE</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>500</th>
<th>800</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBAT</td>
<td>0.024761</td>
<td>0.032030</td>
<td>0.048363</td>
<td>0.052022</td>
<td>0.063648</td>
<td>0.089008</td>
</tr>
<tr>
<td>XFBAT</td>
<td>0.453300</td>
<td>0.987500</td>
<td>2.206250</td>
<td>4.137500</td>
<td>9.187500</td>
<td>21.475000</td>
</tr>
<tr>
<td>X-F</td>
<td>0.428539</td>
<td>0.955470</td>
<td>2.157887</td>
<td>4.085478</td>
<td>9.123852</td>
<td>21.385992</td>
</tr>
<tr>
<td>X/F</td>
<td>18.307015</td>
<td>30.830471</td>
<td>45.618551</td>
<td>79.533659</td>
<td>144.348605</td>
<td>241.270448</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIZE</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>500</th>
<th>800</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBAT</td>
<td>0.021771</td>
<td>0.026838</td>
<td>0.048363</td>
<td>0.052022</td>
<td>0.063648</td>
<td>0.089008</td>
</tr>
<tr>
<td>XFBAT</td>
<td>0.482075</td>
<td>1.140625</td>
<td>3.600000</td>
<td>5.987500</td>
<td>15.765625</td>
<td>39.659375</td>
</tr>
<tr>
<td>X-F</td>
<td>0.460304</td>
<td>1.113787</td>
<td>3.565281</td>
<td>5.940324</td>
<td>15.707317</td>
<td>39.561519</td>
</tr>
<tr>
<td>X/F</td>
<td>22.142988</td>
<td>42.500373</td>
<td>103.689622</td>
<td>126.918348</td>
<td>270.385282</td>
<td>405.283018</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIZE</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>500</th>
<th>800</th>
</tr>
</thead>
<tbody>
<tr>
<td>FBAT</td>
<td>0.023599</td>
<td>0.024815</td>
<td>0.031324</td>
<td>0.043827</td>
<td>0.057366</td>
<td>0.090082</td>
</tr>
<tr>
<td>XFBAT</td>
<td>0.422833</td>
<td>0.903125</td>
<td>2.303125</td>
<td>5.253125</td>
<td>11.721875</td>
<td>28.171875</td>
</tr>
<tr>
<td>X-F</td>
<td>0.399234</td>
<td>0.878310</td>
<td>2.271801</td>
<td>5.209298</td>
<td>11.664509</td>
<td>28.081793</td>
</tr>
<tr>
<td>X/F</td>
<td>17.917412</td>
<td>36.394318</td>
<td>73.525891</td>
<td>119.860474</td>
<td>204.334885</td>
<td>312.735896</td>
</tr>
</tbody>
</table>
Figure 3-2. CPU computation time comparison between FBAT and XFBAT.
The result of this study demonstrates that XFBAT using an additive model is significantly more burdensome computationally than XFBAT under either the dominant or recessive models. This discrepancy increases as the sample size becomes larger, as illustrated in Figure 3-4.
Our simulation study of biallelic family-based data demonstrates that FBAT consistently underestimates the exact p-values, particularly for small samples. However, while XFBAT yields exact p-values, its computational runtimes increase dramatically for larger sample sizes. In addition, XFBAT is much more computationally intensive when assuming the additive model as opposed to either the dominant or recessive models. This simulation study can be extended to evaluate the computational efficiency of the Monte Carlo exact p-value tool already available with the standard FBAT software. The Monte Carlo approach provides an unbiased estimate of the exact p-value (as opposed to the biased estimate yielded by the large-sample approximation), at a computational cost that is somewhat greater than asymptotic FBAT but presumably not as great as XFBAT.
REFERENCES


Appendix I. Data Simulation in R Code.

```r
pedisimu=function(familynum,childnum,PA,K,AF,model) {
  # generate parents' genotype
  child=rep(0,childnum)
  c=childnum*familynum
  childID=rep(0,c)
  familyID=rep(0,c)
  fatherID=rep(0,c)
  motherID=rep(0,c)
  affectstat=rep(0,c)
  sex=rep(0,c)
  allel.1=rep(0,c)
  allel.2=rep(0,c)
  PB=1-PA
  j=1
  while (j <= familynum){
    parent=rep(0,2)
    parent.allel = runif(2)
    for (i in 1:2){
      if (parent.allel[i] < (PA*PA)) parent[i] = 2
      if (parent.allel[i] < (1-PB*PB) && parent.allel[i] >= (PA*PA)) parent[i] = 1
      if (parent.allel[i] >= (1-PB*PB)) parent[i] = 0
    }
    # determine offsprings' genotype
      for (i in 1:childnum) child[i] = 2
    }
    if ((parent[1] == 0 && parent[2] == 0)) {
      for (i in 1:childnum) child[i] = 0
    }
      for (i in 1:childnum) child[i] = 1
    }
      for (i in 1:childnum) child[i] = 1
    }
    child.allel = runif(1)
  }
}
```

APPENDIX
if (child.allel < 0.5) {
    child[i] = 2
} else {child[i] = 1}

    for (i in 1:childnum) {
        child.allel = runif(1)
        if (child.allel < 0.25) {
            child[i] = 2
        } else if ((child.allel >= 0.25) && (child.allel < 0.75)) {
            child[i] = 1
        } else {child[i] = 0}
    }
}

    for (i in 1:childnum) {
        child.allel = runif(1)
        if (child.allel < 0.5) {
            child[i] = 0
        } else {child[i] = 1}
    }
}

# Calculate penetrance function
q0 = K * (1 - AF)

# Additive model
if (model == "add") {
    q2 = q0 + (K - q0) / PA
    q1 = (q0 + q2) / 2
}

# Dominant model
if (model == "dom") {
    q2 = ((1 - PA)^2 * q0 - K) / ((1 - PA)^2 - 1)
    q1 = q2
}

# Recessive model
else {
    q2 = (K - q0) / PA + q0
    q1 = q0
}

sum affect = 0
#determine the affection status

```r
for (i in 1:childnum) {
  k = childnum*(j-1)+i
  familyID[k]=j
  fatherID[k]=100*j+0
  motherID[k]=100*j+1
  childID[k]=k
  #determine the sex
  s = runif(1)
  if (s<0.5) {sex[k]=1}  
                     else {sex[k]=2}
  if( child[i]==0 ){
    affect = runif(1)
    if( affect<q0) {affectstat[k]=2}
                     else {affectstat[k]=1}
    allel.1[k]=2
    allel.2[k]=2
  }
  if( child[i]==1 ){
    affect = runif(1)
    if( affect<q1) {affectstat[k]=2}
                     else {affectstat[k]=1}
    allel.1[k]=1
    allel.2[k]=2
  }
  if( child[i]==2 ){
    affect = runif(1)
    if( affect<q2) {affectstat[k]=2}
                     else {affectstat[k]=1}
    allel.1[k]=1
    allel.2[k]=1
  }
  sumaffect = affectstat[k] + sumaffect
}
#Make sure at least one offspring is affected
if (sumaffect > 3) {j=j+1}
pedigree = cbind(familyID,childID,fatherID,motherID,sex,affectstat,allel.1, allel.2)
```

#example

#set.seed(16)
x=pedisimu(800,3,0.5,0.05,0.5,"rec")
pedigreefile<-write.table(x,file="F:/simulation/800R-PA0.5-K0.05",row.names=FALSE,col.names=FALSE)
#check
sum(affectstat==2)/c
(sum(allele.1==2)+sum(allele.2==2))/(2*c)

Appendix II. A sample of simulation data file

F50A-PA20.PED

gene50
1 1 100 101 2 1 1 2
1 2 100 101 2 2 1 1
1 3 100 101 1 1 1 2
2 4 200 201 1 1 1 2
2 5 200 201 1 2 1 2
2 6 200 201 1 1 1 2
3 7 300 301 1 1 1 2
3 8 300 301 1 1 1 2
3 9 300 301 1 1 1 2
4 10 400 401 1 1 1 2
4 11 400 401 1 2 1 2
4 12 400 401 1 1 1 2
5 13 500 501 1 1 1 2
5 14 500 501 1 1 1 2
5 15 500 501 1 1 2 2
6 16 600 601 1 1 2 2
6 17 600 601 1 2 2 2
6 18 600 601 1 2 2 2
7 19 700 701 1 1 2 2
7 20 700 701 1 2 2 2
7 21 700 701 1 2 2 2
8 22 800 801 1 2 2 2
8 23 800 801 1 1 2 2
8 24 800 801 2 1 1 2
9 25 900 901 2 1 2 2
9 26 900 901 2 2 2 2
9 27 900 901 2 1 2 2
.....
Appendix III. Perl Code for Time Comparison between FBAT and XFBAT

1. Xfbat.pl

use Time::HiRes;
use File::Basename;
foreach $file (@ARGV)
{
  if (! -e $file)
  {
    print "$file does not exist
    next;
  }
  # Strip off the leading directory path
  $base = basename($file);
  # Strip off the .ped suffix
  $base =~ s/\.ped$//;
  # Construct the log file name
  $log_x = $base . "_xtbat.log";
  # Measure time under a second
  # record the system start time
  $start_xtbat=Time::HiRes::time;
  # open pipe to xfat
  open(XOUT,"| xfat") or die "Could not open xfat: $!
  ;
  # Print commands to xfat
  print XOUT "$file
  print XOUT "\n
  print XOUT "$file\n
  print XOUT "\n
  print XOUT "1\n
  print XOUT "1\n
  print XOUT "1\n
  print XOUT "3\n
  print XOUT "$log_x\n
  # execute command XFBAT
  print XOUT "$\n
  print XOUT "$\n
  print XOUT "$\n
  # Close the pipe to xfat
  close(XOUT);
  # record the system end time
  $end_xtbat=Time::HiRes::time;
  printf "xtbat start time %.6f", $start_xtbat, "\n
  printf "xtbat end time %.6f", $end_xtbat, "\n";
2. fbat.pl

use Time::HiRes;
use File::Basename;

foreach $file (@ARGV)
{
    if (! -e $file)
    {
        print "$file does not exist\n";
        next;
    }

    # Strip off the leading directory path
    $base = basename($file);

    # Strip off the .ped suffix
    $base =~ s/\ped$//;

    # Construct the log file name
    $log_f = $base . "_fbat.log";

    # Measure time under a second
    # record the system start time
    $start_fbat = Time::HiRes::time;

    # Open pipe to fbat
    open(hOUT, "| fbat") or die "Could not open fbat: $!
"
;

    # Print commands to fbat
    print hOUT "load $file\n";
    print hOUT "log $log_f\n";
    print hOUT "mode m\n";
    print hOUT "model a\n"; # A: additive D: Dominant R: Recessive

    # execute command FBAT
    print hOUT "fbat\n";
    print hOUT "quit\n";

    # Close the pipe to fbat
    close(hOUT);

    # record the system end time
    $end_fbat = Time::HiRes::time;

    printf "fbat took \%.6f CPU second of user time\n", $end_fbat - $start_fbat;
}

# Success
exit(0);
print STDOUT "\n";
printf "fbat end time % .6f", $end_fbat;
print STDOUT "\n";
printf "fbat took % .6f CPU second of user time\n",$end_fbat-$start_fbat,"\n";
print STDOUT "\n";
}

# Success
exit(0);